hochschule für angewandte wissenschaften
FACHBEREICH ELEKTROTECHNIK UND INFORMATIK hamburg
university of applied sciences

Informatik Masterstudiengang SS 2005 – Anwendungen I

Web Service Security

Thies Rubarth

Übersicht

- Einleitung
- Secure Socket Layer
- XML Encryption & XML Signature
- WS-*
  - WS-Security
  - WS-Policy
  - WS-Trust
- Angebot für das Projekt

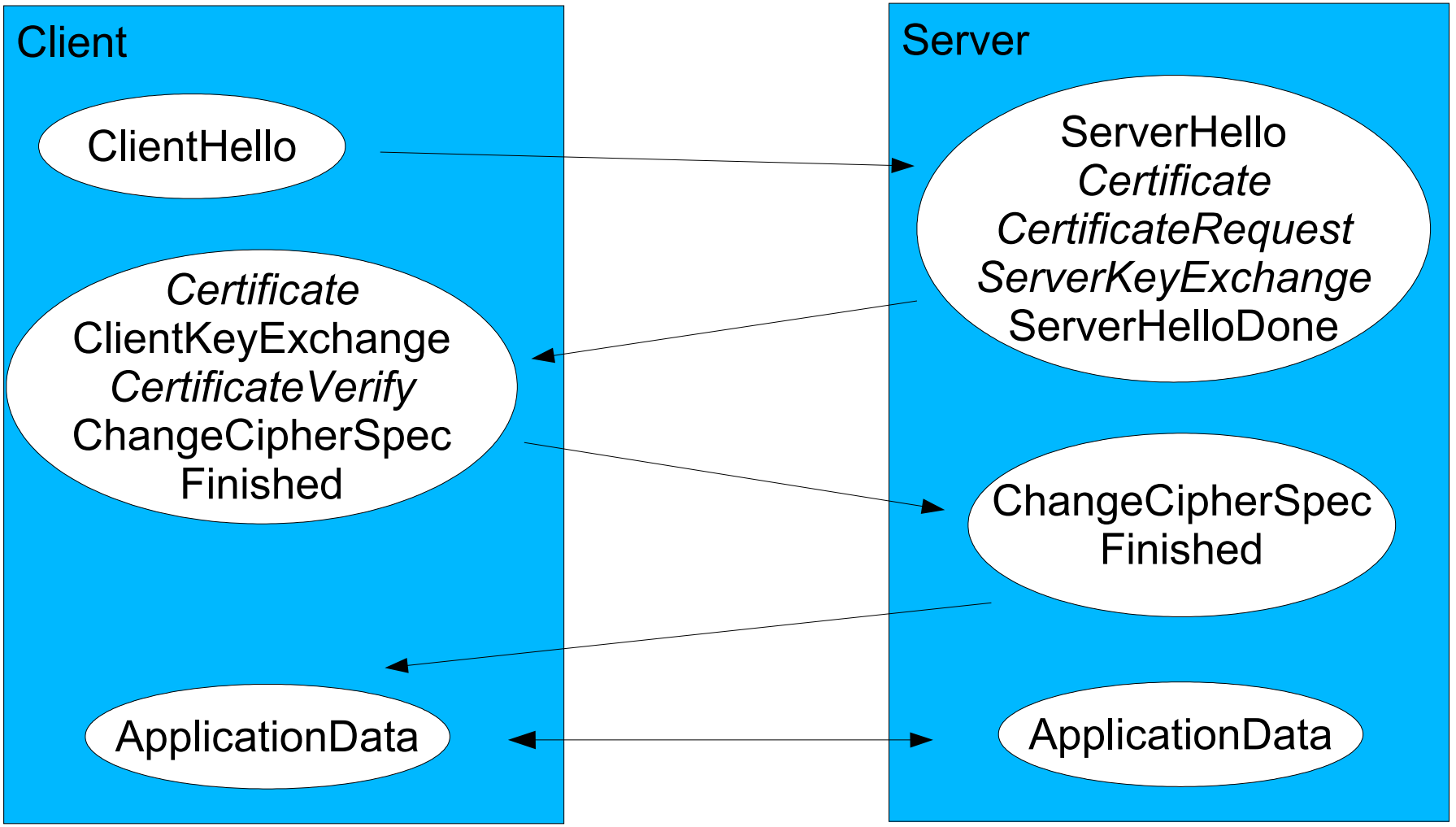
Einleitung

- Sicherheitsanforderungen
  - Authentifikation & Autorisierung
  - Vertraulichkeit & Integrität
- Web Services
  - „Offene“ Infrastruktur
  - Einfaches überwinden von Firewalls

Secure Socket Layer (SSL)

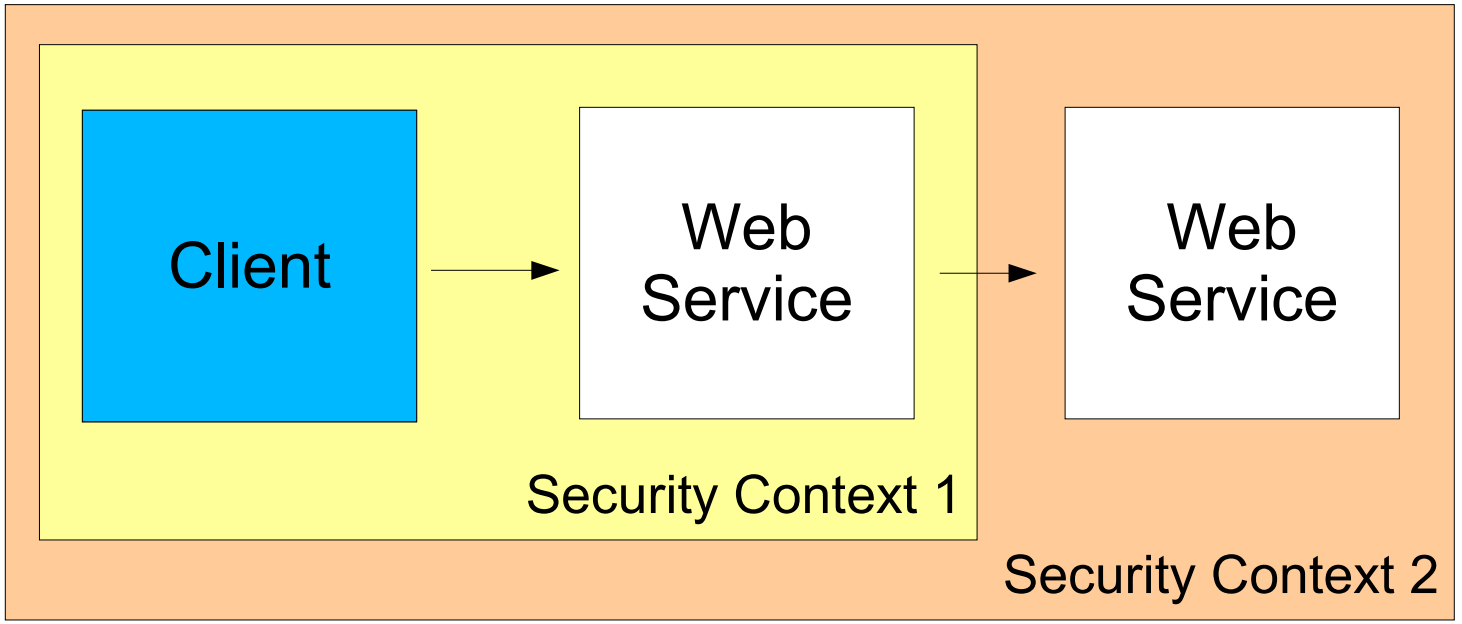
- Sichere Internet-Kommunikation über HTTPS
- Authentifizierung des Servers
- Verschlüsselte Datenübertragung
- Authentifizierung des Clients

SSL Handshake



Grenzen von SSL

- Keine Autorisierung
- Sicherheit nur zwischen zwei direkt kommunizierenden Partnern



XML-Encryption

- Partielle Verschlüsselung von XML-Dokumenten
- Meta-Informationen über verwendeten Algorithmus und Schlüssel
- Einbinden verschlüsselter Daten in das XML-Dokument

XML-Dokument Verschlüsseln

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <CreditCard Limit='5,000' Currency='USD'>
      <Number>4019 2445 0277 5567</Number>
      <Issuer>Example Bank</Issuer>
      <Expiration>04/02</Expiration>
    </CreditCard>
  </PaymentInfo>
```

```
<?xml version='1.0'?>
  <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
   MimeType='text/xml'>
    <CipherData>
      <CipherValue>A23B45C56</CipherValue>
    </CipherData>
  </EncryptedData>
```

XML-Element verschlüsseln

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <CreditCard Limit='5,000' Currency='USD'>
      <Number>4019 2445 0277 5567</Number>
      <Issuer>Example Bank</Issuer>
      <Expiration>04/02</Expiration>
    </CreditCard>
  </PaymentInfo>
```

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
     Type='http://www.w3.org/2001/04/xmlenc#Element'>
      <CipherData>
        <CipherValue>A23B45C56</CipherValue>
      </CipherData>
    </EncryptedData>
  </PaymentInfo>
```

Inhalt eines Elements verschlüsseln

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <CreditCard Limit='5,000' Currency='USD'>
      <Number>4019 2445 0277 5567</Number>
      <Issuer>Example Bank</Issuer>
      <Expiration>04/02</Expiration>
    </CreditCard>
  </PaymentInfo>
```

```
<?xml version='1.0'?>
  <PaymentInfo xmlns='http://example.org/paymentv2'>
    <Name>John Smith</Name>
    <CreditCard Limit='5,000' Currency='USD'>
      <EncryptedData xmlns='http://www.w3.org/2001/04/xmlenc#'
       Type='http://www.w3.org/2001/04/xmlenc#Content'>
        <CipherData>
          <CipherValue>A23B45C56</CipherValue>
        </CipherData>
      </EncryptedData>
    </CreditCard>
  </PaymentInfo>
```

"Super-Verschlüsselung"

```
<PaymentInfo xmlns:pay='http://example.org/paymentv2'>
    <EncryptedData Id='ED1' xmlns='http://www.w3.org/2001/04/xmlenc#'
     Type='http://www.w3.org/2001/04/xmlenc#Content'>
      <CipherData>
        <CipherValue>originalEncryptedData</CipherValue>
      </CipherData>
    </EncryptedData>
  </PaymentInfo>
```

```
<PaymentInfo xmlns:pay='http://example.org/paymentv2'>
    <EncryptedData Id='ED2' xmlns='http://www.w3.org/2001/04/xmlenc#'
     Type='http://www.w3.org/2001/04/xmlenc#Element'>
      <CipherData>
        <CipherValue>newEncryptedData</CipherValue>
      </CipherData>
    </EncryptedData>
  </PaymentInfo>
```

Meta-Informationen Verschlüsselung

```
<EncryptedData Id='ED'
   xmlns='http://www.w3.org/2001/04/xmlenc#'>
  <EncryptionMethod
           Algorithm='http://www.w3.org/2001/04/xmlenc#aes128-cbc'/>
  <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#'>
    <ds:RetrievalMethod URI='#EK'
         Type="http://www.w3.org/2001/04/xmlenc#EncryptedKey"/>
    <ds:KeyName>Sally Doe</ds:KeyName>
  </ds:KeyInfo>
  <CipherData><CipherValue>DEADBEEF</CipherValue></CipherData>
</EncryptedData>
<EncryptedKey Id='EK' xmlns='http://www.w3.org/2001/04/xmlenc#'>
   <EncryptionMethod
    Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5"/>
   <ds:KeyInfo xmlns:ds='http://www.w3.org/2000/09/xmldsig#'>
     <ds:KeyName>John Smith</ds:KeyName>
   </ds:KeyInfo>
   <CipherData><CipherValue>xyzabc</CipherValue></CipherData>
   <ReferenceList>
     <DataReference URI='#ED'/>
   </ReferenceList>
   <CarriedKeyName>Sally Doe</CarriedKeyName>
</EncryptedKey>
```

XML-Signature

- Partielles Signieren von XML-Dokumenten
- Meta-Informationen über Signierungen
- Einbinden der Signierung in XML-Dokumente

Aufbau einer XML-Signatur

Information über die Signatur

Referenz auf signiertes Objekt

Hash-Wert des Objektes

Kryptographischer Wert der Signatur

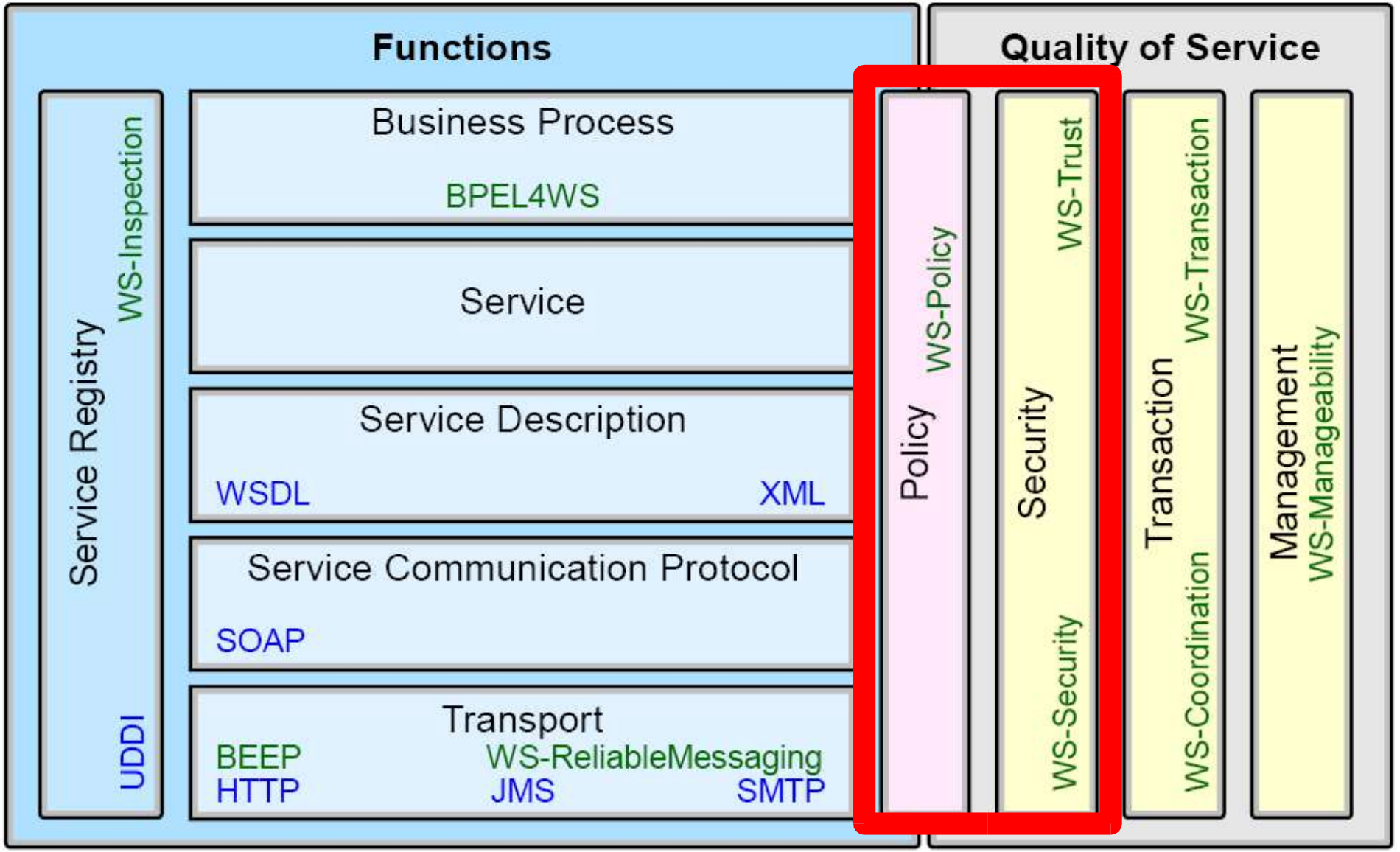
Informationen über den Schlüssel

Das signierte Objekt

```
<Signature>
   <SignedInfo>
      <CanonicalizationMethod .../>
      <SignatureMethod/>
      <Reference URI="...">
         <Transforms .../>
         <DigestMethod/>
         <DigestValue>
            j61wx3rvEOI8f...
         </DigestValue>
      </Reference>
   </SignedInfo>
   <SignatureValue>
      MC0r7Vff2Lk...
   </SignatureValue>
   <KeyInfo ... />
   <Object .../>
</Signature>
```

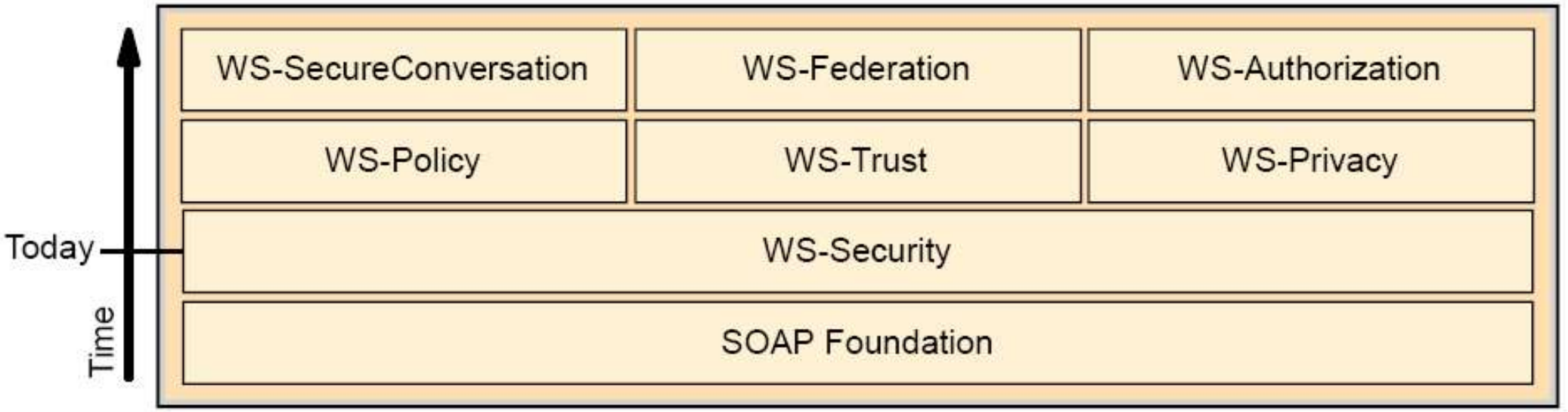
[1]

WS-* Spezifikationen



[8]

WS-* Verfügbarkeit (April 2004)



[8]

WS-Security

- Einbinden von Security-Tokens in SOAP-Nachrichten
- Integrität
- Vertraulichkeit
- Unabhängig von verwendeten Verfahren

WS-Security

- Keine Ziele
  - Sicherheitskontext über Austausch mehrerer Nachrichten
  - Schlüsselaustausch
  - Festlegen was „Vertrauen“ ist

WS-Security - UserName-Token

```
<S:Envelope xmlns:S="http://www.w3.org/2001/12/soap-envelope"
 xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext">
   <S:Header>
      <wsse:Security>
         <wsse:UsernameToken>
            <wsse:Username>Zoe</wsse:Username>
            <wsse:Password>ILoveDogs</wsse:Password>
         </wsse:UsernameToken>
      </wsse:Security>
      ...
   </S:Header>
   ...
   <S:Body>
      ... The SOAP Message ...
   </S:Body>
</S:Envelope>
```

WS-Security – Binary Security Tokens

- Benötigt für z.B.:
  - X.509-Zertifikate
  - Kerberos-Tickets
- Können in Signaturen referenziert werden

```
<wsse:BinarySecurityToken
 xmlns:wsse="http://schemas.xmlsoap.org/ws/2002/04/secext"
 Id="myToken" ValueType="wsse:X509v3"
 EncodingType="wsse:Base64Binary">
   MIIEZzCCA9CgAwIBAgIQEmtJZc0...
</wsse:BinarySecurityToken>
```

WS-Security - Signaturen

- Signaturen mit XML-Signature
- Signaturen müssen Elemente aus dem umschließenden S:Envelope referenzieren
- Enveloping Signatures dürfen nicht verwendet werden

WS-Security - Signaturbeispiel

```
<?xml version="1.0" encoding="utf-8"?>
<S:Envelope xmlns:S=".../soap-envelope" xmlns:ds=".../xmldsig#" xmlns:wsse=".../secext"
 xmlns:xenc=".../xmlenc#">
   <S:Header>
      <wsse:Security>
         <wsse:BinarySecurityToken ValueType="wsse:X509v3"
          EncodingType="wsse:Base64Binary" Id="X509Token">
            MIIEZzCCA9CgAwIBAgIQEmtJZc0rqrKh5i...
         </wsse:BinarySecurityToken>
         <ds:Signature>
            <ds:SignedInfo>
               <ds:SignatureMethod Algorithm=".../xmldsig#rsa-sha1"/>
               <ds:Reference>
                  <ds:DigestMethod Algorithm=".../xmldsig#sha1"/>
                  <ds:DigestValue>EULddytSo1...</ds:DigestValue>
               </ds:Reference>
            </ds:SignedInfo>
            <ds:SignatureValue>BL8jdfToEb1l/vXcMZNNjPOV...</ds:SignatureValue>
            <ds:KeyInfo>
               <wsse:SecurityTokenReference>
                  <wsse:Reference URI="#X509Token"/>
               </wsse:SecurityTokenReference>
            </ds:KeyInfo>
         </ds:Signature>
      </wsse:Security>
   </S:Header>
   <S:Body>
      ...
   </S:Body>
</S:Envelope>
```

WS-Security - Verschlüsselung

- Verschlüsselung mit XML-Encryption
- xenc:ReferenceList
  - Referenzieren der Verschlüsselten Elemente
- xenc:EncryptedKey
  - Angeben eines verschlüsselten Schlüssels
- xenc:EncryptedData
  - Einfügen der Verschlüsselten Daten

WS-Security – Verschlüsselung

```
<S:Envelope xmlns:S=".../soap-envelope" xmlns:ds=".../xmldsig#"
 xmlns:wsse=".../secext" xmlns:xenc=".../xmlenc#">
   <S:Header>
      <wsse:Security>
         <xenc:ReferenceList>
            <xenc:DataReference URI="#bodyID"/>
         </xenc:ReferenceList>
      </wsse:Security>
   </S:Header>
   <S:Body>
      <xenc:EncryptedData Id="bodyID">
         <ds:KeyInfo>
            <ds:KeyName>CN=Hiroshi Maruyama, C=JP</ds:KeyName>
         </ds:KeyInfo>
         <xenc:CipherData>
            <xenc:CipherValue>...</xenc:CipherValue>
         </xenc:CipherData>
      </xenc:EncryptedData>
   </S:Body>
</S:Envelope>
```

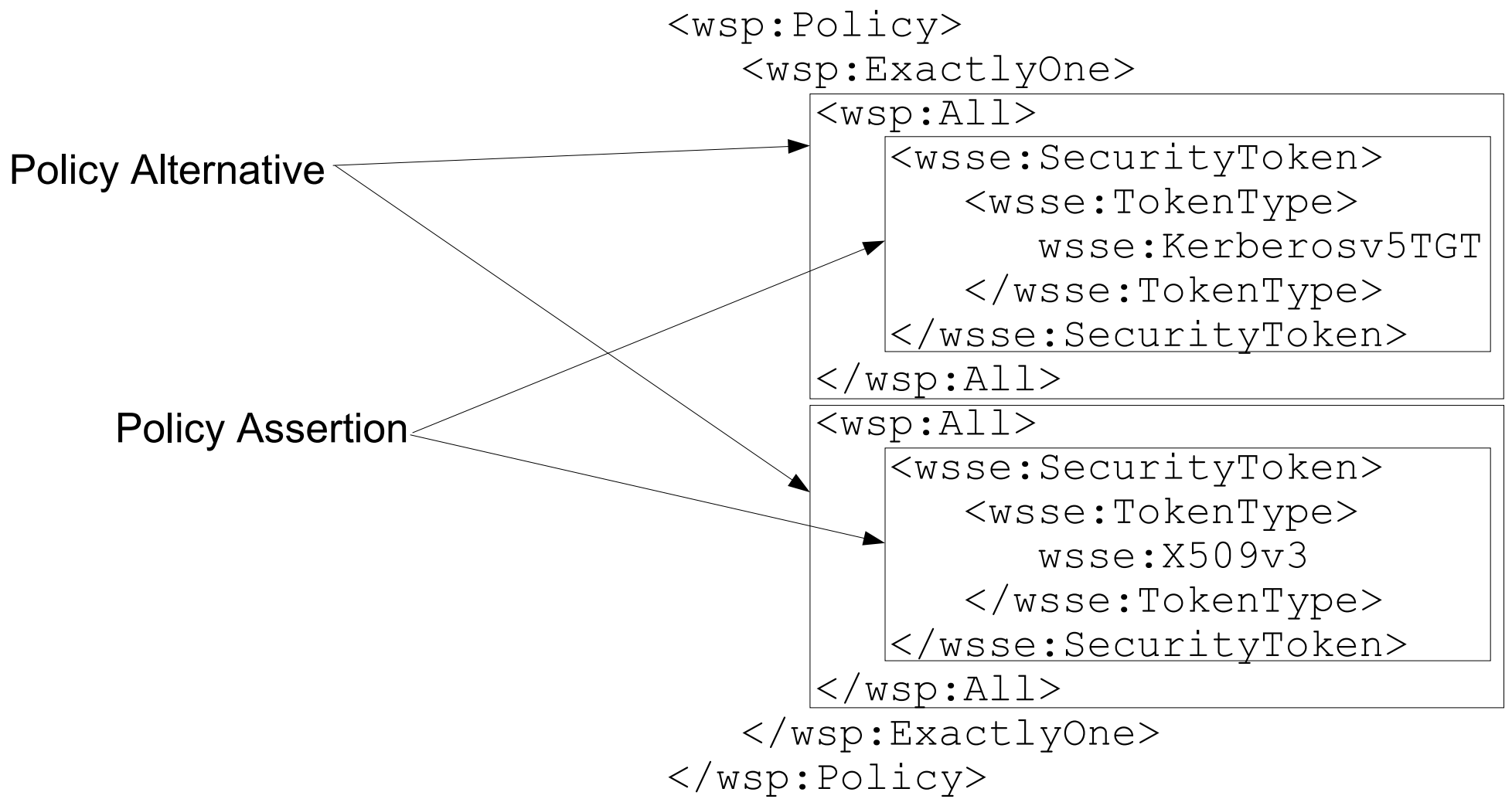
WS-Policy

- Beschreibung von Anforderungen eines Web Service
- Ausdrücken der Kombinationen von verschiedenen Anforderungen

WS-Policy Begriffe

- Policy
  - Sammlung von Policy Alternatives
- Policy Alternative
  - Sammlung von Policy Assertions
- Policy Assertion
  - Repräsentiert eine einzelne Anforderung

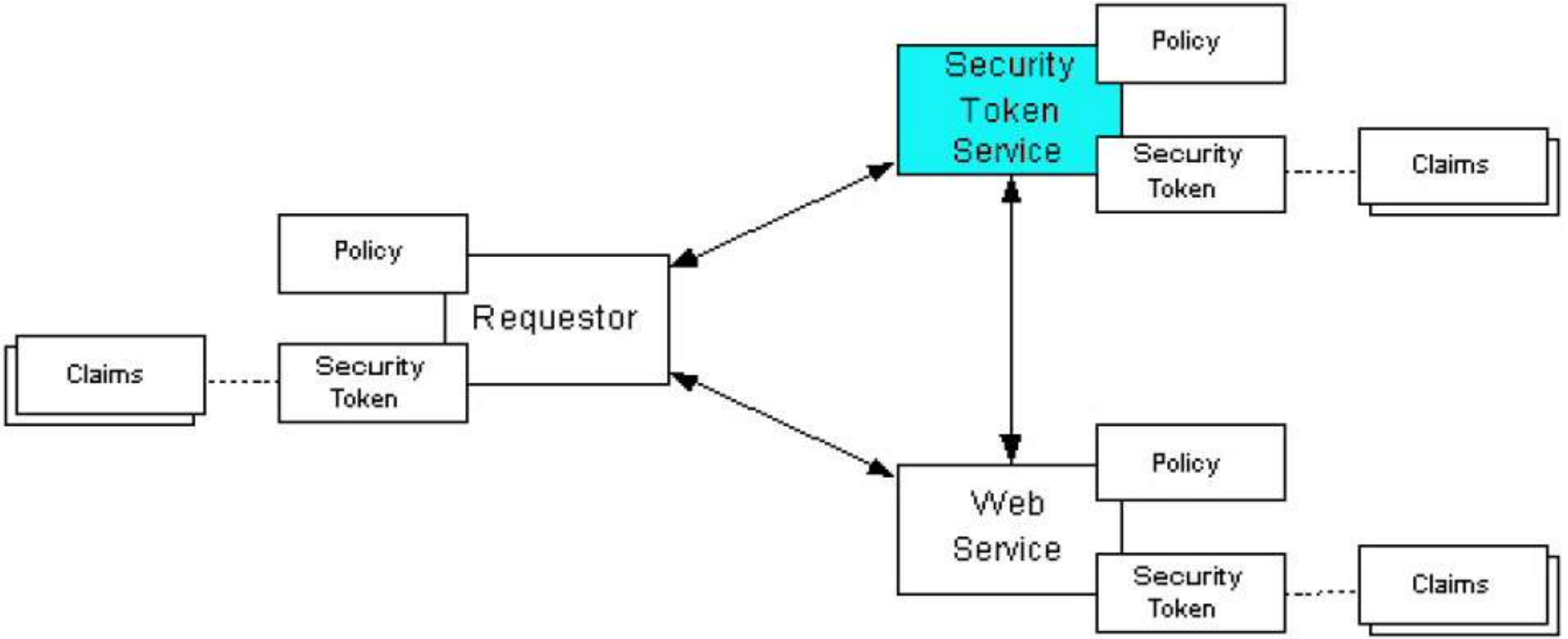
WS-Policy Beispiel



WS-Trust

- Erstellen vertrauenswürdiger SOAP-Nachrichten
  - Austausch und Vermittlung von Security Tokens
  - Protokoll unabhängige
    - Erstellung von Tokens
    - Erneuerung von Tokens
    - Validierung von Tokens

WS-Trust Model

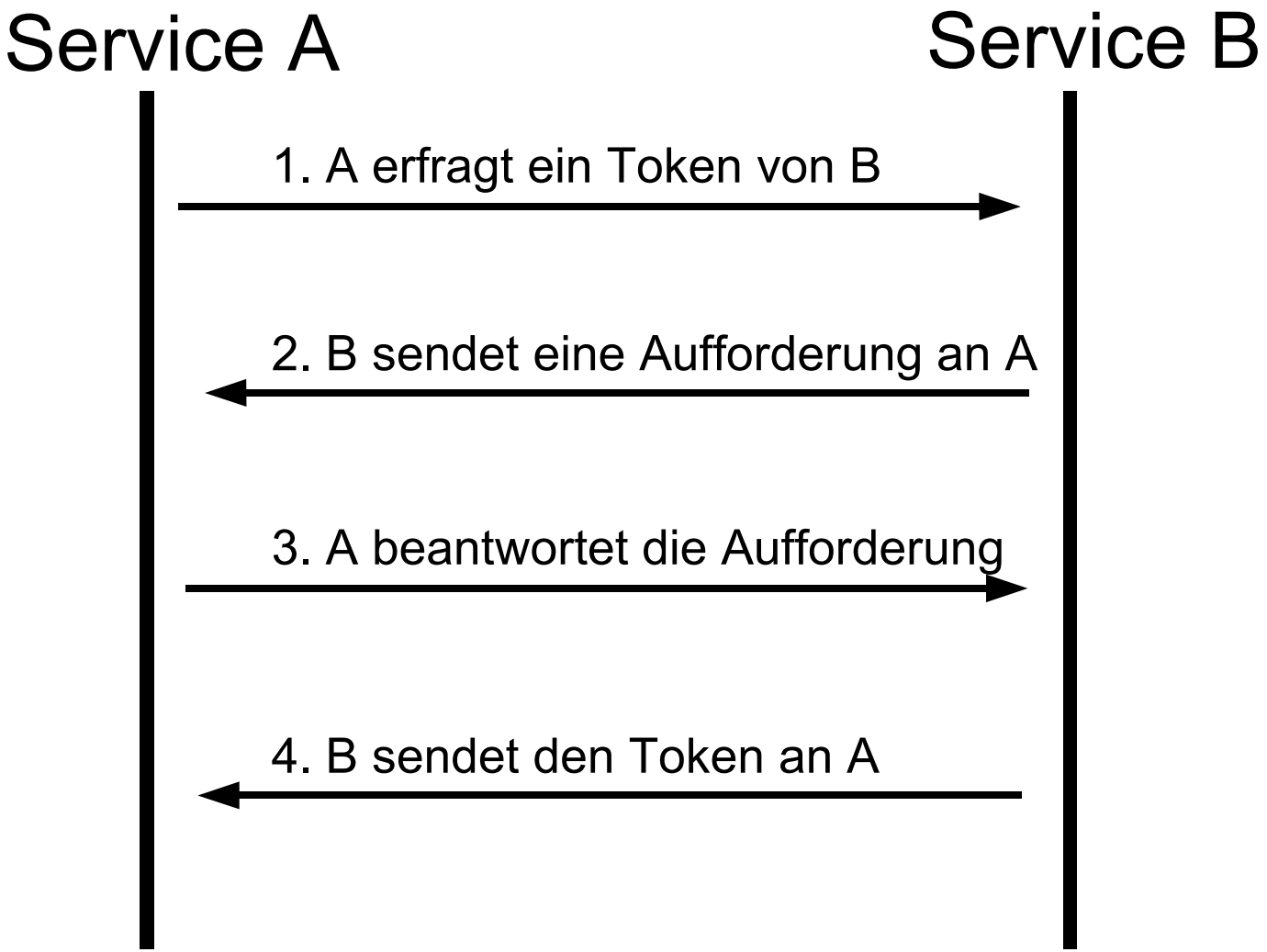


[4]

WS-Trust Bootstrap

- Fixed Trust Roots
  - Es gibt feste Dienste, denen vertraut wird
- Trust Hierarchies
  - Der Client muss die komplette Hierarchie liefern
  - Der Service kann die Hierarchie rekursiv ermitteln
- Authentication Service
  - Es gibt einen zentralen Dienst, bei dem Tokens überprüft werden können

WS-Trust Challenges



[4]

Zusammenfassung

- SSL bietet vertrauliche Kommunikation auf der Transportebene
- XML Encryption & XML Signature bieten Verschlüsselung und Signierung auf Nachrichtenebene
- WS-Security ermöglicht die Verwendung von Security-Tokens in SOAP-Messages
- WS-Policy ermöglicht die Ausführung von Web-Services an Bedingungen zu binden

Zusammenfassung - Fortsetzung

- WS-Trust bietet Framework für beliebige Sicherheitsarchitekturen (z.B. Kerberos)

Angebot für das Projekt

- Zusammen mit Martin, Sven und Tobias
  - Erstellen eines Trust-Konzepts
  - Erstellen einer Sicherheitsarchitektur
  - Zur Verfügung stellen der notwendigen Infrastruktur

Quellen

[1] Prof. Dr. Stefan Fischer - VL-Skript: Enterprise Applications (SS 2004)
[2] Bob Atkinson et. al. - WS-Security Version 1.0 (5. April 2002)
[3] Siddharth Bajaj et. al. - WS-Policy (September 2004)
[4] Steve Anderson et. al. - WS-Trust (Februar 2005)
[5] Alan O. Freier et. al. - The SSL Protocol Version 3.0 (Stand März 1996)
[6] Takeshi Imamura et. al. - XML Encryption Syntax and Processing (Stand 10. Dezember 2002)
[7] Mark Bartel et. al. - XML-Signature Syntax and Processing (Stand 12. Februar 2002)
[8] Mark Endrai et. al. - Patterns: Service- Oriented Architecture and Web Services (IBM Redbook)
[9] Mark O'Neill - Web Services Security