Network Architectures and Services, Georg Carle
Faculty of Informatics
Technische Universität München, Germany

Network Security

Chapter 10

Application Layer Security: Web Services (Part 2)

Part I: Introduction to Web Services

- Part I: Introduction to XML and Web Services
- Part II: Securing Web Services
- Part III: Identity Federation

Towards a Web Services “Stack”

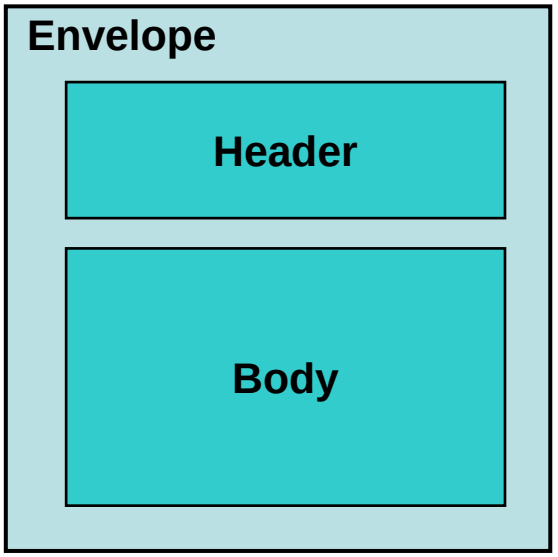
Web Services Protocols	Message Exchange between Web Services; SAML; …
SOAP	„Transport Layer“ for Web Services
HTTP	Application Layer
Transport Layer	End-to-end connectivity between processes (port concept)
Network Layer	Routing between networks
Data Link Layer	Interface to physical media
Physical Layer	

Use Cases For XML DSig/XML Encryption

- You know XML Digital Signature and XML Encrytion now
- These standards **form the foundation of many Web Service security protocols**:
  - SOAP
  - WS-Security
  - SAML
  - WS-Federation
  - ID-FF (Identity Federation by Liberty Alliance)
  - …

A Closer Look At SOAP

- **Defines how to send structured XML over a network**
  - Follows paradigm of state-less, one-way messages
  - But applications can create complex communication patterns from this by supplying additional application-specific information
  - Thus, SOAP is agnostic to what it conveys
  - Used as a foundation layer for Web Service protocols
- **Simple message format:**




```
<soap:Envelope xmlns:soap="http://...">

     <soap:Header>
         <app-specific:requestor id="…" />
     </soap:Header>


     <soap:Body>
         <app-specific:request item="…" />
     </soap:Body>



</soap:Envelope>
```

SOAP

- SOAP defines **bindings**: important specifications how to use SOAP with underlying protocols
  - HTTP + (SSL +) TCP
  - SMTP

- Some criticism:
  - May lead to abuse of HTTP semantics
  - Firewalls are often configured to accept HTTP → must now inspect XML content → increases attack surface
  - However, HTTP is a core element in Web Services anyway

- **Information in SOAP can be cryptographically secured with XML Signature and Encryption**
- However, there are many ways to get this wrong!
  → self-designed crypto protocols are often flawed

Security Issues To Think About

- Web Services are a **valuable target for attackers**
  → business-relevant data = money

- We have seen that **XML Signature and XML Encryption can provide security, but at the price of high complexity**
- **Designing a crypto protocol** and protocol handlers must thus be done with **extra great care here**
  - Simple example: ***first*** verify that the signature is from a known key, ***then*** do the signature check
  - Otherwise, you leave yourself open to complexity or DoS attacks

- Some further attacks to think of:
  - SQL injection
  - XPath and XQuery injection
  - Complexity and DoS attacks on parsers
  - More are listed on owasp.org

Example of Parser DoS: Entity Expansion

- The following may expand to 2 GB when parsed
  (note: we did not try it; it probably depends on the parser)

```
<!DOCTYPE foo [
<!ENTITY a "1234567890" >
<!ENTITY b "&a;&a;&a;&a;&a;&a;&a;&a;" >
<!ENTITY c "&b;&b;&b;&b;&b;&b;&b;&b;" >
<!ENTITY d "&c;&c;&c;&c;&c;&c;&c;&c;" >
<!ENTITY e "&d;&d;&d;&d;&d;&d;&d;&d;" >
<!ENTITY f "&e;&e;&e;&e;&e;&e;&e;&e;" >
<!ENTITY g "&f;&f;&f;&f;&f;&f;&f;&f;" >
<!ENTITY h "&g;&g;&g;&g;&g;&g;&g;&g;" >
<!ENTITY i "&h;&h;&h;&h;&h;&h;&h;&h;" >
<!ENTITY j "&i;&i;&i;&i;&i;&i;&i;&i;" >
<!ENTITY k "&j;&j;&j;&j;&j;&j;&j;&j;" >
<!ENTITY l "&k;&k;&k;&k;&k;&k;&k;&k;" >
<!ENTITY m "&l;&l;&l;&l;&l;&l;&l;&l;" >
]>
<foo> fooo &m; bar </foo>
```

Source: [iSec2010]

Securing SOAP: WS-Security

- Framework that **defines how XML Signature and XML Encryption can be employed safely for SOAP and XML-based application protocols**
- **WS-Security does not define new mechanisms**
  → "standardizing the standards"
- Some WS-Security Features:
  - **Signatures** with XML Signature (sane methods)
  - **Encryption** with XML Encryption (sane methods)
  - **Transports Security Tokens**:
    - X.509 certificates
    - Kerberos Tokens
    - SAML Tokens (more about SAML shortly)
    - Passwords
    - Password digests
  - **Timestamps**
- Also describes alternatives for use cases where only host-to-host security is required: simpler, uses SSL/TLS

WS-I BSP

- **WS-Interoperability Basic Security Profile**
  - A standard by the Web Services Interoperability Organization
  - Defines comprehensively how to use the mechanisms in Web Services security ***safely***
- Intent is **clarification** → improve ease of use
- **Some remarkable points:**
  - Prohibits the use of some protocols with flaws, like older SSL versions (SSL 2.0 disallowed!)
  - Defines ciphersuites to use
  - Restrictions on SOAP envelope, header and processing
  - Enveloping XML Signature disallowed, enveloped signature discouraged → emphasis on detached signature!
  - Rules for transforms
  - Rules to facilitate encryption processing

Security Assertion Markup Language (SAML)

- **Motivation for SAML**:
  - Web Services may cross organisational boundaries
    → need for authentication and authorization for access control
    → convey “security attributes” between organisations
  - Portable (shared) “identities” with attributes between organisations
- **SAML works with assertions.** We speak of:
  - Subject: an entity that is asserting its identity
  - Assertion: a claim about a subject that must be proved
- **SAML can be used to exchange assertions between organisations**
- **SAML consists of three parts:**
  - Assertions
  - Protocol: XML schema and request/response protocol
  - Bindings: e. g. to SOAP/HTTP
- So-called **SAML Profiles** specify **use patterns** for SAML, i.e. how assertions are embedded, extracted and processed
  - E. g. a profile for use with Web Browsers

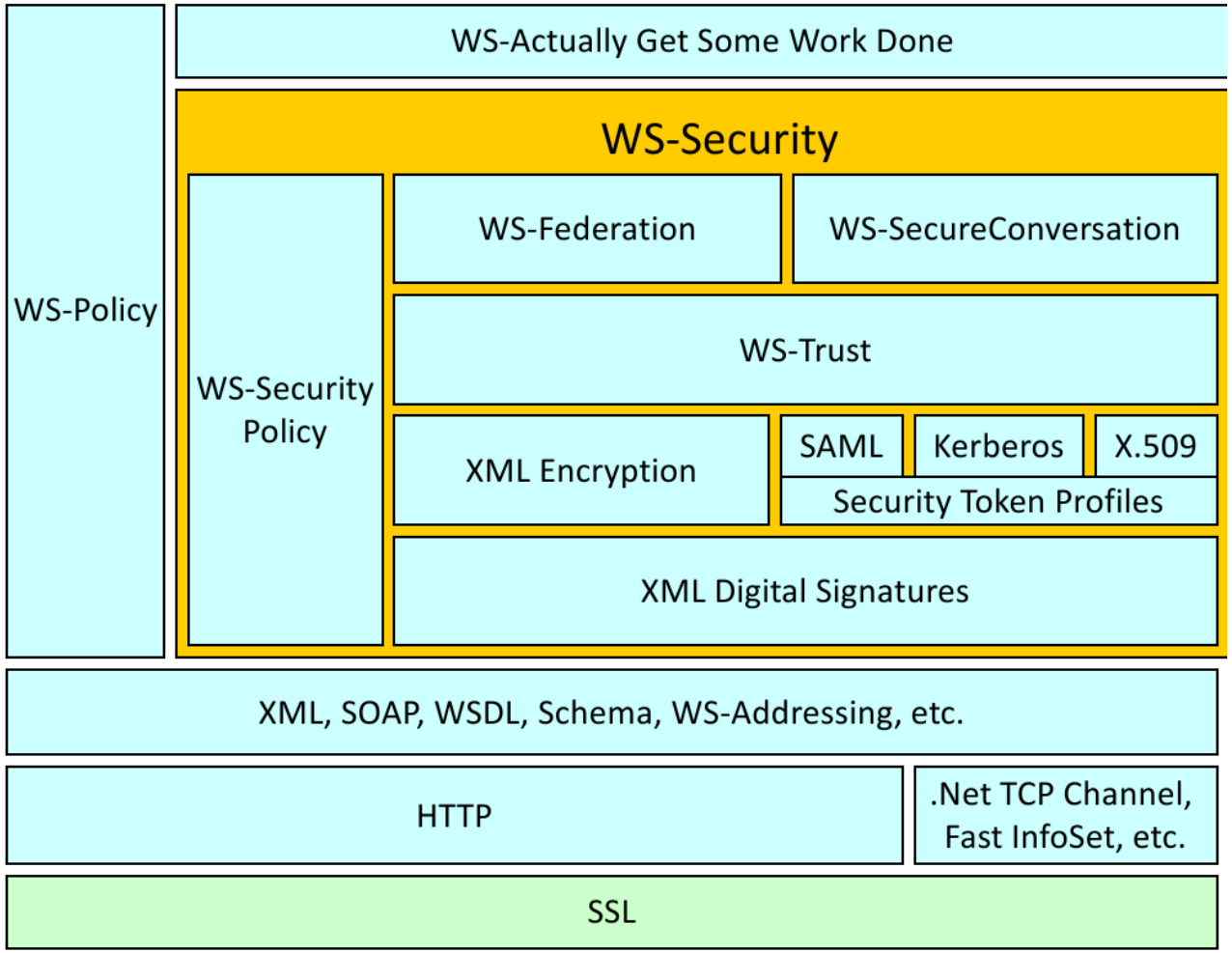
SAML Assertions

- **Three types of assertions**:
  - **Authentication**: states that an authority has authenticated the subject of the assertion
  - **Authorization**: states that an authority has granted or denied access to the subject of the assertion
  - **Attributes**: qualifying information about an authentication or authorization
- Some elements that are **common to all assertions**:
  - Issuer
  - Timestamp
  - Subject
  - Conditions on assertion (e. g. “not valid after…”)
  - Intended audience
  - Signatures

Example: SAML Authentication Response

```
<samlp:Response xmlns:samlp="urn:..." InResponseTo="..." Version="2.0" IssueInstant="2007-12-
      10T11:39:48Z" Destination="...">
    <saml:Issuer>the-issuer</saml:Issuer>
    <samlp:Status xmlns:samlp="...">
        <samlp:StatusCode xmlns:samlp="..." Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
    </samlp:Status>
    <saml:Assertion xmlns:saml="urn:..." Version="2.0" ID="..." IssueInstant="2007-12-
       10T11:39:48Z">
        <saml:Issuer>the-issuer</saml:Issuer>
        <Signature xmlns="...">
          ...
        </Signature>
        <saml:Subject>
            <saml:NameID>...</saml:NameID>
            <saml:SubjectConfirmation Method="...">
                <saml:SubjectConfirmationData>...</saml:SubjectConfirmationData>
            </saml:SubjectConfirmation>
        </saml:Subject>
        <saml:Conditions NotBefore="2007-12-10T11:29:48Z" NotOnOrAfter="2007-12-10T19:39:48Z">
          ... e. g. audience restrictions
        </saml:Conditions>
        <saml:AuthnStatement AuthnInstant="2007-12-10T11:39:48Z" SessionIndex="...">
            <saml:AuthnContext>
                <saml:AuthnContextClassRef>urn:...Password</saml:AuthnContextClassRef>
            </saml:AuthnContext>
        </saml:AuthnStatement>
        <saml:AttributeStatement>
            <saml:Attribute Name="givenName">
                <saml:AttributeValue xmlns:saml="...">...</saml:AttributeValue>
            </saml:Attribute>
              ... more attributes ...
        </saml:AttributeStatement>
    </saml:Assertion>
</samlp:Response>
```

Bringing It All Together



Source: [iSec2010]

Recap: Security Guidelines for Web Services

- **Recommendations in several standards**
  - WS-Security
  - WS-I Basic Security Profile
  - Following these recommendations is strongly encouraged
- **Decrease attack surface**:
  - Always use SSL/TLS for host-to-host communication
  - Complexity is (one) enemy of security
  - Where you can, reduce the complexity of your protocol
- **Do not create/use protocols that you do not actually need**
  - Even SAML Profiles have been found to have weaknesses
- **Do not forget attacks outside cryptography**:
  - DoS
  - Injection attacks
- **Conclusion**: Security for Web Services can be much work and should be addressed with great care.

Further Pointers

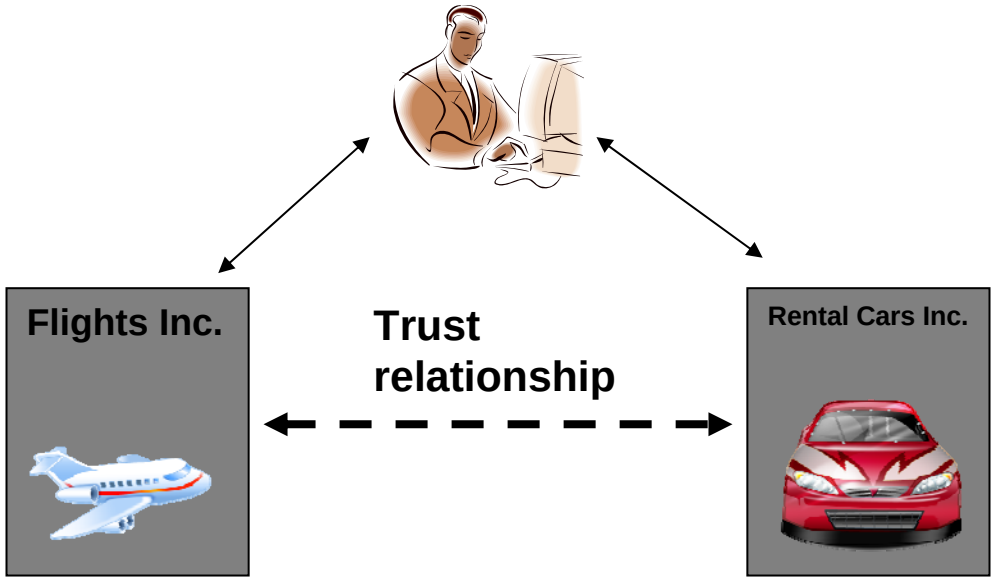
- There are more security-relevant standards, which we will not discuss further here
- Have a look yourself, if you want, at:
  - WS-SecureConversation
    → establishes security contexts, SSL-like pattern
  - WS-Reliability
    - Reliable communication for, e.g., transactions
  - WS-Trust
  - WS-Policy
  - WS-Interoperability

Part III: Identity Federation

- Part I: Introduction to XML and Web Services
- Part II: Securing Web Services
- Part III: Identity Federation

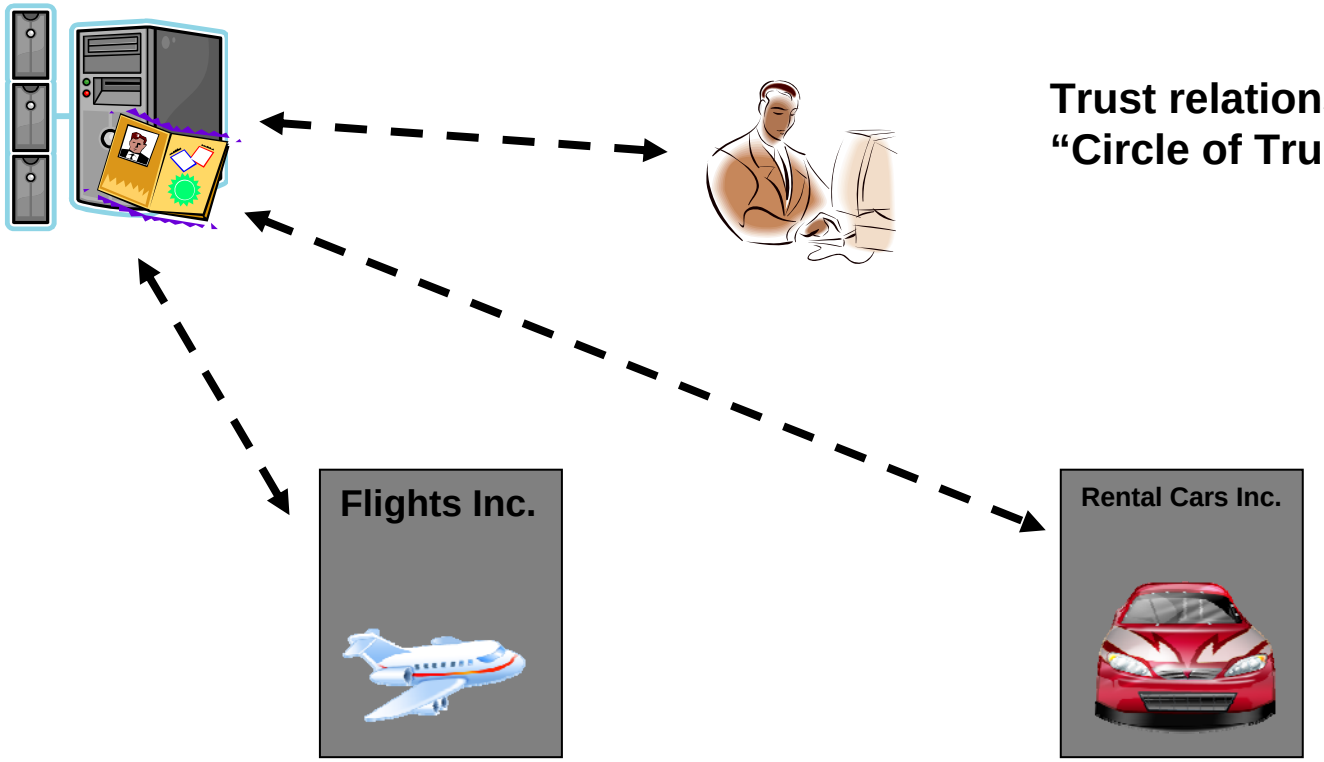
Identity Federation As Shared Authentication

- Entity Bob wishes to do business:
  - Bob wants to reserve a flight from Flights Inc.
  - Bob also wants to rent a car from Rental Cars Inc.
- On booking the flight, Bob consents to federate an identity
  - A pseudonym for use with Rental Cars Inc. is generated
  - Bob is redirected to Rental Cars Inc. with a security token that proves his membership with Flights Inc. (with the pseudonym!)
    Assertion: *"pseudo_bob is a member of domain Flights Inc."*
- Identity Federation: propagation of trust / authentication across organizational boundaries



Identity Provider

- Example may be extended by having a third party acting as the Identity Provider for Bob
- Bob authenticates with credential from Identity Provider



Identity Federation: Concepts

- **Concept is not new: sharing of Identities between organisations**
  - Portability of an identity
  - You know similar concepts, e. g. Kerberos
- **Use-cases**:
  - Allows users (or Web Services) to access services outside their own administrative domain
  - Most common example: Single Sign-On
- **Several standards implement Identity Federation,** also with Web Service technology, esp. SAML:
  - WS Federation (OASIS), part of the Web Services suite
  - ID-FF by Liberty Alliance: large consortium to establish open standards for Identity Federation
  - Shibboleth (Internet2)
  - OpenID: decentralized, more “community-oriented” and simpler standard

Identity Federation: Concepts

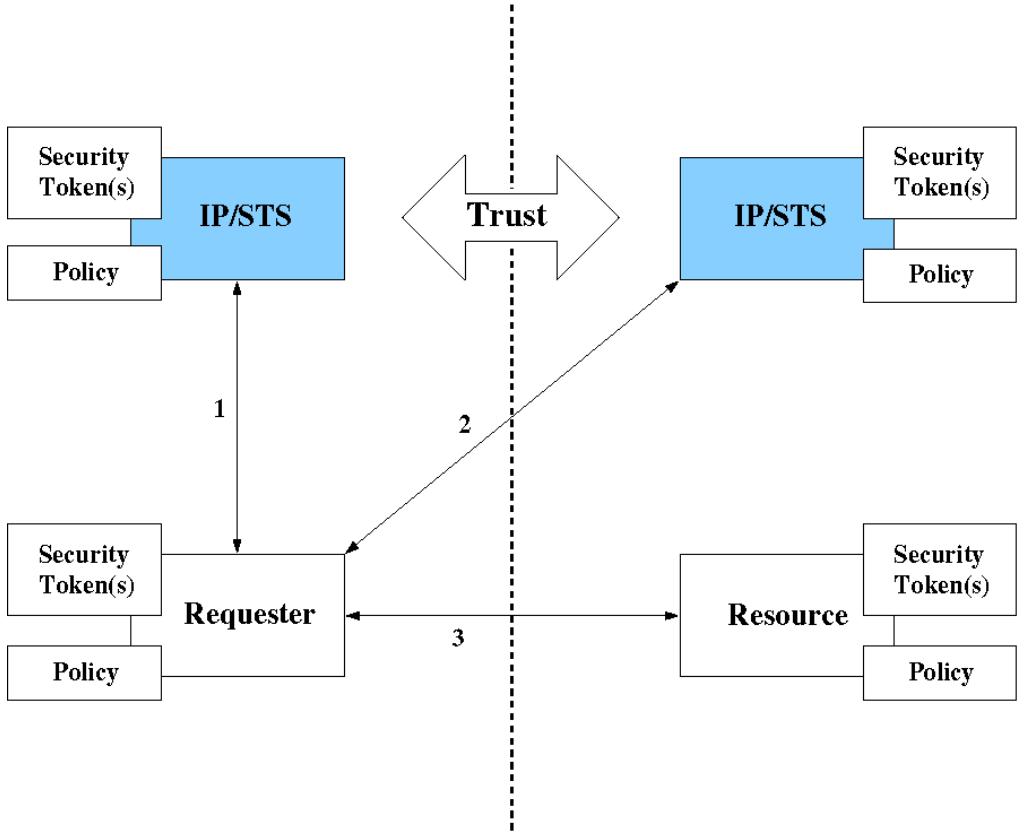
- The **basic schema** is always the same
  - An **entity has an Identity Provider (IdP) vouching for its identity**
  - In order to access a service, the entity **requests a credential from IdP**
    - May be explicitly for the service or generic
  - Entity **presents this credential to the Service Provider**

- Participants in an Identity Federation form a **“Circle of Trust”**
  - Within this circle of trust, an entity may use its federated identity to authenticate, access services etc.
  - Any organisation may act as an Identity Provider (if it is trusted by reyling participants)

- Nota bene: concepts like **Identity Management** that (may) build on Identity Federation **require much more than the pure security concepts** we present here
  - Validity between domains
  - Expiry
  - Secure administration
  - Roles & Access Control
  - Etc.

Identity Federation: Relationships 1

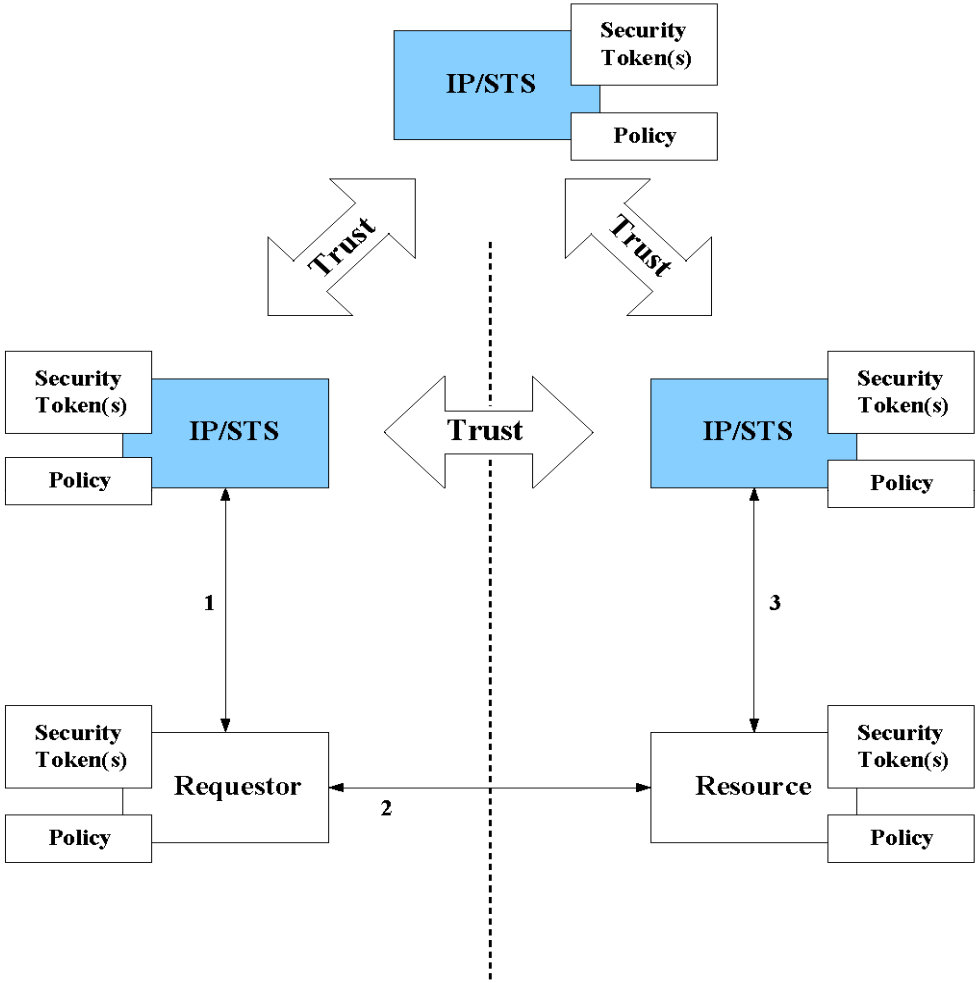
Note:
STS = Security Token Service



- **Simple model: direct trust between organisations**
  - Each organisation has an Identity Provider
  - Requester asks for a credential from his Identity Provider and presents it to the STS of the Service Provider he wishes to access
  - That STS may then grant access to the service
- Each participant may follow his own policies in this process

Identity Federation: Relationships 2

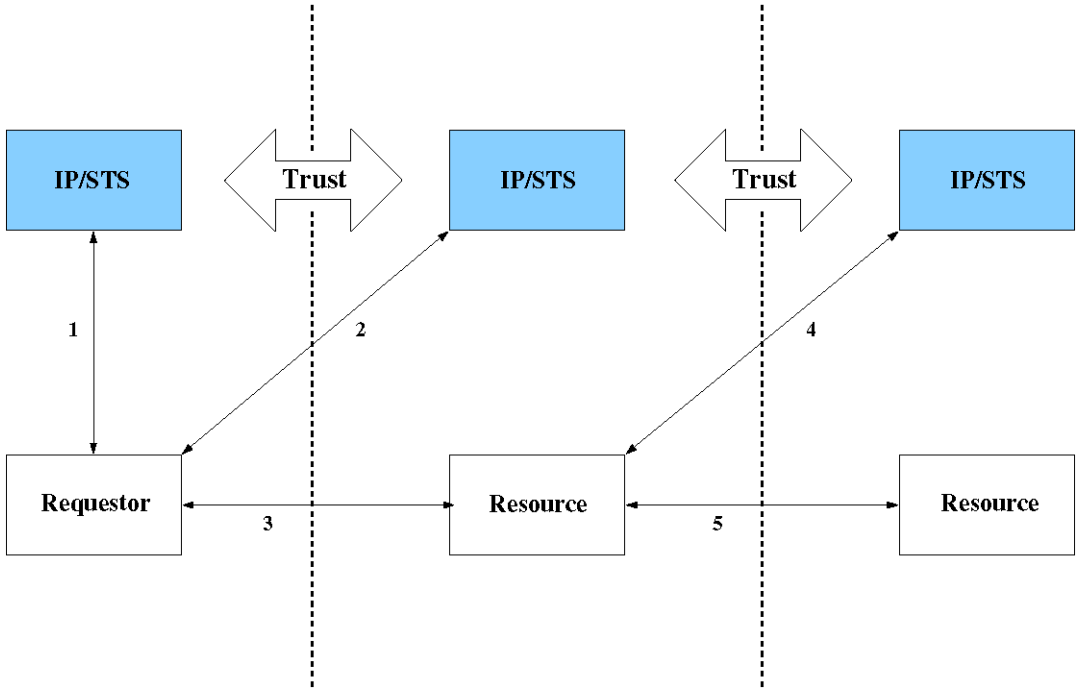
Note:
STS = Security Token Service



- **Extended model: trust between organisations is mediated by a Trusted Third Party**

Identity Federation: Relationships 3

Note:
STS = Security Token Service



- **Extended model with delegation**:
  - In order to fulfill a request, a resource accesses another (third-party) resource first
  - First resource acts “on behalf” of requestor

OpenID

- OpenID is a "**more decentralized"** system for Identity Federation
  - **No *a priori* trust** relationships envisaged → no Circles of Trust
  - Idea is that you **login with an identity you registered with an OpenID provider**
  - It is left to the Service Provider to decide whether to accept authentication with an unknown OpenID provider
- **Some features**:
  - **XML-based**
  - Supports **Discovery** mechanisms for OpenID providers
  - More **aimed at a Web scenario**: less comprehensive and generic in comparison with Web Services standards
  - Allows **delegation**: you can **host your own identity** and delegate each authentication process to your OpenID provider
  - OpenID is **well supported** on the Web

References

[XMLEnc] W3C. *XML Encryption*. http://www.w3.org/standards/techs/xmlenc.

[XMLDSig] W3C. *XML Signature*. http://www.w3.org/standards/techs/xmlsig

[SAML2010] OASIS. *OASIS Security Services (SAML) TC*. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[RoRe2004] J. Rosenberg, D. Remy. *Securing Web Services with WS-Security*. SAMS Publishing. 2004.

[OWASP] Open Web Application Security Project. 2010. http://www.owasp.org

[WSI] Web Services Interoperability Organization. *Basic Security Profile Version 1.0*. 2010. http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html

[OpenID] OpenID Foundation Web Site. http://openid.net/

[iSec2010] iSEC Partners. *Attacking XML Security*. http://www.isecpartners.com/files/iSEC_HILL_AttackingXMLSecurity_bh07.pdf