Single Sign On

Wenli He (wenli-he@uiowa.edu)

Spring 2007 22C196:001

March 06, 2007

Papers

- D. P. Kormann and A.D. Rubin, *Risks of the Passport Single Signon Protocol*, Computer networks 33 (2000) 51-58
- V. Welch et al, *Security for grid services*, Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing (HPDC'03)

What is Single Sign-On (SSO)

- Definition from wikipedia
  - **Single sign-on** (SSO) is a specialized form of software authentication that enables a user to authenticate once and gain access to the resources of multiple software systems.
- Definition from whatis.com
  - **Single sign-on** (SSO) is a session/user authentication process that permits a user to enter one name and password in order to access multiple applications. The process authenticates the user for all the applications they have been given rights to and eliminates further prompts when they switch applications during a particular session.
- Fundamentally, single sign-on authentication means the sharing of authentication data.

Why Single Sign-On?

- **Web:** Proliferation of password-protected online services on Web
  - Email service
  - Online bank
  - Online store
  - Online office tools
- **Enterprise:** introduce more and more protected resources and applications into enterprise
  - Databases
  - Customer relationship management (CRM) systems
  - Supply Chain Management systems
- **PC/Workstation users**: access to protected applications
  - Windows application (example: outlook express)
  - Web application (example: online bank, email service)
  - Host-based Application (example: PuTTY)

Current Work

- Web single sign on (WebSSO, mostly free)
  - Passport (Microsoft)
  - CAS (Central Authentication Service ,Yale University)
  - CoSign (University of Michigan)
- Enterprise (domain) Single Sign-On
  - Kerberos (Authentication method and SSO solution)
  - GSI (Grid Security Infrastructure)
- Password Manager (mostly commercial)
  - Citrix (citrix.com)
    http://paths.citrix.com/pwm/ssogoogle/E/sso/fulfill.aspx
  - **WiseGuard SSOWatch (evidian.com)**
    http://www.evidian.com/wiseguard/demo1/index.htm

Microsoft Passport System

- A WebSSO System

Passport Model

- Three entities
  - Client at a Web browser
  - The merchant (Web) server
  - Passport login server (central)

    Maintains :
    - Authentication information
    - Client profile (address, shoe size)
    - Wallet information (credit card)

Authentication Dialogue

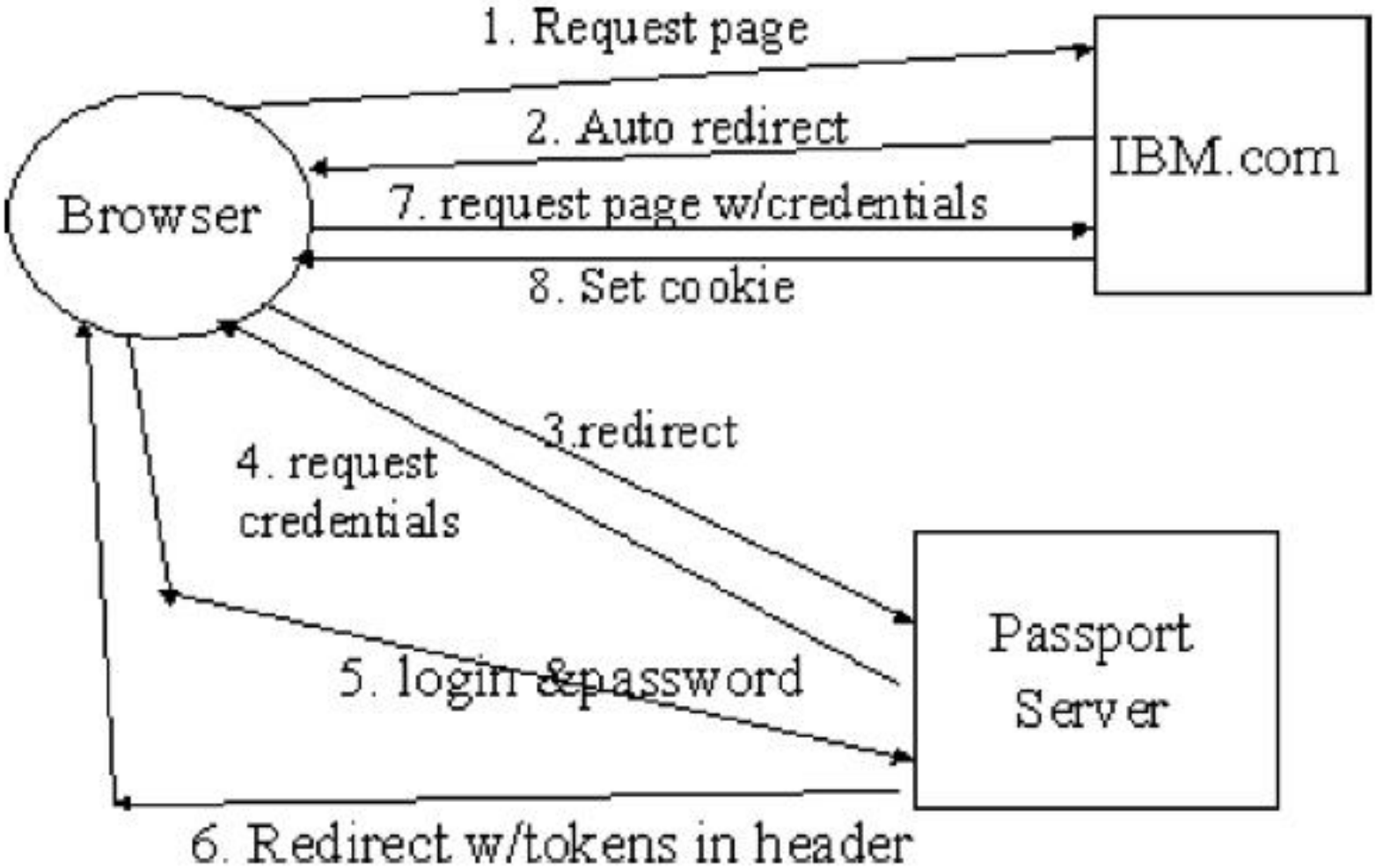


Fig. 1. The Passport architecture.

Authentication Dialogue (II)

(1) User visits a merchant site and needs to authenticate;

(2)(3) Merchant Web server redirects the customer's browser to a well-known Passport Server;

(4) Passport server presents the user with a login page over an SSL connection;

(5) User provides credential to Passport Server;

(6)(7) Passport server redirects the user back to the end server (authentication information included in the redirect message in the query string);

(8) End server sets an encrypted cookie in the client's browser.

Authentication Model

- Passport server shares a triple DES key with each merchant server
- Cookies are decrypted using triple DES with the key
- Exchange of credentials between end user and Passport server is over an SSL (Secure Sockets Layer) connection

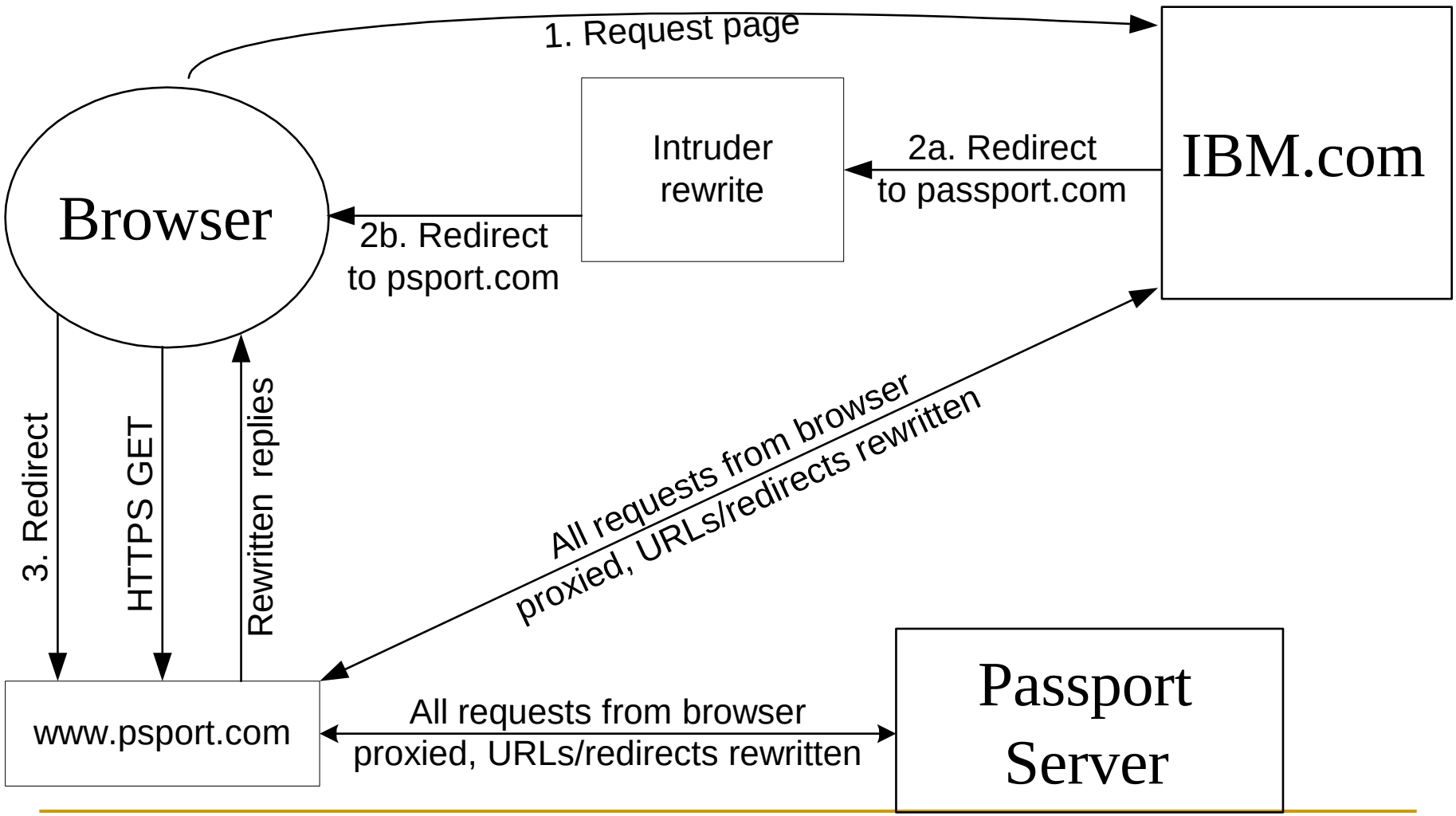
Risks

- User Interface (logout)
- Key management
  - generate keys, transfer keys
  - Single key used to encrypt all cookies (key exposure, suggest master key to generate key for each client)
- Central point of attack (centralized service model)
  - Suggest replicate of service
- Cookies and Javascript
  - Cookie can be recovered and reused (within time window in the cookie) if user forget to logout
  - User privacy
- Persistent cookies
  - Passport leaves “authenticator”, cookies, on the client machine
  - Compare with Kerberos
- Automatic credential assignment

Attacks

- Bogus merchant
  - careless/inexperienced user
  - Example: Redirect to pasport.com (instead of passport.com)
- Active Attack (figure)
- DNS Attack
  - SSL depend on the DNS
  - HTTP redirects specify a host to receive the redirection in form of a DNS name
  - Solution: DNSSEC (digitally signed DNS information)

Attack (II)


1. Request page
Browser
Intruder
rewrite
2a. Redirect
to passport.com
IBM.com
2b. Redirect
to psport.com
3. Redirect
HTTPS GET
Rewritten replies
All requests from browser
proxied, URLs/redirects rewritten
www.psport.com
All requests from browser
proxied, URLs/redirects rewritten
Passport
Server

Reliability

- How Vulnerable is the system to failure of a single component?
  - Network or server failure
- How easy is it to run multiple instances of that component to reduce the risk of failure (i.e., can a resilient infrastructure be built?)
  - Active-passive configuration
  - Multi-master setup
  - Standby server

Security

- How secure does it seem to be and/or what are its vulnerabilities ?
- What form of encryption is used and where is it used
- How dependent is the protocol on the security of the underlying transport? Will SSL be required for all services
- How vulnerable is it to man-in-the-middle and impersonation attacks

Related Work (WebSSO)

- Two major ways to implement a single sign-on system
  - Cookies
    - Security
      - the ability to intercept a cookie and for a third party to use it is probably its biggest weakness and mandates the use of a secure transport layer.
    - Usability:
      - cookies are handled well by all browsers and if cookies are accepted by the user, their use is altogether more transparent than the use of X.509 certificates
  - X.509 Certificates
    - Security:
    - Usability:
      - client code must be installed on the end-user's system

Related Work (Single Sign-On for Web)

- **CAS** (Central Authentication Service)
  - Originator: Yale University: http://www.yale.edu/tp/auth/.
- **Pubcookie**
  - Originator: Pubcookie Team originally from the University of Washington: http://pubcookie.org/.
- **WebAuth**
  - Originator: Stanford University http://webauthv3.stanford.edu/.
- **Cosign**
  - Originator: University of Michigan: http://www.umich.edu/~umweb/software/cosign/.
- **KX.509** (X.509 certificates via Kerberos)
  - Originator: University of Michigan: http://www.umich.edu/~x509/.
- **A-Select**
  - Originator: Surfnet (Netherlands): http://a-select.surfnet.nl.

Security for Grid Services

Single Sign-On using GT2 and GT3

Terminology

- Grid
- GT (Globus Toolkit)
- GSI (Grid Security Infrastructure)
- OGSA (Open Grid Services Architecture)
- GRAM (Grid Resource Allocation and Management)

Characteristics of Grid

- User population is large and dynamic
- Resource pool is large and dynamic
- A computation consists of dynamic group of processes during execution
- Variety of communication mechanisms, including unicast and multicast, for a computation
- Resources may require different authentication and authorization mechanisms and policies (Kerberos, plaintext passwords, Secure Socket Library and secure shell)
- Different local name spaces, credentials, or accounts, at different sites for individual user
- Resources and users may be located in different countries.

Grid Security Model

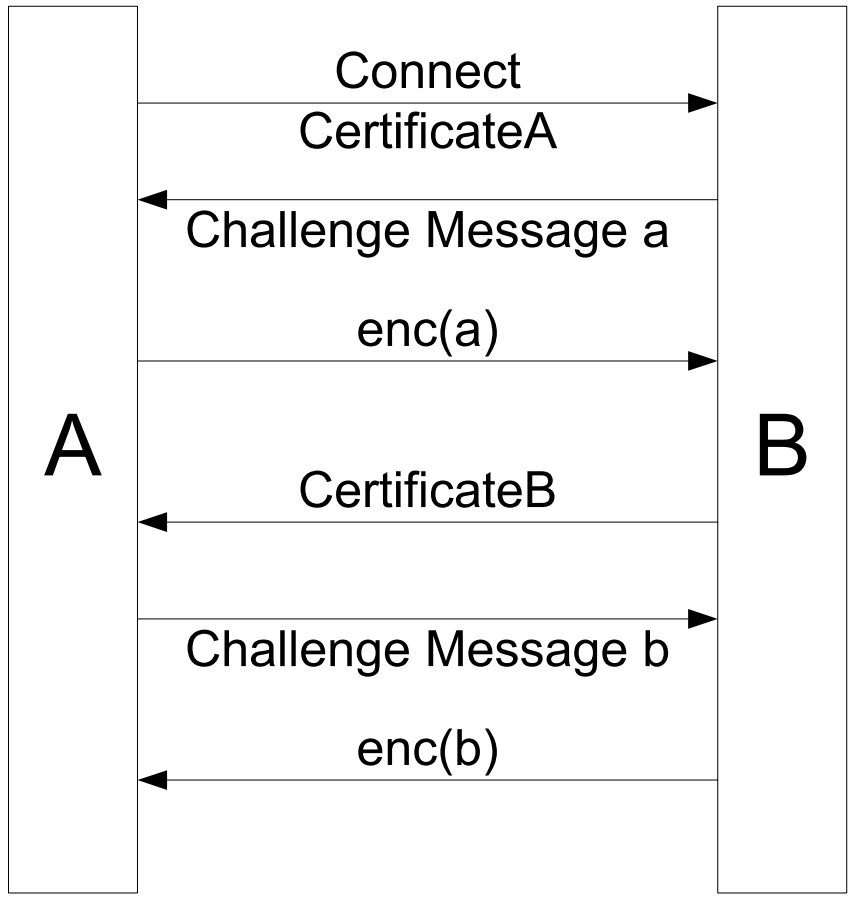
- 3 Key Functions in a Grid security model
  - Multiple security mechanisms
  - Dynamic creation of services (delegation)
  - Dynamic establishment of trust domains (skip)

GSI in GT2

- GSI is Composed of 4 distinct functions:
  - Authentication
    - Credential set=X.509 identity certificates + associated private key
    - TLS-based protocol
    - Gateways are used to translate between the common GSI infrastructure and local site mechanism (example: KCA)
  - Authorization
    - Grid-map file
      - Provide access control based on a list of acceptable user identifiers
    - CAS (Community Authorization Service) assertions
  - Message Protection (encryption, integrity checking)
    - TLS-based protocol
  - Delegation
    - X.509 proxy certificates

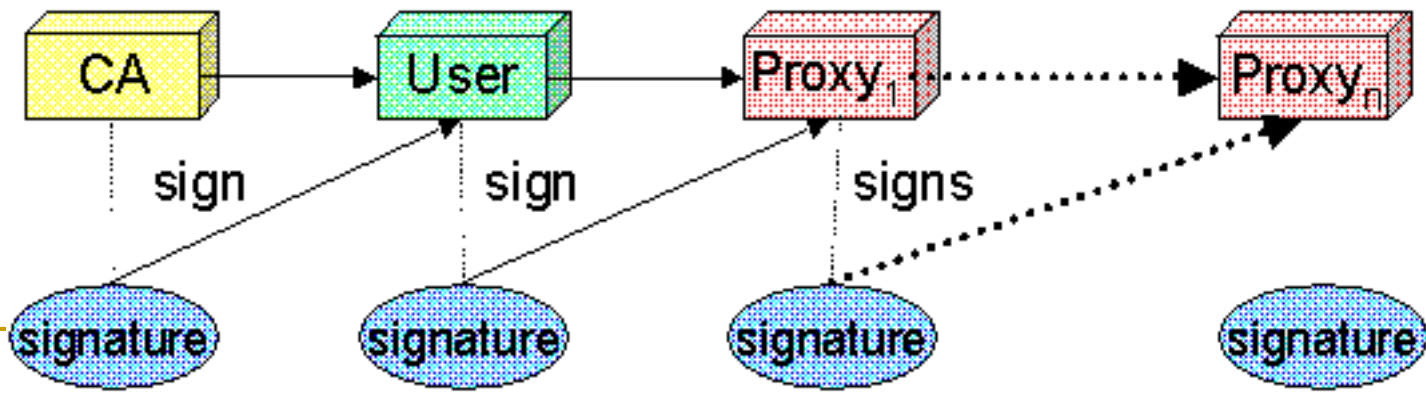
Mutual Authentication

- Every user and service on the Grid is identified via a certificate (encoded in X.509 certificate format)
- Trusted CA
- Uses the Transport Layer Security (TLS) for its mutual authentication protocol



Delegation

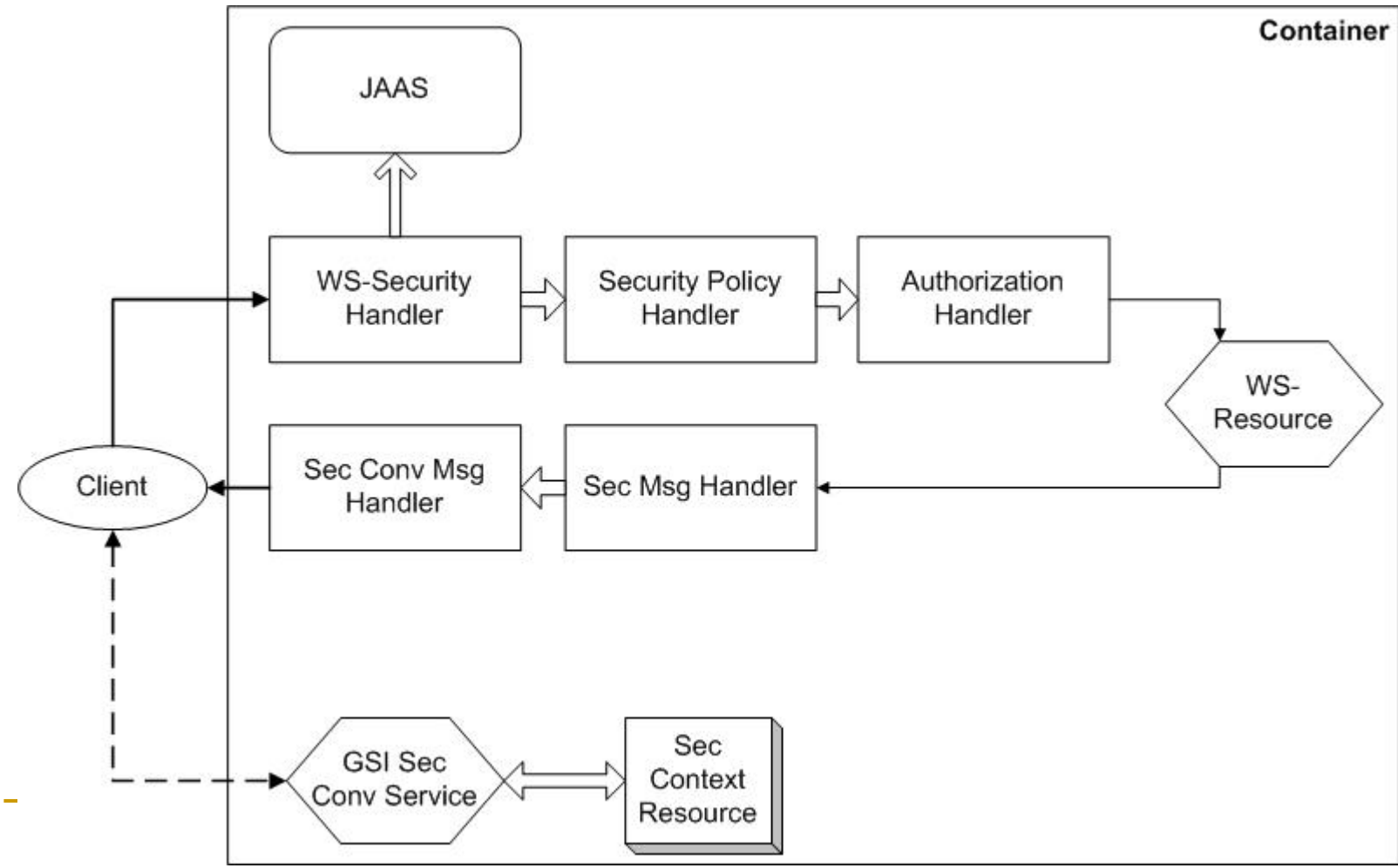
- When used?
  - A Grid computation requires that several Grid resources be used (each requiring mutual authentication)
  - there is a need to have agents (local or remote) requesting services on behalf of a user
- How? Proxy
  - A proxy consists of a new certificate (with a new public key in it) and a new private key , signed by the owner
  - Mutual authentication with proxy slightly different



GSI in GT4 (similar to GT3)

- Pre-WS and WS components
  - Applying WS technology (e.g., SOAP, WSDL)?
  - similar authentication, delegation, and authorization mechanisms
  - fewer authorization options for pre-WS
  - Pre-WS components are similar to GT2

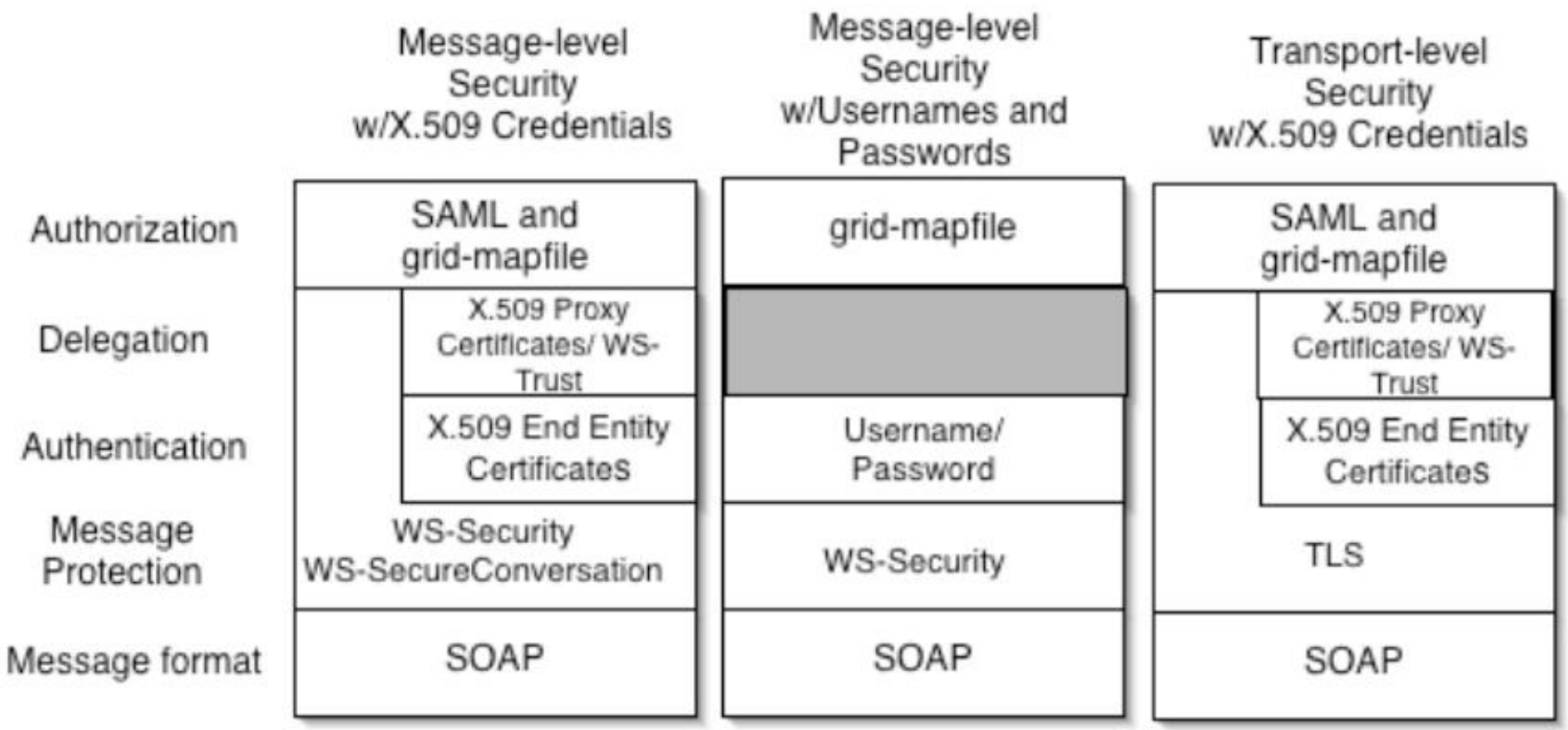
Message-level security (server side message processing)



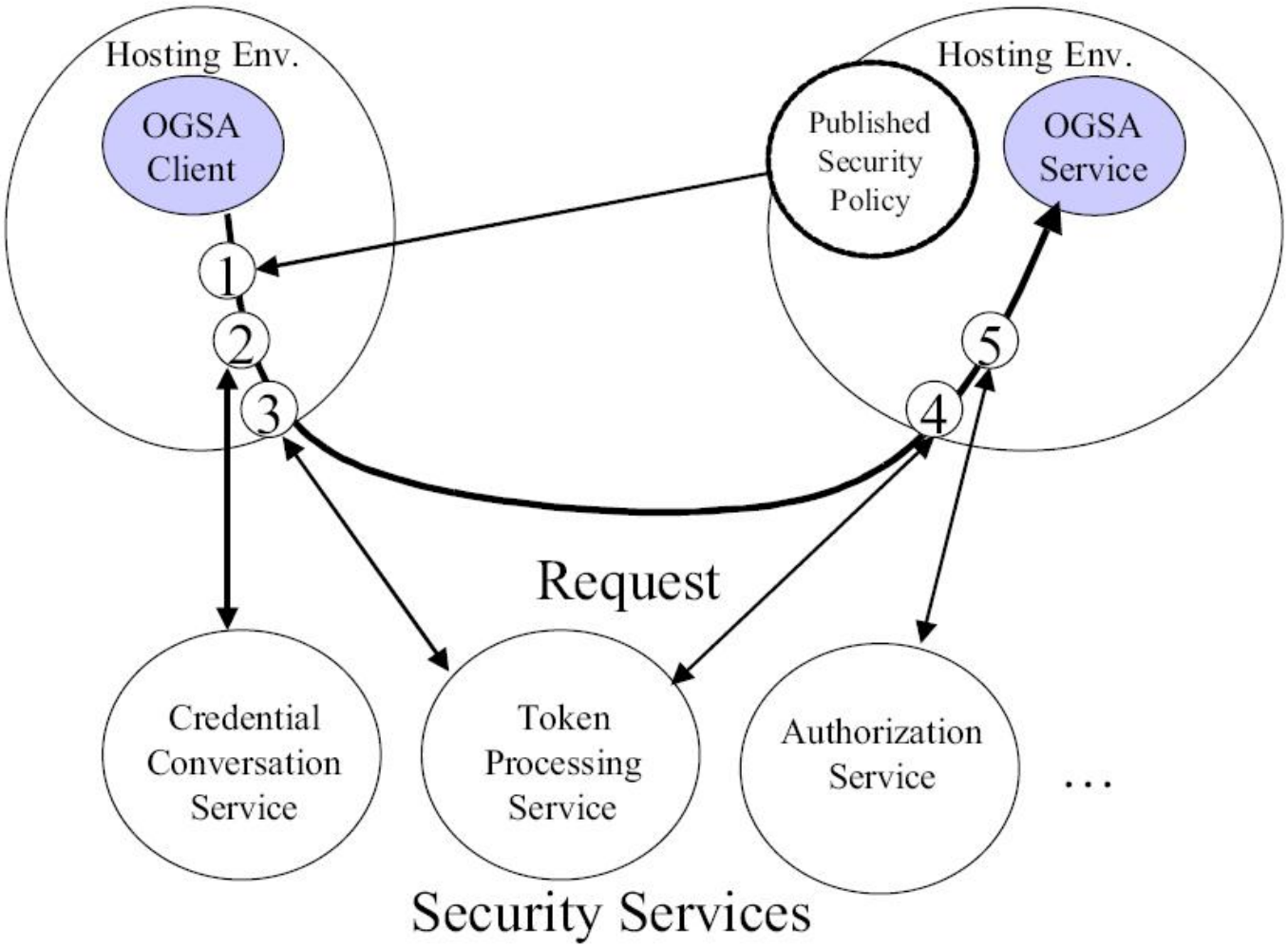
GT4 GSI for WS components

- Message-level and transport-level security (new in GT4)
- Container level authorization
  - allows chains of authorization modules with well defined interfaces to be associated with various entities, e.g. services, in the container
  - different authorization module implemented in GT4
    - grid-mapfile based authorization module
    - a module that uses the SAML protocol to query an external service for an authorization decision (new in GT4 WS)
  - Allow user customized authorization modules

GT4 GSI
and Standards used for Security Functions



Example of a secured request in the OGSA security model



References

- D. P. Kormann and A.D. Rubin, *Risks of the Passport Single Signon Protocol*, Computer networks 33 (2000) 51-58
- V. Welch et al, *Security for grid services*, Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing (HPDC'03)
- The Globus Security Team, *Globus toolkit version 4 grid security infrastructure: A standards perspective*, www.globus.org/toolkit/docs/4.0/security/GT4-GSI-Overview.pdf, 2005.

Thanks!

Backup slides

GSI in GT3

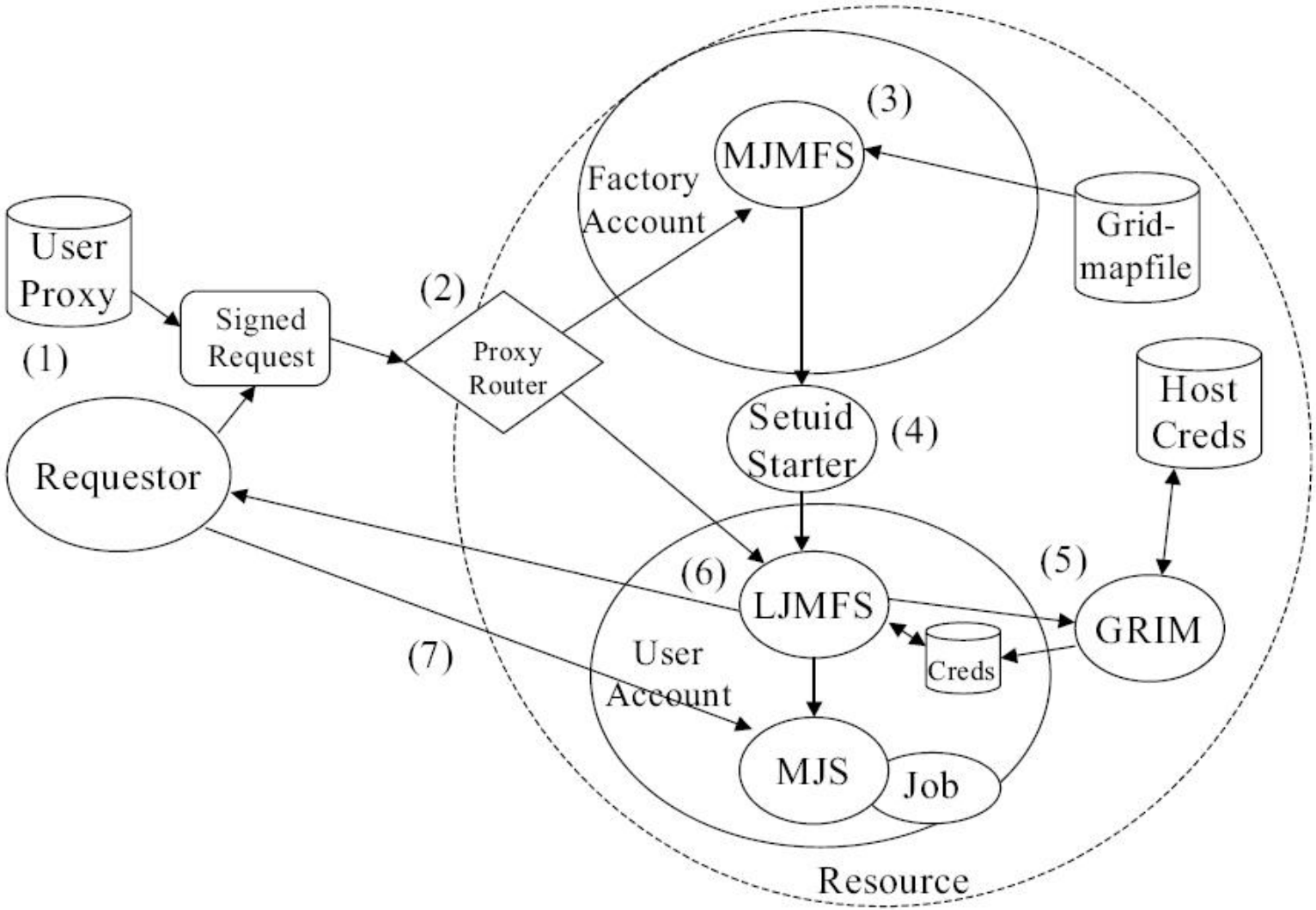
Open Grid Services Architecture

- Developed within the Global Grid Forum (GGF).
- Describes an architecture for a service-oriented grid computing environment for business and scientific use
- OGSA is based on several other Web service technologies, notably WSDL and SOAP.
- A refinement of the emerging Web Services architecture, specifically designed to support Grid requirements

GT3 Security Model for OGSA

- Security functions as services
  - Credential processing service
  - Authorization service
  - Credential conversion service
  - Identity mapping Service
  - Audit

GT3 GRAM Implementation



Discussion

- Any security issues in all these 3 GSI versions?

Question

- Do you think the GRAM implementation consistent with the security model presented before?