Security for Distributed Web-Applications via Aspect-Oriented Programming

Concepts of authentication, authorisation, and access control, and their implementation using aspect oriented programming in a service oriented architecture

Andreas U. Schmidt,
Nicolai Kuntze, Thomas Rauch
Fraunhofer-Institute for Secure Information Technology SIT
Darmstadt, Germany

ISSA 05, Sandton, South Africa
29th June – 1st July 2005

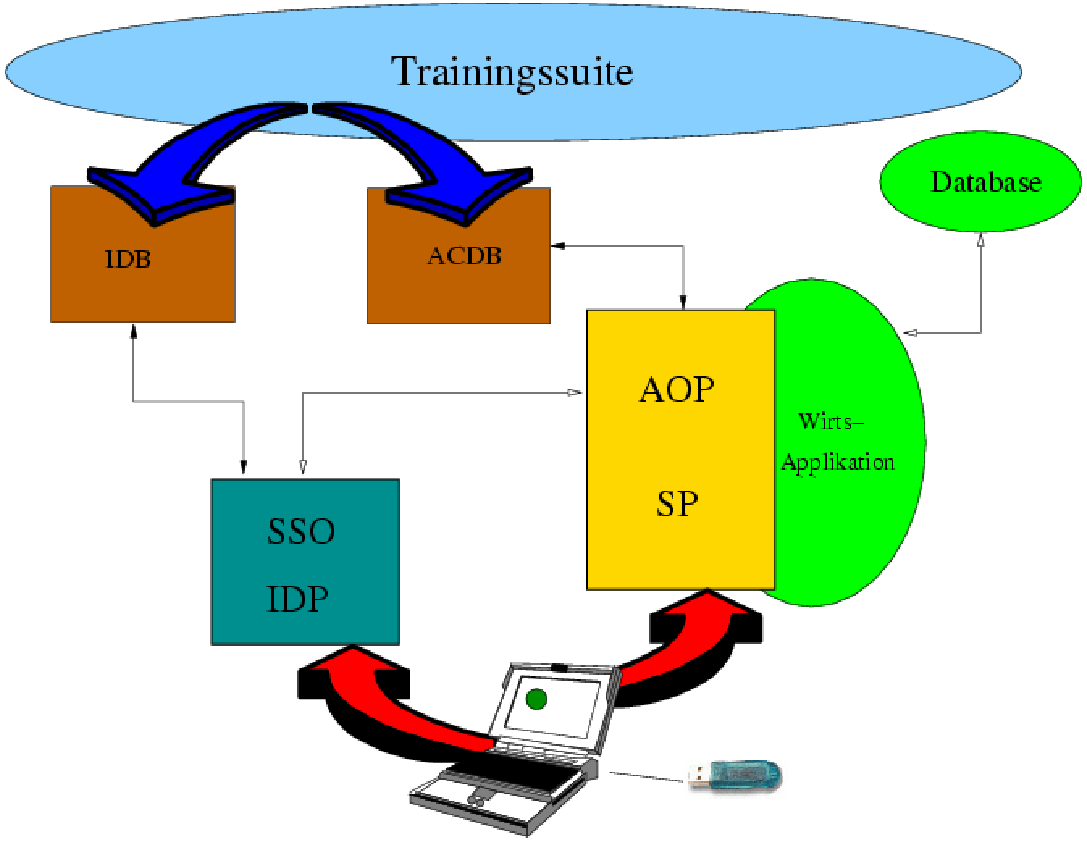
SIT
Fraunhofer Institut
Sichere Informations-
Technologie

issa
information security
south africa

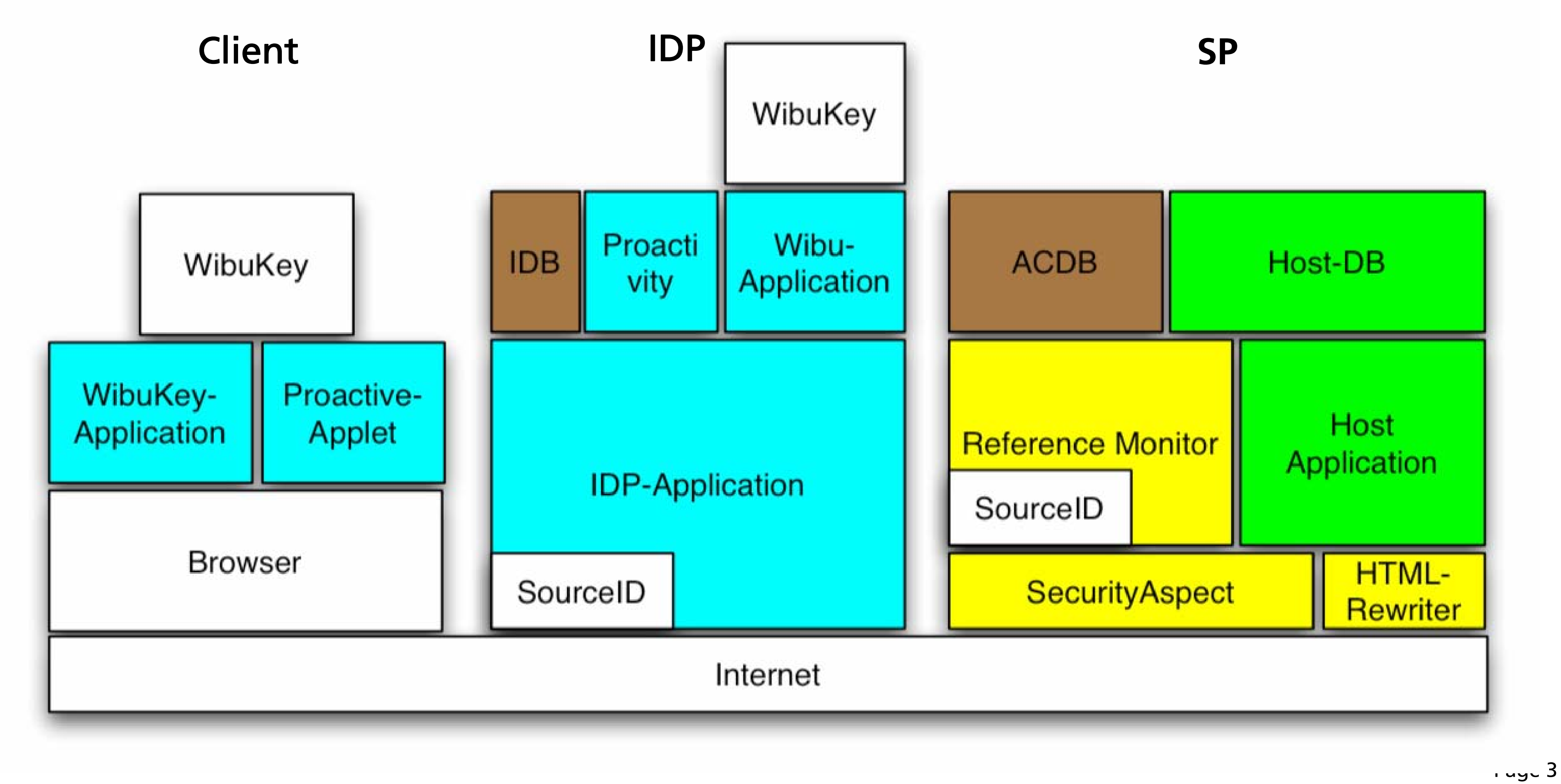
Concept and Coarse Architecture

Augment a Web-Application with AAA functionality

- ID-Management
- Role Based Access Control
- Workflows
- Implementation of a generic *security module*
- Subsequent addition to an *existing system*
- Intuitive deployment using a *training suite*



Components


Client
IDP
SP
WibuKey
WibuKey
IDB
Proacti vity
Wibu-Application
ACDB
Host-DB
WibuKey-Application
Proactive-Applet
IDP-Application
Reference Monitor
SourceID
Host Application
Browser
SourceID
SecurityAspect
HTML-Rewriter
Internet

Deployment process

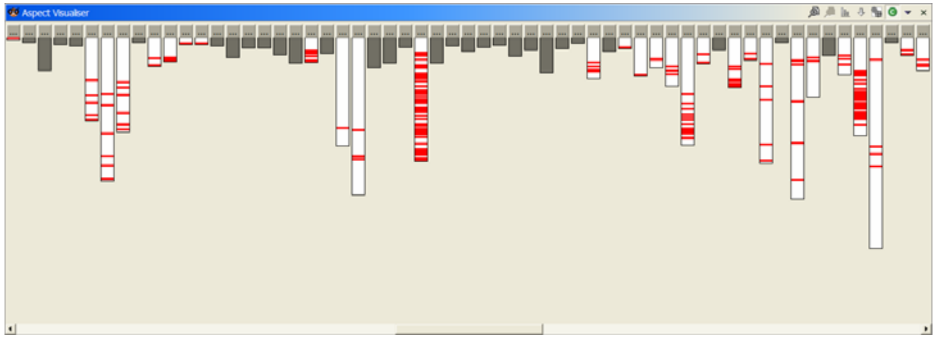
- **Training Phase**
    - adapts security module to host application
    - Training is done using a special training suite in a “learning-by-doing” approach
    - The administrator uses the application and records interaction
    - Then defines the security policy can be defined
- ➢ produces a XML file containing AAA rules for Workflows

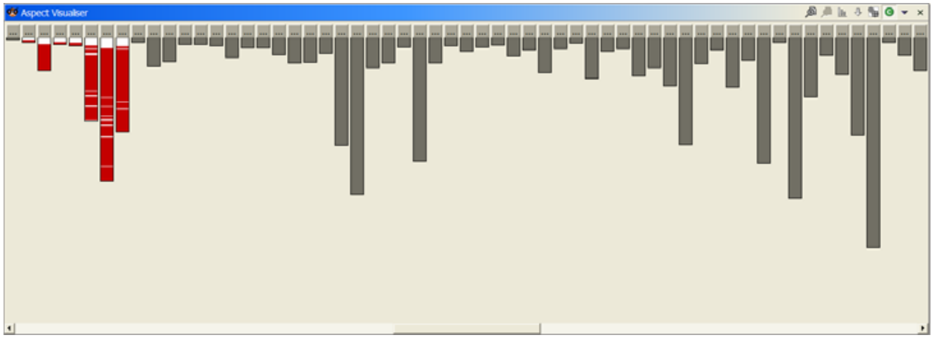
- **Production phase**
    - IDB and ACDB are fed
    - Security Aspect is ***weaved*** with host application
    - At run-time AAA decisions by the Reference Monitor are enforced via the Security Aspect

Aspect Oriented Programming - Fundamentals

- Orthogonal extension of the object oriented programming paradigm
- Isolation of *concerns* which are *scattered* in the source code in ***aspects***
- Adds functionality ***without changes in the source code***

  (but weaver declarations - pointcuts)
- ***Very different from wrapping***
- Logging in Tomcat: scattered across the packages and classes
- likewise: error handling, security, business rules, …
- class loading in Tomcat: concentrated in one package (9 classes)





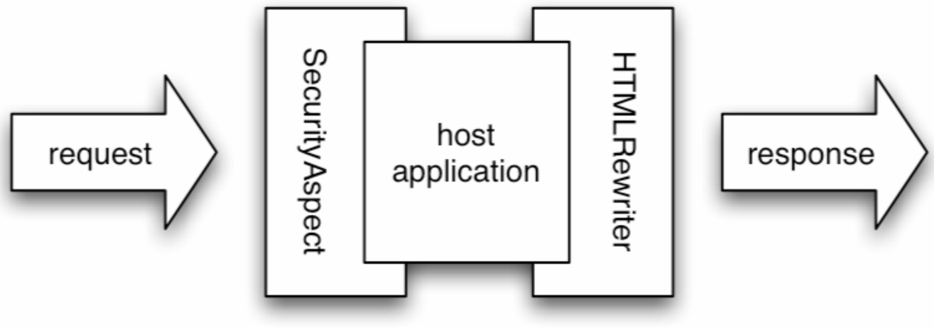
Aspect Oriented Programming - Practise

- Aspect Oriented Programming (AOP) is available in the form of programming frameworks for a wide range of languages such as
  - Java (AspectJ, AspectWorkz, ..)
  - C#
  - C++ (AspectC++)

- We are using **AspectJ** as it is one of the best developed implementations, ‘alive’, and well-documented

Aspect Oriented Programming - Usage

- The demonstrator isolates the security concerns by *pointcuts* to the interface methods in the host application's container and thus
- covers all incoming and outgoing flows of information



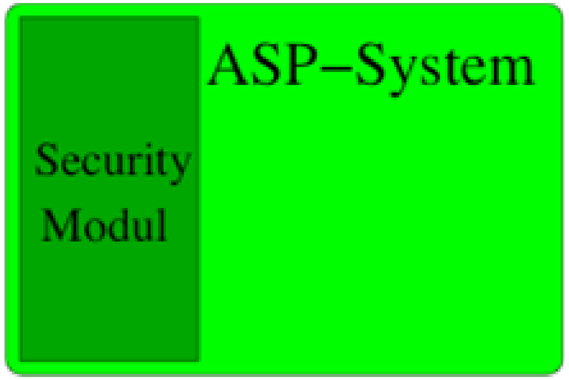
- Provides all information needed for AAA enforcement
- Enables changes in the produced HTML-pages of the host system

- Also possible: filtering of interaction of methods to prevent buffer overflows

Modularity and Separation of Duties

Security Module

- All security concerns are isolated in one module - not scattered through source code
- Security Module can be developed independently from system
- Configuration and adaptation by ACDB
- Security module is small, enabling specialised module tests
- This results in a highly reliable code

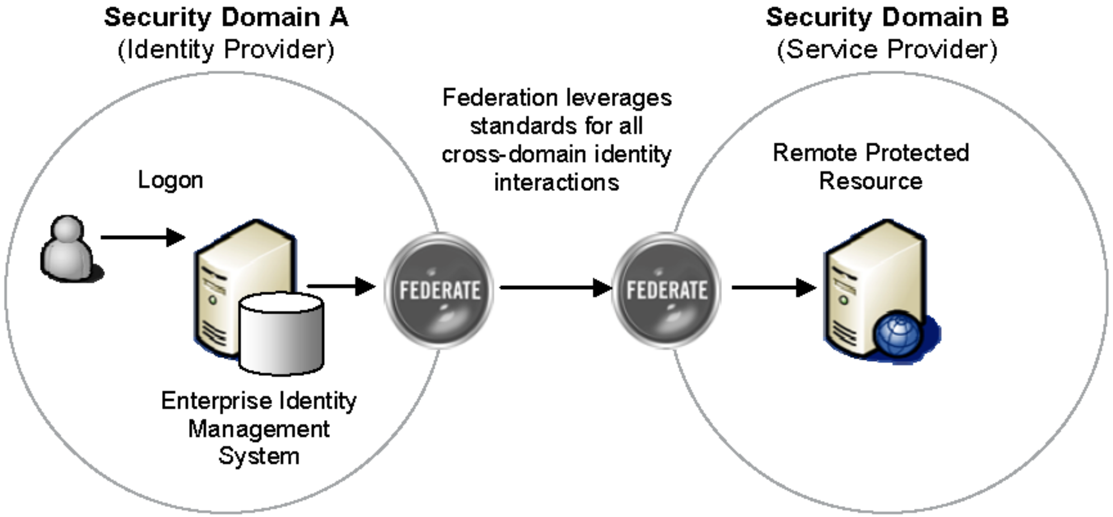


AAA logic

- Implemented by ***two*** distinct roles:
  - IDP for authentication
  - SP for authorisation, access control
- *SP* Can provide detailed ***audit*** information through data at user interaction level,
- enabling optimisation of the software usability –
- but bears privacy concerns: Separation of duties between IDP and SP and pseudonymity concepts are helpful

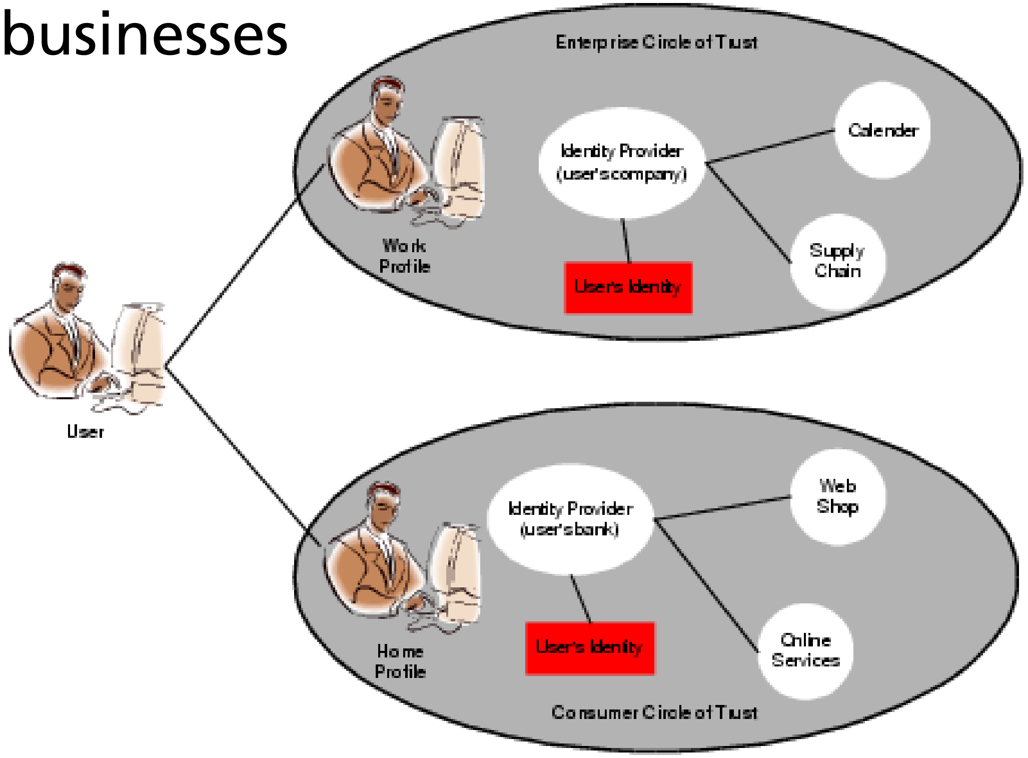
Authentication: Liberty Alliance Protocol

- Single Sign On (SSO)
- Integration of Smartcards etc.
- Proven to be scaleable
- Open for future collaborations
- SSO implemented in ID-Federation Framework (ID-FF)
- Open technology specifications
- Federated Identity
    - Identity Provider (IDP)
    - Service Provider (SP)
- ID-Web Service Framework (ID-WSF)
- Limited “Anonymity” through pseudonymity


Security Domain A
(Identity Provider)
Logon
Enterprise Identity Management System
FEDERATE
Federation leverages standards for all cross-domain identity interactions
FEDERATE
Security Domain B
(Service Provider)
Remote Protected Resource

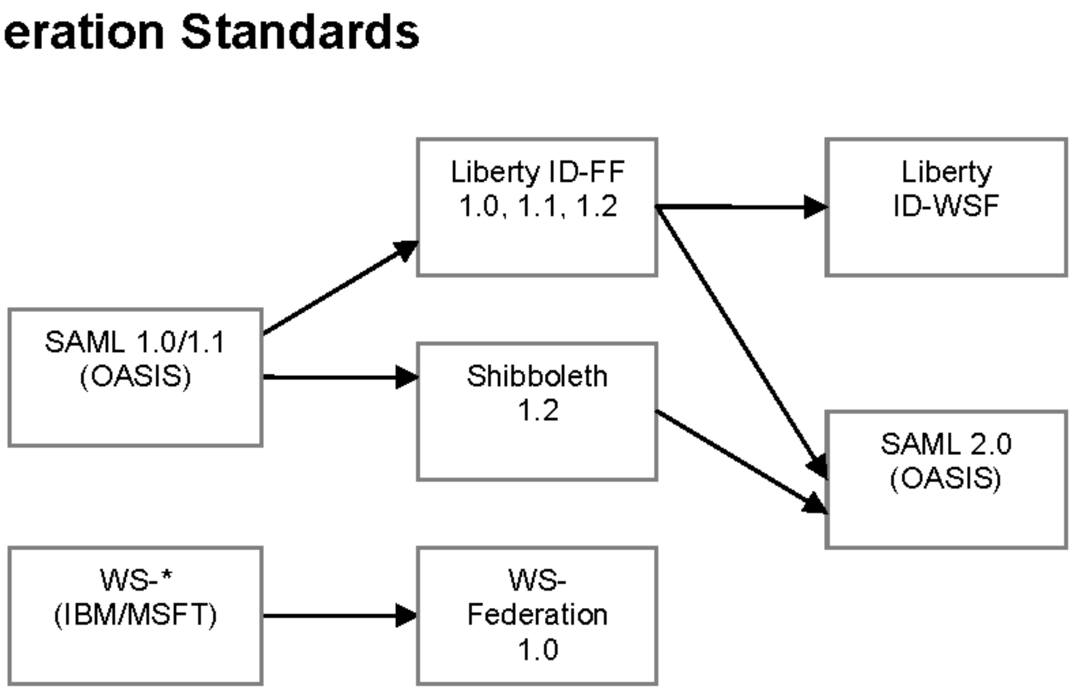
Circles of Trust

- Federation of service providers
- enable trust relationships between co-operating companies
- Requires operational agreements defining trust relationships between the businesses

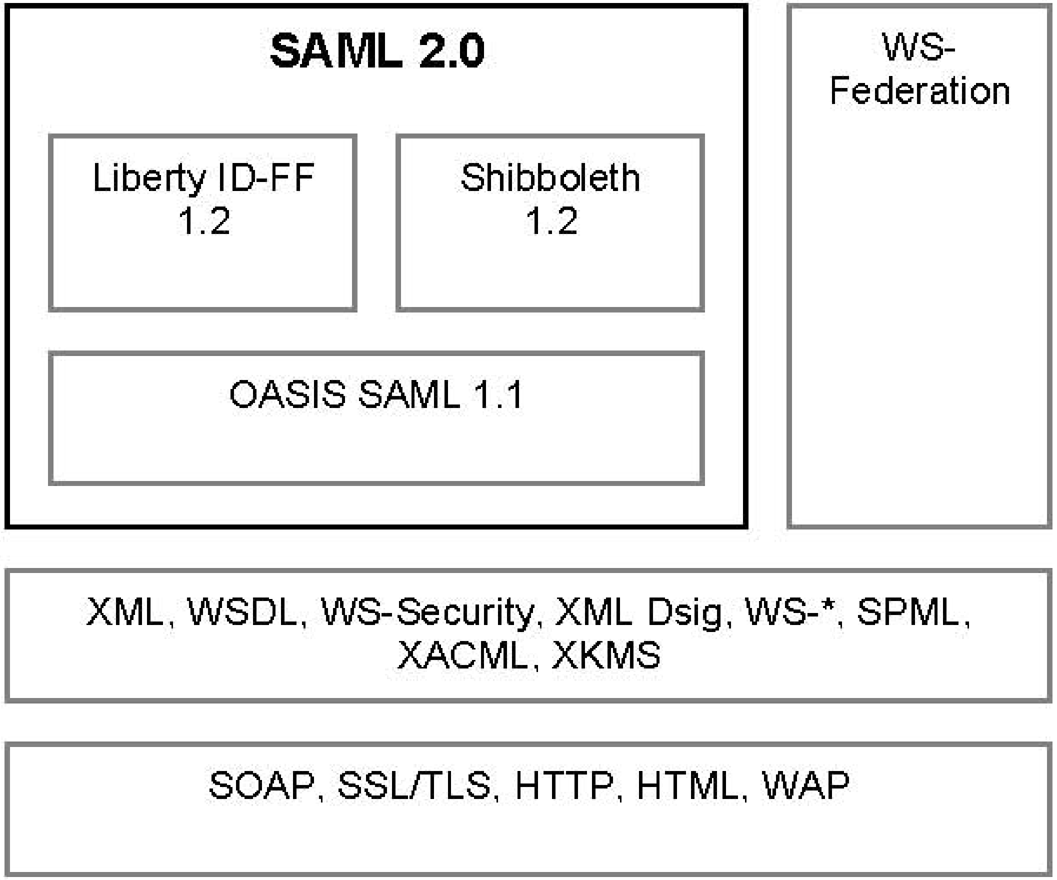


LAP History

- Recent developments
  - Web SSO MEX (Metadata Exchange Protocol)
  - Web SSO Interoperability Profile
  - 3GPP plans integration of Liberty into USIM cards


History of Federation Standards

Integration into SAML 2.0



SAML 2.0

Liberty ID-FF 1.2

Shibboleth 1.2

OASIS SAML 1.1

WS-Federation

Federation Standards

XML, WSDL, WS-Security, XML Dsig, WS-*, SPML, XACML, XKMS

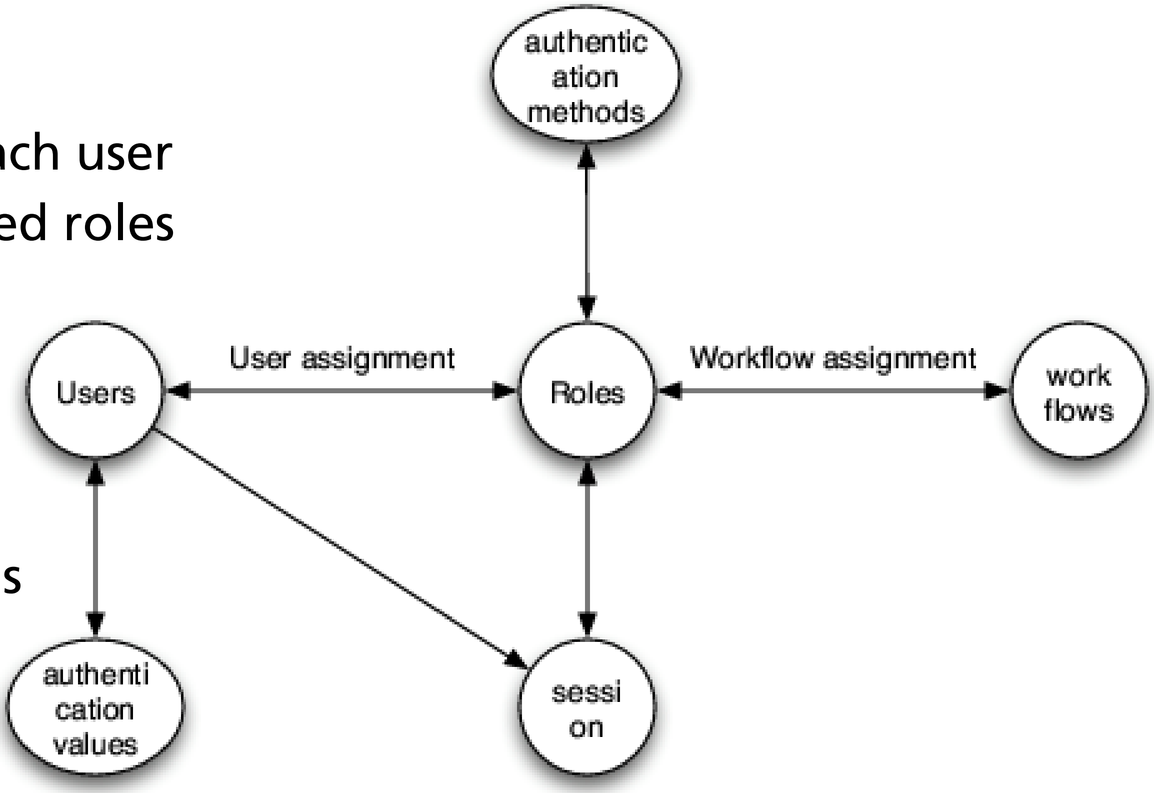
XML & Security Standards

SOAP, SSL/TLS, HTTP, HTML, WAP

Binding, Network, Transport Standards

Authorisation Concept

- We use a special variation of Role Based Access Control
  - Static explicit access rights
  - Protected object: workflows
  - workflows grouped by roles
  - Users are mapped to roles, each user owns all rights of all associated roles
- Workflows
  - implicit dynamic access rights
  - Security module decides if an access is allowed based on the input and the previous shown page (state)
  - 'Minimal need to know' policy



Workflows

- Finite state machine describing the transitions between dynamically created Web-pages

- Each shown page represents a state in the machine description

- Authorisation and access control constraints are
  - Parameters returning from the user interaction
  - State of the database of the host application

- **Fallback in case of auth. failure**
  (In standard Web applications there is a starting page)

- FSM needs concurrently active states

Workflow issues

Parallel Workflows

- Every Workflow describes single task in the program usage e.g. performing a transfer in a banking environment
- As all possible workflows starting with the same initial page all workflows have this page in common.
- Two workflows can have more pages in common

Exclusive Workflows

- One user can have different active sessions
- Each session owns a set of active workflows
- Enables the definition of two (or more) workflows as exclusive, implements a variation of a chinese wall security policy
- Extends the RBAC concepts by more flexible constraints

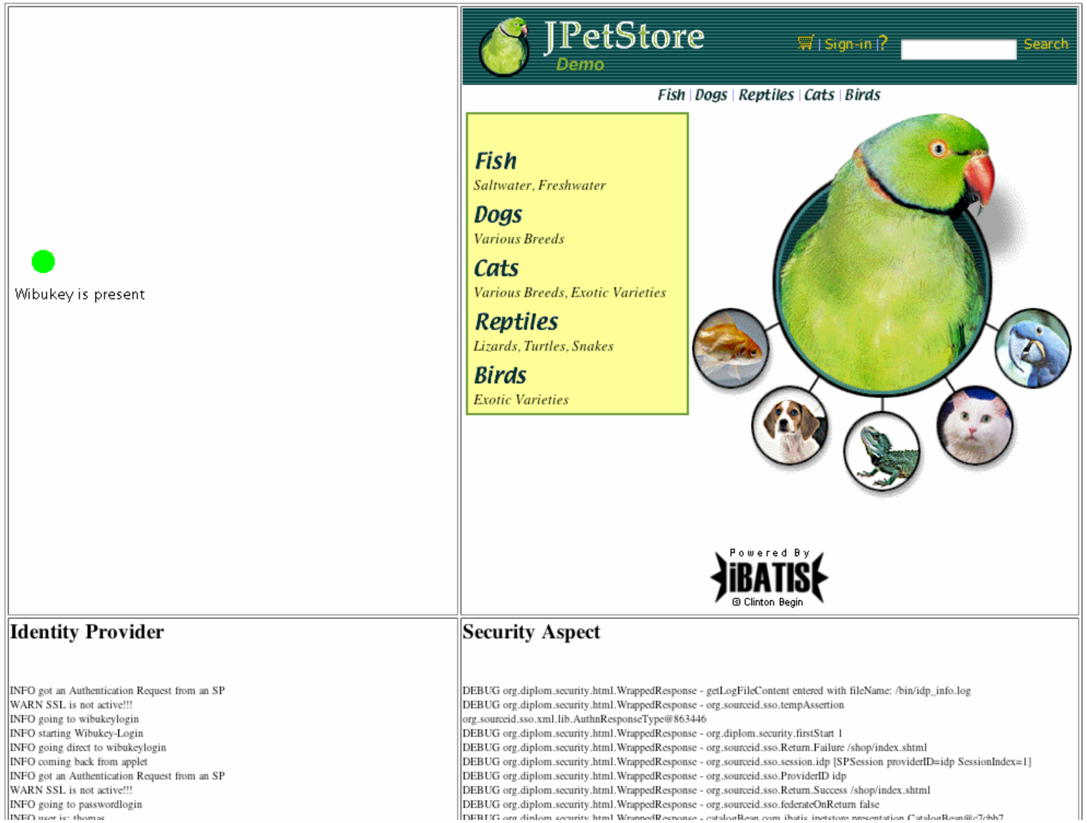
Exemplified Authentication: Hardwaretoken

- Simplified authentification by
- USB-Token, SmartCard, ...
- Reauthentication
- Zero-interaction
- security concerns require proactive methods

Pro-activity

- Methods “Working on behalf of the user acting on their own initiative”
- Common: User leaving authentication tokens plugged when leaving workplace

- Education of users by *messages*
- Aiding users by *auto log-off*
- Integration through the HTMLRewriter, embedding Applet
- Yields audit data



Identity Provider

```
INFO got an Authentication Request from an SP
WARN SSL is not active!!!
INFO going to wibukeylogin
INFO starting Wibukey-Login
INFO going direct to wibukeylogin
INFO coming back from applet
INFO got an Authentication Request from an SP
WARN SSL is not active!!!
INFO going to passwordlogin
INFO user is: thomas
```

Security Aspect

```
DEBUG org.diplom.security.html.WrappedResponse - getLogFileContent entered with fileName: /bin/idp_info.log
DEBUG org.diplom.security.html.WrappedResponse - org.sourceid.sso.tempAssertion
org.sourceid.sso.xml.lib.AuthnResponseType@863446
DEBUG org.diplom.security.html.WrappedResponse - org.diplom.security.firstStart 1
DEBUG org.diplom.security.html.WrappedResponse - org.sourceid.sso.Return.Failure /shop/index.shtml
DEBUG org.diplom.security.html.WrappedResponse - org.sourceid.sso.session.idp [SPSession providerID=idp SessionIndex=1]
DEBUG org.diplom.security.html.WrappedResponse - org.sourceid.sso.ProviderID idp
DEBUG org.diplom.security.html.WrappedResponse - org.sourceid.sso.Return.Success /shop/index.shtml
DEBUG org.diplom.security.html.WrappedResponse - org.sourceid.sso.federateOnReturn false
DEBUG org.diplom.security.html.WrappedResponse - catalogBean com.ibatis.jpetstore.presentation.CatalogBean@c7cbb7
```

Demonstrated Security Methods

Authorisation Constraints depending

- on input values using regular expressions
- on host database states with SQL-queries

Authentication Methods

- Username/password
- USB token, (Smartcard), …
- Forced re-authentication after idle time limits (pro-activity)

- Transport layer security (SSL)

Conclusion

- Security functionality can be developed separately and later added to an existing system
- without (much) knowledge of the system's source code
- Aspect oriented programming offers new concepts for implementing full-fledged security exclusively utilising distributed services

'AOP enables AAA in an SOA'