KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

POLITECHNIKA BIAŁOSTOCKA

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Rozproszone systemy internetowe

Bezpieczeństwo usług WWW

„Podniesienie potencjału uczelni wyższych jako czynnik rozwoju gospodarki opartej na wiedzy”

Nr projektu: UDA-POKL.04.01.01-00-143/09-00

Usługi bezpieczeństwa w sieci

- Poufność (confidentiality)
- Weryfikacja i autoryzacja (authentication/ authorization)
- Integralność (integrity)
- Niezaprzeczalność / Gwarancja oryginalności (non-repudiation / authenticity)
- Stemple czasowe (być może poza kanonem, ale implementowane)

Wyzwania i realizacje dla sieci WWW

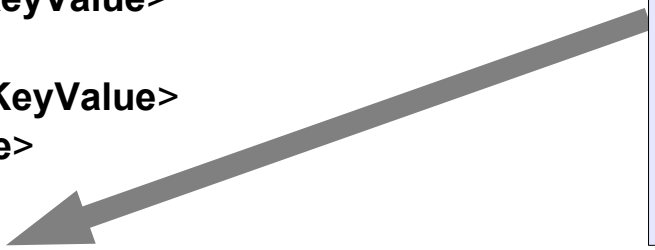
- Transport Layer Security
- „point-to-point” vs „end-to-end” w wydaniu sieci WWW
- Podpisywanie i szyfrowanie części dokumentu XML
- Postać kanoniczna dokumentu XML (canonicalization – c14n)
- Architektury zarządzania tożsamością

Proponowane standardy

- XML Digital Signature
- XML Encryption
- XML Key Management Specification
- Extensible Access Control Markup Language
- Secure Assertion Markup Language
- WS-Security
- WS-SecurityPolicy, WS-SecureConversation, WS-Trust

XML Digital Signature

```
<Signature>
  <SignedInfo>
   <CanonicalizationMethod Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
    <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"/>
    <Reference URI="http://www.w3.org/TR/2000/REC-xhtml1-20000126/">
       <Transforms>
            <Transform Algorithm="http://www.w3.org/2006/12/xml-c14n11"/>
       </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <DigestValue> ... </DigestValue>
    </Reference>
  </SignedInfo>
  <SignatureValue>...</SignatureValue>
  <KeyInfo>
    <KeyValue>
        <DSAKeyValue>
            ...
        </DSAKeyValue>
    </KeyValue>
   </KeyInfo>
   <Object />
</Signature>
```



Różnica pomiędzy **enveloped** a **enveloping**

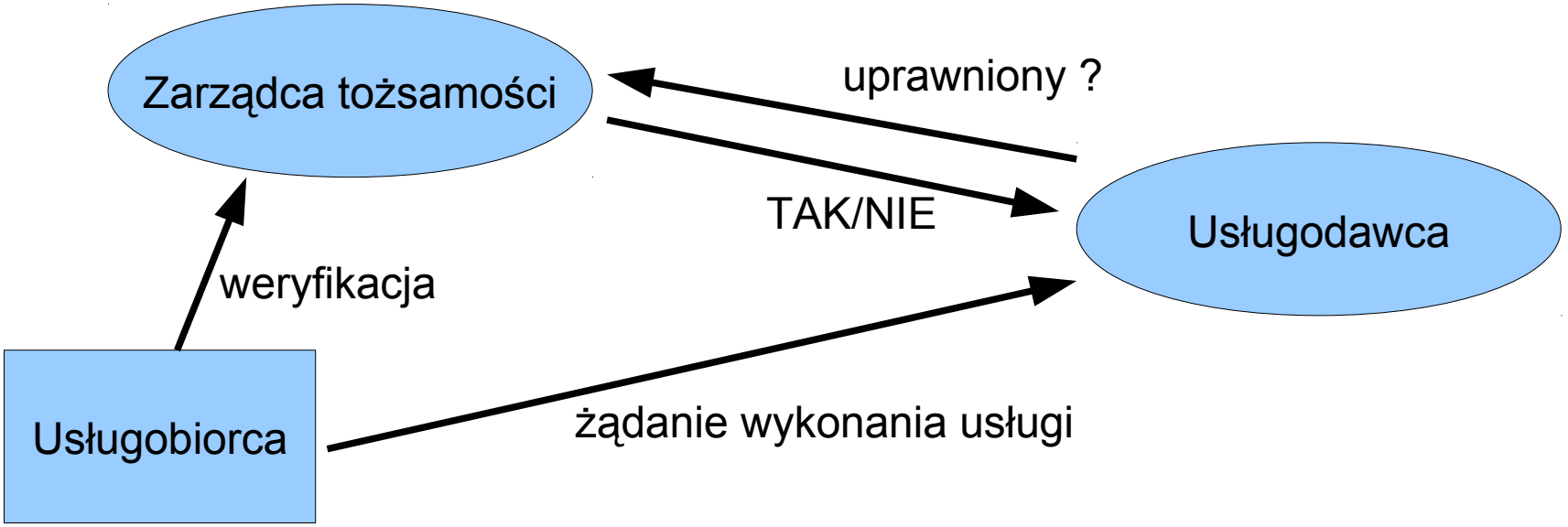
XML Encryption

```
<EncryptedData >
  <EncryptionMethod   Algorithm="..."> ...  </EncryptionMethod>
  <ds:KeyInfo>
    <ds:KeyName>Ala</ds:KeyName>
    <ds:RetrievalMethod URI="http://exmample.org/Reagle/PublicKey"
          Type="http://www.w3.org/2001/04/xmlenc#EncryptedKey"/>
    <EncryptedKey>
      <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc"/>
      <CipherData>
        <CipherValue>qZk+NkcGgWq6PiVxeFDCbJzQ2J0=</CipherValue>
      </CipherData>
    </EncryptedKey>
  </ds:KeyInfo>
  <CipherData>
    <CipherReference URI="http://example.org/pgpkeys/reagle.b64">
      <Transforms>
        <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#base64"/>
      </Transforms>
    </CipherReference>
  </CipherData>
</EncryptedData>
```

Szyfrowanie elementu i zawartości

XACML i SAML

- XACML – język opisu kontroli dostępu
  - Subject, Resource, Action, Obligation
- SAML – język do wymiany danych weryfikacyjnych i autoryzacyjnych pomiędzy domenami (assertions)



Przykład: elektroniczny urząd jako zestaw usług

- Usługa realizująca przeglądanie, dodawanie i usuwanie dokumentów
- Narzędzia: JAX-WS, Apache CXF
- Zabezpieczenia
  - Stemple czasowe dla każdej operacji
  - Weryfikacja usługobiorcy
  - Podpis elektroniczny
  - Szyfrowanie danych

Przykład: stemple czasowe

```
EndpointImpl el = (EndpointImpl)
EndpointImpl.publish("http://localhost:6060/urzad", u);
Endpoint ep = el.getServer().getEndpoint();
HashMap<String,Object> inProps = new
                HashMap<String,Object
inProps.put("action","Timestamp");
WSS4JInInterceptor wIn = new
        WSS4JInInterceptor(inProps)
ep.getInInterceptors().add(wIn);
HashMap<String,Object> outProps = ne
                HashMap<String,Object>();
outProps.put("action","Timestamp");
WSS4JOutInterceptor wOut = new
        WSS4JOutInterceptor(outProps);
ep.getOutInterceptors().add(wOut);
```

Operacja jaką należy wykonać podczas przetwarzania komunikatu

Przechwytywanie Komunikatów wchodzących i wychodzących

Przykład: weryfikacja użytkownika

Wybór metody weryfikacji

```
outProps.put("action","UsernameToken");
```

Sposób kodowania hasła w nagłówku SOAP

```
outProps.put("passwordType","PasswordDigest");
```

Klient: identyfikator użytkownika

```
outProps.put(WSHandlerConstants.USER,"...");
```

Klasa obsługująca ustawianie/sprawdzanie hasła

```
outProps.put(WSHandlerConstants.PW_CALLBACK_CLASS
,"...");
```

Przykład: podpisywanie komunikatu

```
outProps.put("action","Signature");
outProps.put(WSHandlerConstants.SIGNATURE_USER,"...");
outProps.put(WSHandlerConstants.SIG_PROP_FILE,"key.props");
```

org.apache.ws.security.crypto.provider=
org.apache.ws.security.components.crypto.Merlin
org.apache.ws.security.crypto.merlin.keystore.type=jks
org.apache.ws.security.crypto.merlin.keystore.password=...
org.apache.ws.security.crypto.merlin.keystore.alias=...
org.apache.ws.security.crypto.merlin.file=...

Zagadnienia do samodzielnego zbadania/przemyślenia

- Rozwiń przykład z wykładu dodając do niego operację szyfrowania (tylko dane zawarte w elemencie **<body>** komunikatu SOAP)
- Obejrzyj moduł *rampart* w implementacji Axis2 i porównaj go z propozycjami Apache CXF
- Zapoznaj się z projektami Liberty Alliance i Kantara Initiative