

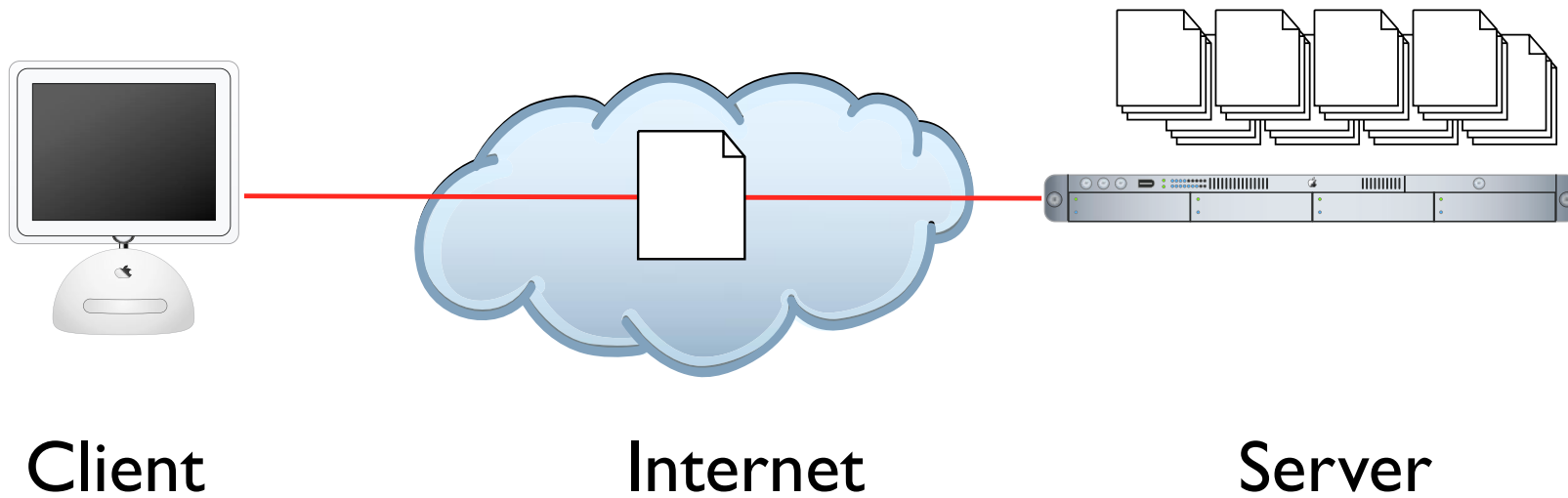
XML-Signature

© 2005, Markus Weissmann
<mail@mweissmann.de>

Table of Contents

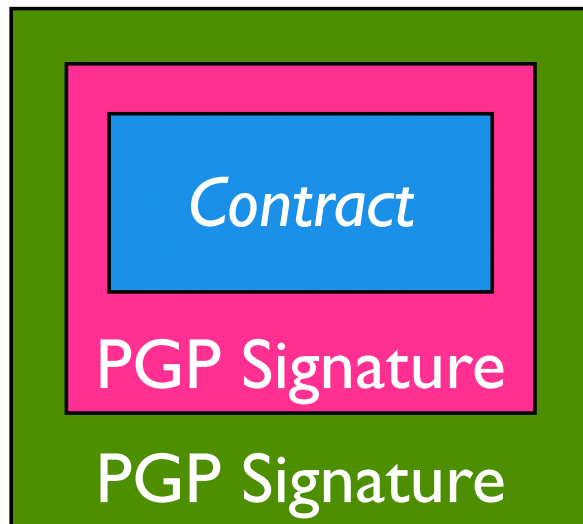
- Motivation
- Overview
- Signature Composition
- Demo
- Implementations
- Conclusions

Motivation

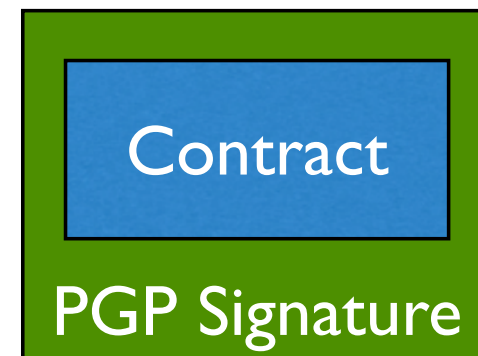


Motivation (cont'd)

Sign new Contract



Sign separate Contract



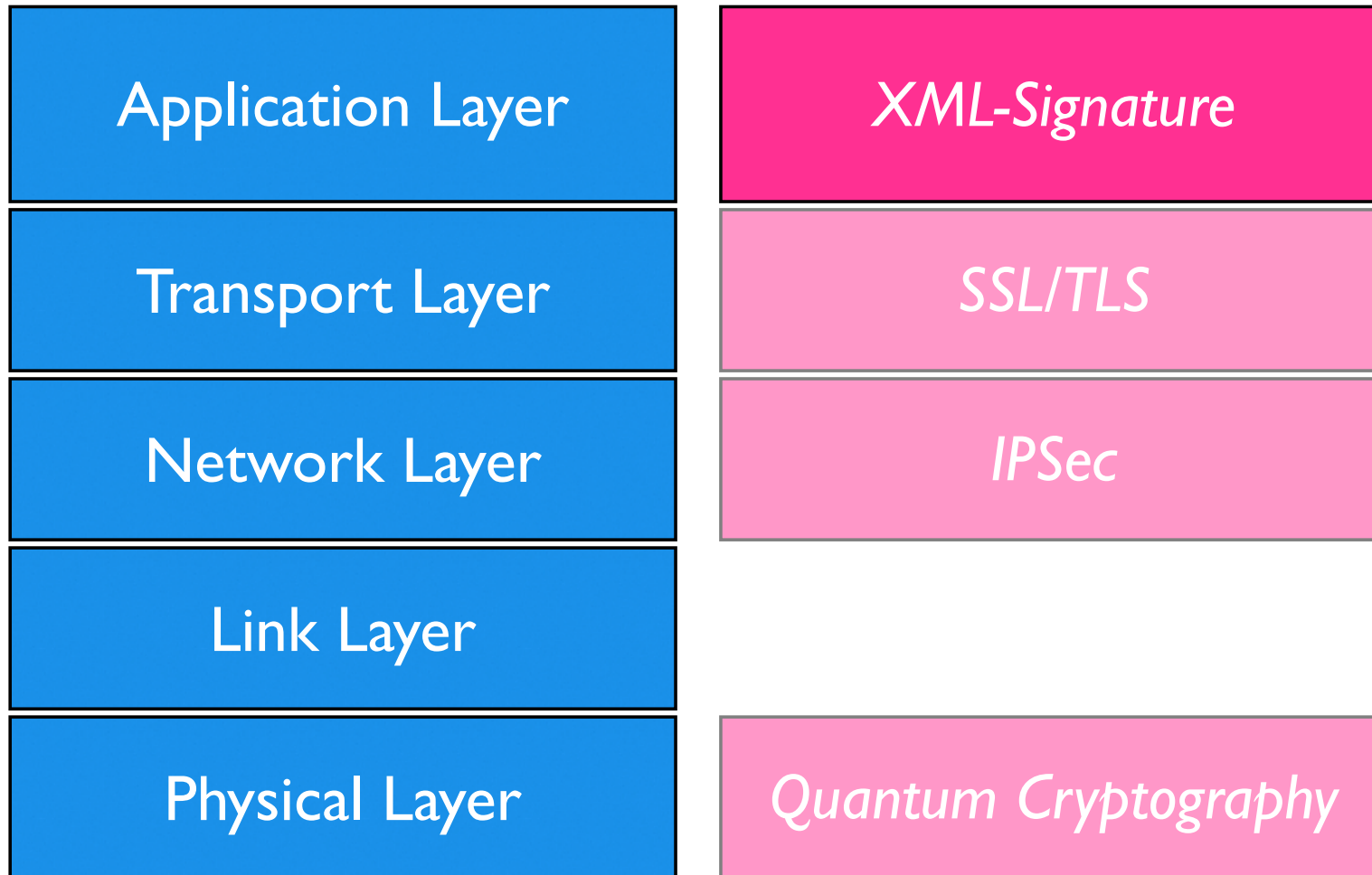
Motivation (cont'd)

- Signature gets lost after leaving SSL connection (auth)
- No standardized way to sign same document more than once

XML-Signature Standard

- W3C Recommendation, 12. Feb 2002
<http://www.w3.org/TR/2002/REC-xmlsig-core-20020212/>
- IETF RFC 3275
<http://www.ietf.org/rfc/rfc3275.txt>

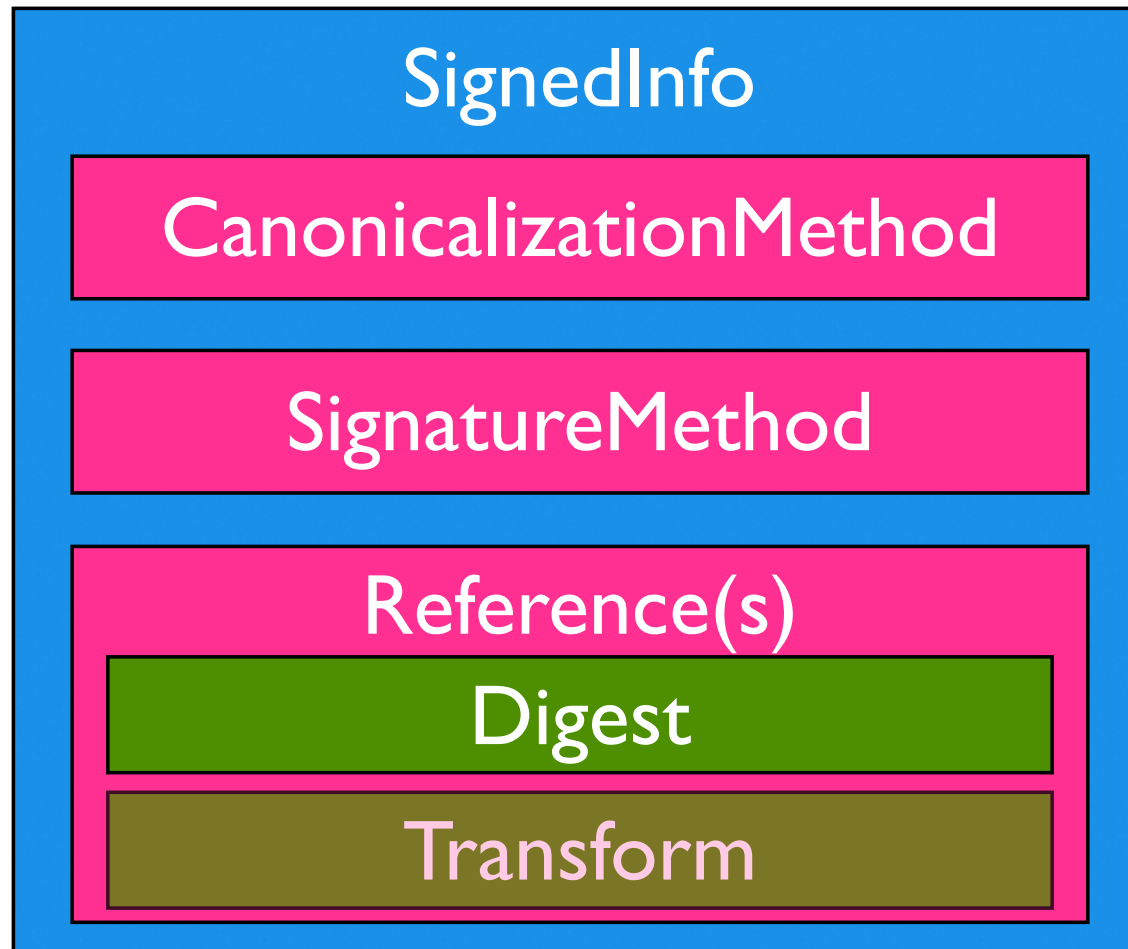
Internet Layer



Signature Composition



SignedInfo Composition



Canonicalization

Equivalency of XML Documents:

`<rsakey p="7" q='11' />`

`==`

`<rsakey q="11" p="7"></rsakey>`

Canonical XML 1.0

<http://www.w3.org/TR/2001/REC-xml-c14n-20010315>

SignatureMethod

- DSA with SHA1
- RSA with SHA1
- HMAC with SHA1

Reference

- Reference to
 - URI (detached signature)
 - Id (enveloped signature)

Reference: Digest

- Base64 encoded SHA1 Hash value

```
<DigestMethod Algorithm="http://www.w3.org/2000/09/xmlsig#sha1" />  
<DigestValue>  
5qJ4lk4gMOEcR5vW9s3v9InV8Ps=  
</DigestValue>
```

Transform: XPath

Query Language for XML

/	root element
//	any element
*	any element in .
foo[5]	5th child of foo
bar[@f=1]	element bar with attribute f==1

Transform: XSLT

eXtensible Stylesheet Language Transformation

- Includes XPath
- Functional Programming Language
- Turing complete 
<http://www.unidex.com/turing/utm.htm>

Signature Composition



SignatureValue

- Base64 encoded signature
- Value dependent on chosen Method

```
<SignatureValue>  
  12345===  
</SignatureValue>
```

KeyInfo

- Information about key used (e. g. public key, filename)
- Dependent on chosen Method
- Highly complicated structure

```
<KeyInfo>  
  <KeyName>  
    /home/mww/my-X509.pem  
  </KeyName>  
</KeyInfo>
```

Object

- Additional Information to be signed
- May be sole information to sign (enveloping signature)

```
<Object>...  
  <timestamp>  
    <date>2005|904</date>  
    <time>|5:45:00:00</time>  
  </timestamp>...  
</Object>
```

DEMO

Available Implementations

- Java: Web Services Developer Pack
javax.xml.crypto.dsig
- C: XMLSec - XML Security Library
<xmlsec/xmlsec.h>
- Python: PyXMLSec
libxml2, xmlsec
- Mono/.NET:
System.Security.Cryptography.Xml

PyXMLSec Verify

```
doc = libxml2.parseFile(xml_file)

# Find start node
node = xmlsec.findNode(doc.getRootElement(), xmlsec.NodeSignature,
xmlsec.DSigNs)

# Create signature context
dsig_ctx = xmlsec.DSigCtx()

# Load key
dsig_ctx.signKey = xmlsec.cryptoAppKeyLoad(key_file,
xmlsec.KeyDataFormatPem, None, None, None)

# Verify signature
dsig_ctx.verify(node)

if dsig_ctx.status == xmlsec.DSigStatusSucceeded:
    print "Signature is valid"
```

Conclusion

Problems not solved by XML-Signature

- verify XML validity
- verify Schema
- Key management
- transparent authentication on application layer

Conclusion (cont'd)

Problems solved by XML-Signature

- Standardized way to sign content of URIs
- Standard to sign XML documents
- Authentication standard on application layer
- Extensible signature framework

Conclusion (cont'd)

