

RUHR-UNIVERSITÄT BOCHUM

COMPSAC  **2011**

RUB

SAPSE 2011: Expressiveness Considerations of XML Signatures

- a duck tale -

18.07.2011

Horst Görtz Institute for IT-Security | Chair for Network and Data Security
Department of Electrical Engineering and Information Sciences



Horst Görtz Institute for IT-Security | Chair for Network and Data Security

Expressiveness Considerations of XML Signatures

Meiko Jensen, Christopher Meyer

XML Signature

A brief introduction

XML Signature

A brief introduction

Two base operations:

- $\text{Sign}(\text{input}, \text{key}_{\text{private}})$
- $\text{Verify}(\text{signature}, \text{input}, \text{key}_{\text{public}})$

Security goals:

- Integrity
- Authenticity
- Non-repudiation

```

<Security>
  <Signature>
    <SignedInfo>
      <CanonicalizationMethod Algorithm="..."/>
      <SignatureMethod Algorithm="..."/>
      <Reference URI="#myID">
        <Transforms>
          <Transform Algorithm="..."/>
        </Transforms>
        <DigestMethod Algorithm="..."/>
        <DigestValue>JVxbSj...</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>d4Uf6...</SignatureValue>
  </Signature>
</Security>
<op:opBlock Id="myID" xmlns:op="...">
  <op:greet>
    <name>Scrooge McDuck</name>
  </op:greet>
</op:opBlock>

```

Horst Görtz Institute for IT-Security | Chair for Network and Data Security

Expressiveness Considerations of XML Signatures

Meiko Jensen, Christopher Meyer

SOAP

A brief introduction

SOAP

A brief introduction

Purpose:

- Structured transfer of data
- Remote Procedure Calls

Usage scenarios:

- Web-Service communication

```
<soap:Envelope xmlns:soap="...">
  <soap:Header>
    <!-- optional SOAP Header information -->
  </soap:Header>
  <soap:Body>
    <!-- mandatory SOAP Body payload -->
  </soap:Body>
</soap:Envelope>
```

May be combined with XML Signature

XML Signature secured SOAP communication

Usage example: *Access the money bin*



Scrooge McDuck

[<http://duckman.pettho.com/characters/scrooge.jpg>]



Money Bin

[<http://duckman.pettho.com/characters/moneybin.jpg>]

XML Signature secured SOAP communication

Usage example: *Access the money bin*



Scrooge McDuck

[<http://duckman.pettho.com/characters/scrooge.jpg>]

Let me in!
I need my daily
money bath!

Name:

Scrooge McDuck

SOAP request

Action:

MoneyBath

Signature:

Scrooge McDuck



Money Bin

[<http://duckman.pettho.com/characters/moneybin.jpg>]

XML Signature secured SOAP communication

Usage example: *Access the money bin*



Scrooge McDuck

[<http://duckman.pettho.com/characters/scrooge.jpg>]



Money Bin

[<http://duckman.pettho.com/characters/moneybin.jpg>]

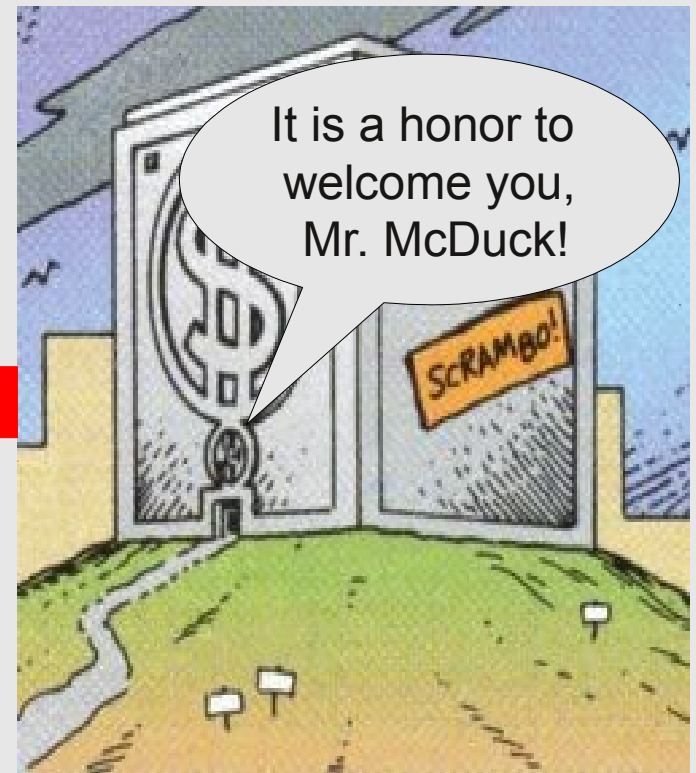
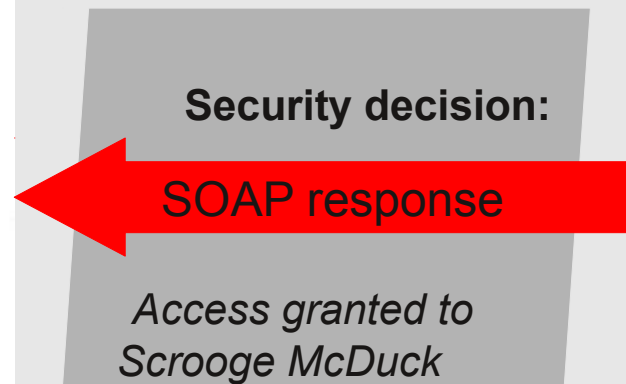
XML Signature secured SOAP communication

Usage example: *Access the money bin*



Scrooge McDuck

[<http://duckman.pettho.com/characters/scrooge.jpg>]



Money Bin

[<http://duckman.pettho.com/characters/moneybin.jpg>]


Where is the problem?

Example: Enveloped Signature

Where is the problem?

Example: Enveloped Signature

```
<Document Id="signMe">
  <Data>I'm important!</Data>
  <Signature>
    <SignedInfo>
      ...
      <Reference URI="#signMe">
        ...
        <DigestValue>
          ...Saduiohoiwefdqw87euovhqi23...
        </DigestValue>
      </Reference>
    </SignedInfo>
    ...
  </Signature>
</Document>
```

A red rectangular box highlights the attribute `Id="signMe"` in the opening `<Document>` tag. A red line extends from the bottom of this box, moving down and then left to point at the attribute `URI="#signMe"` in the `<Reference>` tag, illustrating the self-referencing nature of the signature.

Where is the problem?

Example: Enveloped Signature

```
<Document Id="signMe">  
  <Data>I'm important!</Data>  
  <Signature>  
    <SignedInfo>  
      ...  
      <Reference URI="#signMe">  
        ...  
        <DigestValue>  
          ...Saduiohoiwefdqw87euovhqi23...  
        </DigestValue>  
      </Reference>  
    </SignedInfo>  
    ...  
  </Signature>  
</Document>
```

What if only **single document parts** should be used?
Signature will be invalidated when removing particular parts!

XML Signature secured SOAP communication

Usage example: *Access the money bin*

XML Signature secured SOAP communication

Usage example: *Access the money bin*



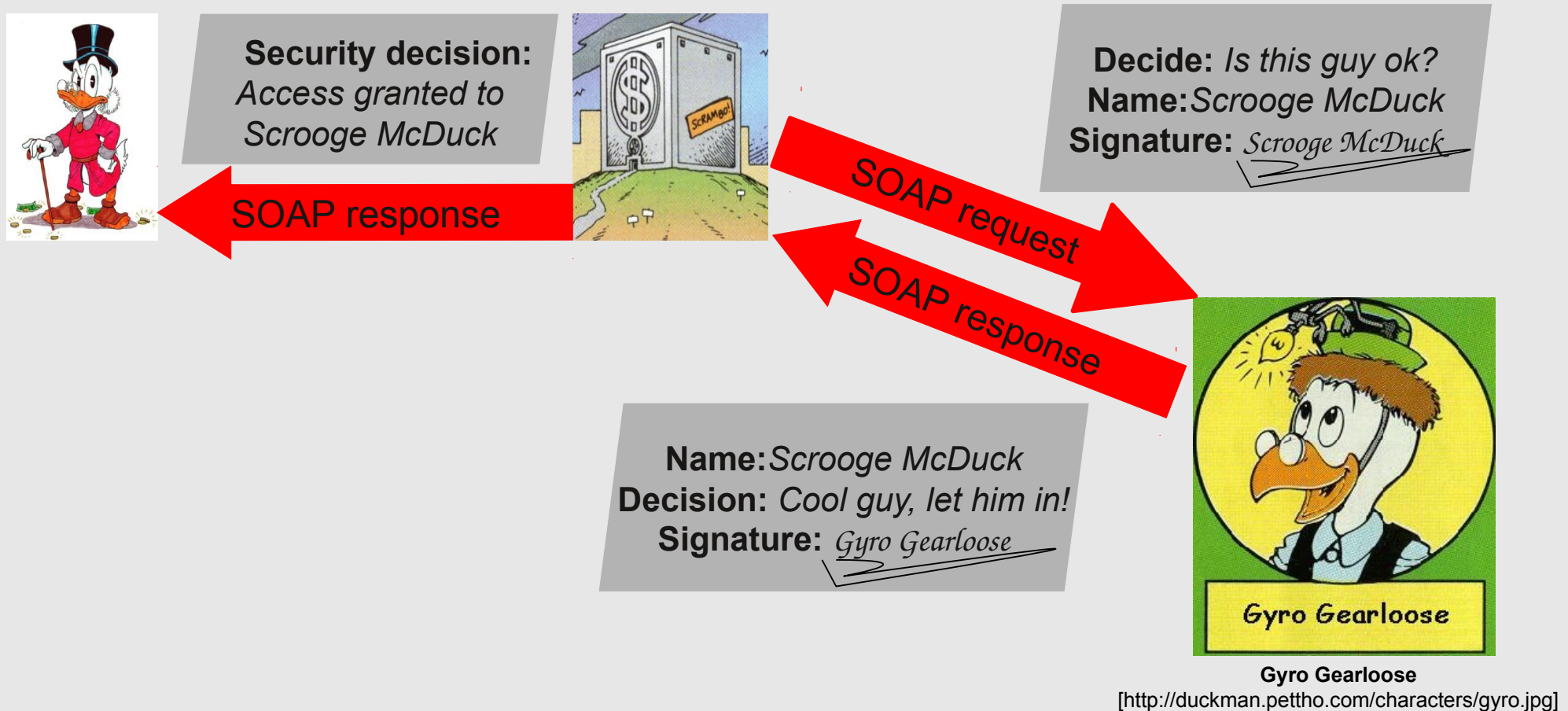
Security decision:
*Access granted to
Scrooge McDuck*



SOAP response

XML Signature secured SOAP communication

Usage example: Access the money bin



Signature invalidation when removing single parts

Consequence

Signature invalidation when removing single parts

Consequence

```
<soap:Envelope xmlns:soap="...">
  <soap:Header>..... </soap:Header>
  <soap:Body xmlns:op="http://my.operation">
    <op:opBlock Id="authBlock">
      <op:auth>
        <name>Scrooge McDuck</name>
      </op:auth>
    </op:opBlock>
    <op:opBlock Id="actionBlock">
      <op:action>
        <MoneyBath />
      </op:action>
    </op:opBlock>
  </soap:Body>
</soap:Envelope>
```



Multiple referenced objects necessary

How to protect multiple references

Combined signature vs. several signatures

How to protect multiple references

One Signature, N References - N Signatures, One Reference

```
<Signature>
  <SignedInfo>
    <CanonicalizationMethod Algorithm="..."/>
    <SignatureMethod Algorithm="..."/>

    <Reference URI="#authBlock">
      ...
    </Reference>

    <Reference URI="#actionBlock">
      ...
    </Reference>

  </SignedInfo>
  <SignatureValue>...</SignatureValue>
</Signature>
```

```
<Signature>
  <SignedInfo>
    <CanonicalizationMethod Algorithm="..."/>
    <SignatureMethod Algorithm="..."/>
    <Reference URI="#authBlock">
      ...
    </Reference>
  </SignedInfo>
  <SignatureValue>...</SignatureValue>
</Signature>

<Signature>
  <SignedInfo>
    <CanonicalizationMethod Algorithm="..."/>
    <SignatureMethod Algorithm="..."/>
    <Reference URI="#actionBlock">
      ...
    </Reference>
  </SignedInfo>
  <SignatureValue>...</SignatureValue>
</Signature>
```

How to protect multiple references

One Signature, N References - N Signatures, One Reference

- + lower overhead
- + easier processing
- must be processed in complete

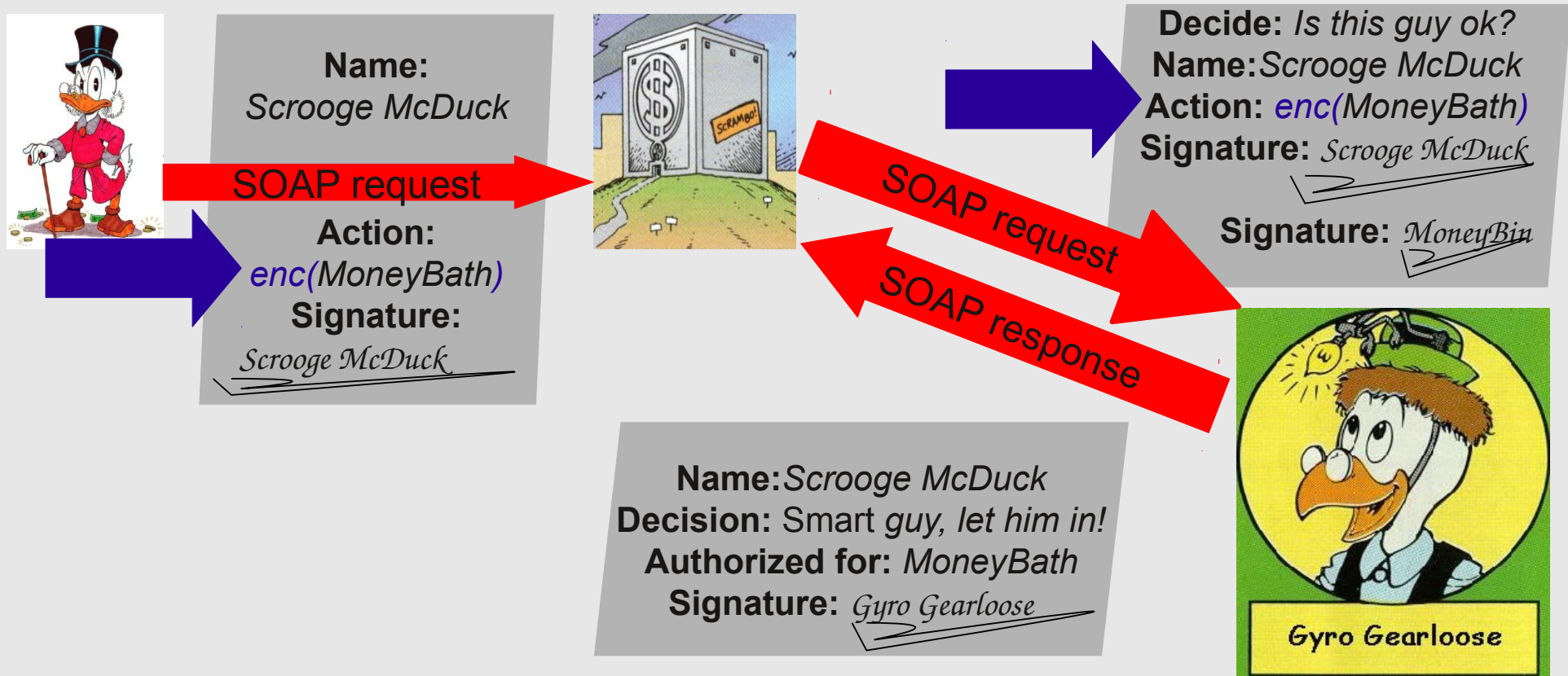
- + re-use of particular parts
- +/- decoupling of message parts
- more complex processing

What about confidentiality?

Signature concepts for additional encryption steps

What about confidentiality?

Signature concepts for additional encryption steps



What about confidentiality?

Signature concepts for additional encryption steps

What about confidentiality?

Signature concepts for additional encryption steps

Sign-then-Encrypt

- depending on encryption algorithm, ciphertext is deterministic
 - identical messages have the same hash value
 - correlation between output and e.g. operation possible
- the recipient can not be sure who applied the encryption



Confidentiality assumption weakened

What about confidentiality?

Signature concepts for additional encryption steps

Encrypt-then-Sign

- encrypted content may be *unreachable* to the signature application
- signer may possibly not know what he is signing
(if encryption took place earlier)
 - violation of *see-what-is-signed* paradigm
 - signer may only have seen the encrypted content



Non-repudiation assumption weakened

What about confidentiality?

Signature concepts for additional encryption steps

Sign-then-Encrypt-then-Sign

- combines advantages from both approaches
 - first signature provides integrity, authenticity, and non-repudiation
 - encryption provides confidentiality (over the payload AND its signature)
 - outer signature proves that message originator applied encryption
- not standardized yet
- more complex

Sign-then-Encrypt-then-Sign

Enhanced processing steps

Sign-then-Encrypt-then-Sign

Enhanced processing steps

Idea: *chain inner and outer signature*

- 1) Compute a hash value over the data that should be protected and store it temporarily
- 2) Encrypt the already processed block
- 3) Hash again the relevant block (which is now encrypted)
- 4) Take both pre-computed hash values and sign them

Other problems where XML Signatures are useful

Usage example: *Prevent adding SOAP Headers*

Other problems where XML Signatures are useful

Usage example: *Prevent adding SOAP Headers*



Other problems where XML Signatures are useful

Usage example: *Prevent adding SOAP Headers*



SOAP request

Name:
Scrooge McDuck
Action:
MoneyLevel
Signature:
Scrooge McDuck



SOAP request

Name:
Scrooge McDuck
Action:
MoneyLevel
Signature:
Scrooge McDuck
ReplyTo:
The Beagle Boys



The Beagle Boys

[<http://duckman.pettho.com/characters/beagleb.jpg>]

Other problems where XML Signatures are useful

Usage example: *Prevent adding SOAP Headers*



SOAP request

MoneyLevel:
90 ft.



SOAP response

Name:
Scrooge McDuck
Action:
MoneyLevel
Signature:
Scrooge McDuck



SOAP request

Name:
Scrooge McDuck
Action:
MoneyLevel
Signature:
Scrooge McDuck
ReplyTo:
The Beagle Boys

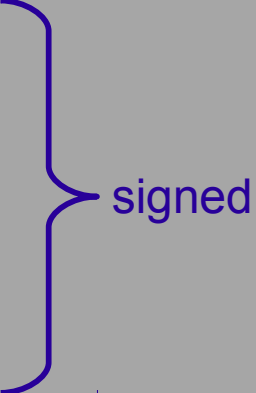
The Beagle Boys

[<http://duckman.pettho.com/characters/beagleb.jpg>]

Other problems where XML Signatures are useful

Usage example: *Prevent adding SOAP Headers*

```
<SupportedSpecs soap:mustUnderstand="true" xmlns="http://...specNegotiation/"  
  xmlns:soap="http://...soap-envelope">  
  <SpecificationSuite URI="http://...specNeg.../XML+NS+SOAP+WSDLSuite" />  
  <Specification URI="http://.../XPath" Version="1.0" />  
  <Specification URI="http://.../WS-Addressing" Version="1.0" />  
  <Specification URI="http://.../WS-Security" Version="1.1" />  
  <Specification URI="http://.../SAML" Version="2.0" />  
</SupportedSpecs>
```



signed

Conclusion

Results of the latest research

- **XML Signature is complex to use**
 - One Signature with N References vs. N Signatures with One Reference
- **Attacker may change message semantics**
 - Injection of non-signed processing advises (e.g. *WS-Addressing* header)
 - Possible countermeasure:
 - Explicitly list capabilities of the client at time of message creation, and sign that information
- **Interplay between signatures and encryption is particularly challenging**
 - Best approach: **Sign-then-Encrypt-then-Sign**
 - Signature on plaintext* (proof of integrity and knowledge)
 - AND**
 - Signature on ciphertext* (proof of encryption)
 - Complex to use, complex to implement

Future Work

Open tasks

- More in-depth analysis on interrelation of XML Signature with WS-* Specs
- Attacker Model development for Web Services scenarios
- Formalization of Expressiveness/Semantics of a given XML Signature

Further research necessary

Lack of long-term studies concerning skin compatibility



Taking a money bath

[<http://2.bp.blogspot.com/-8J7Vuk21xrY/TfY8ygBpYI/AAAAAAAAABSc/7rECOG-ag9o/s1600/scrooge-mcduck.jpg>]

Questions and discussion are welcome!

Horst Görtz Institute for IT-Security | Chair for Network and Data Security
Expressiveness Considerations of XML Signatures
Meiko Jensen, Christopher Meyer

Authors



Dipl.-Inform. Meiko Jensen
Research Assistant

Research topics: *Cloud Computing Security, SOA and Web Service Security, Business Process Security*

<http://www.nds.rub.de/chair/people/meiko-jensen/>
Meiko.Jensen@rub.de



Dipl.-Ing. Christopher Meyer
Research Assistant

Research topics: *XML Security, Java Security*

<http://www.nds.rub.de/chair/people/cmeyer/>
Christopher.Meyer@rub.de