Jetspeed-2 Portal Federated Authentication
SSO with Jetspeed-2

David Sean Taylor
Apache Portals Meetup, Amsterdam April 7 2008

Federated Authentication

The Need for Federated Authentication

- The average user interacts with all sorts of social, business, financial and government agencies digitally. Each of these identity services require their own ID and password as a user authentication process. Most servers also require additional user identity information to be provided to each entity which the user interacts with.
- As a result the user is increasingly frustrated with:
  - Having to remember more and more user Ids and passwords
  - Providing more identity information than they would otherwise chose to each service
- Companies become more frustrated with:
  - Maintaining multiple configurations of authentication
  - Duplicated identity store and access controls information

Solving the problem

- Federated Authentication solves these problems by:
  - Providing a single point of web sign-on across multiple domains (Web SSO)
  - Extending SSO Across Application as well as organizational boundaries
  - Enabling users to authenticate from one organization into another via a trust (Circle of Trust)
  - Providing for a means for users or entities to have a single, federated identity across applications
  - Working with standards such as SAML to enable applications to inter-operate based on a set of standards

Federations

- Federations establish trust between two or more entities
- Federations make assertions about identities
- Federations protect user privacy
- A federation is a group of organizations (universities, corporations, content providers, etc.) who agree to exchange attributes using a common protocol such as the SAML/Shibboleth protocols
- Federations abide by a common set of policies and practices. In so doing, they must implicitly or explicitly agree to a common set of guidelines. Joining a federation is not explicitly necessary for operation of SSO, but it dramatically expands the number of federated partners that can interact without defining bilateral agreements between all these parties.

Single Sign on Benefits

- Ability to enforce uniform enterprise authentication and/or authorization policies across an suite of applications as well as with partner applications
- End to end user audit sessions to improve security reporting and auditing
- Removes application developers from having to understand and implement identity security in their applications

Single Sign on Example

- Single Sign On Use Case

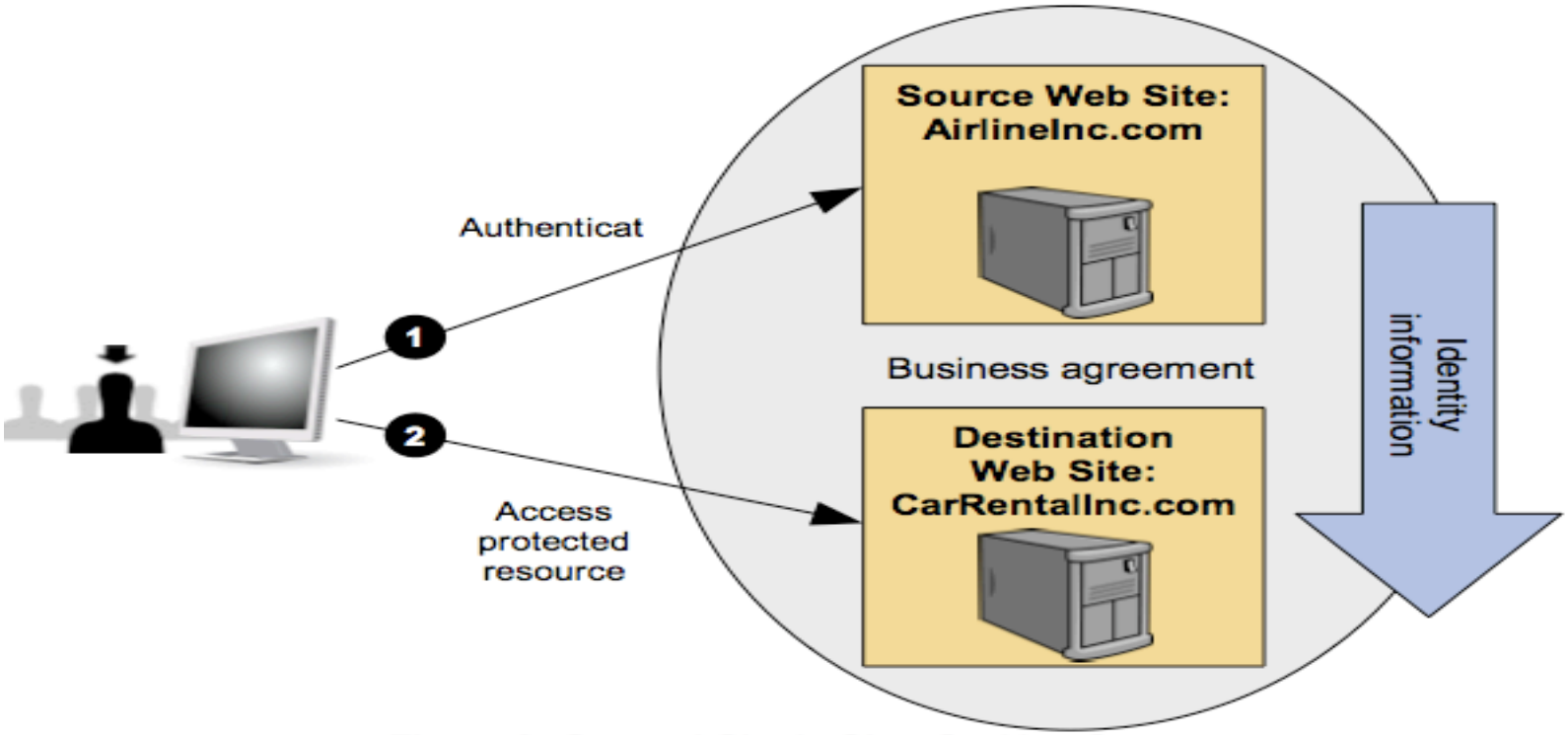


Figure 2: General Single Sign-On Use Case

SAML

The OASIS Security Assertion Markup Language (SAML) standard defines an XML-based framework for describing and exchanging security information between on-line business partners.

- This security information is expressed in the form of portable SAML assertions that applications working across security domain boundaries can trust.
- The OASIS SAML standard defines precise syntax and rules for requesting, creating, communicating, and using these SAML assertions.
- SAML solves the MDSSO problem by providing a standard vendor-independent grammar and protocol for transferring information about a user from one web server to another independent of the server DNS domains.

Open Source Federated Solutions

- Shibboleth (Federated, SSO, Identity Management)
- Athens Eduserv (Federated, SSO, Identity Management)
- Open SSO / Open Federation (Federated, SSO, Identity Management)
- Apache Triple-sec (SSO, Identity Management)
- A-Select / DigiD (SSO, Identity Management)
- JOSSO (SSO, Identity Management)
- Kerberos, NTLM (SSO, Identity Management)

Jetspeed Supports

- Jetspeed has been tested with the following solutions:
- Athens Eduserv
- A-Select / DigiD
- NTLM SSO
- Shibboleth
- JAAS Login Module

Jetspeed and Security Standard

- Jetspeed Security is based on Security Standards:
  - Java Security API
  - JAAS API (Java Authentication and Authorization)
    - Login Module
    - Java Security Policy
  - Java Permissions
  - Jetspeed Constraints
  - Jetspeed runs all portlets under a protected Java Security Policy
  - All resources in a portal can be secured by policy

Jetspeed Security and JAAS

- Jetspeed leverages JAAS Authentication and Authorization standards through the implementation of
  - A default **Login Module** for authorization. Jetspeed works out of the box with its own login module.
  - A **JAAS Authorization Policy** based on a portal permission set**.**
  - The authorization policy is stored in a relational database, making for a live and updatable security policy

Jetspeed Security Integration

- Jetspeed has several integration solutions to securing the portal:
  - Built-in solutions:
    - Login Portlets interacting with Jetspeed's Java Login Module (default)
    - Login Portlet without Java Login Module
  - CMA - Container Managed Authentication
    - Integration with Java Login Module implemented outside Jetspeed
  - External Intranet Solutions
    - Custom authentication to corporate identity servers
    - NTLM or Kerberos authentication
  - Federated External Identity Provider Solutions
    - Shibboleth
    - Athens Eduserve

Levels of Jetspeed Security Integration

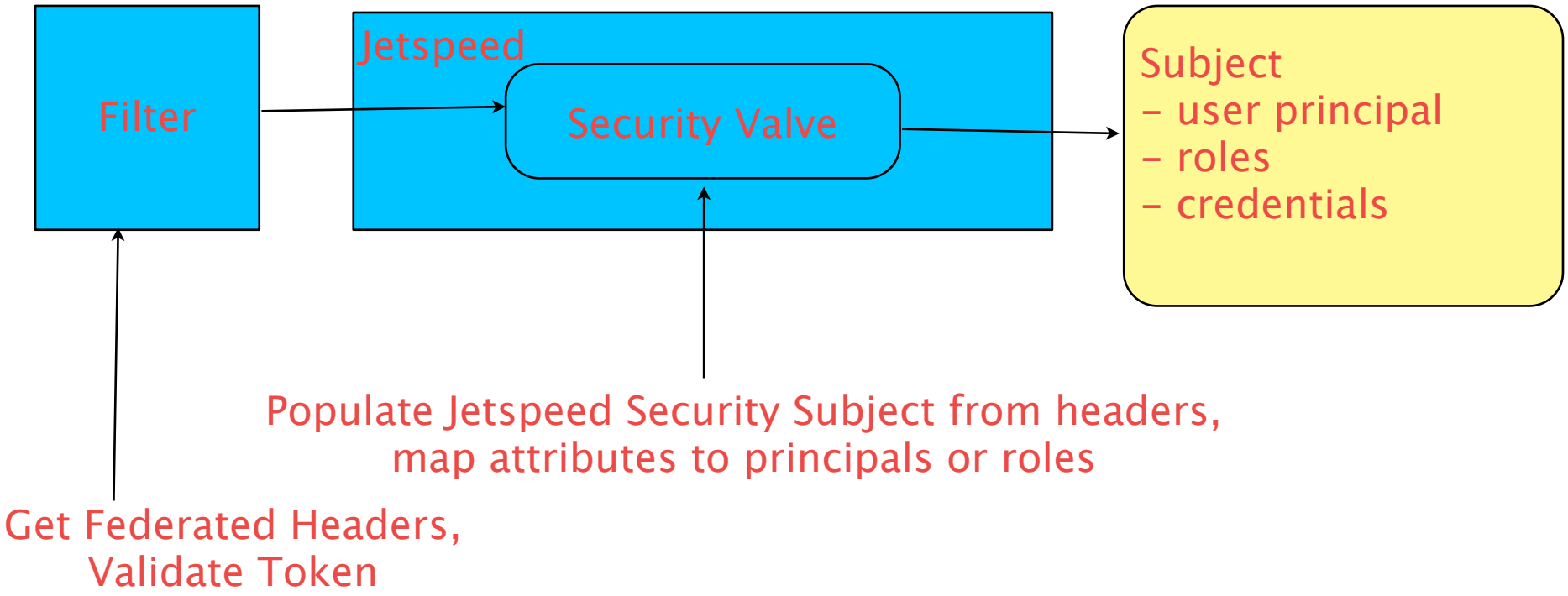
- **Security Integration**
  - We can also look at the security integration from another POV, at what level to you integrate with Jetspeed...:
  - **Default Integration**
    - Use 100% Jetspeed Security services and management portlets out of the box
  - **Service-Level Integration**
    - implement some level of Jetspeed Security APIs such as User Manager, Role Manager...
    - required if you want Jetspeed to manage, via Administrative portlets, your users that exist in an external credentials store
    - with service-level integration, you can access your security information using the Jetspeed API
  - **Pipeline-level Integration**
    - only integrate at the authentication level, using a Servlet Filter and/or Jetspeed Security Valve
    - provide a JAAS Subject to Jetspeed from the filter or valve
    - Do not integrate with administrative portlets
    - Do not integrate with Jetspeed Security APIs
    - limited access to your security model from the portal

Jetspeed Authentication

- **Authentication** establishes the identity of the user and populates the Subject with all the user principals.
- The populated Subject is added to the session in the Security Valve implementation. The Subject principals are then used to authorize the user's access to a given resource. Jetspeed Security leverages JAAS authorization by checking the user's permission with the Java Access Controller.
- **Authorization** - not fully implemented by federated solutions. Future direction: XACML - Extensible Access Control Markup Langage
- In a federation solution, we need to override Jetspeed's default implementation on provide hooks to bring in a federated solution…

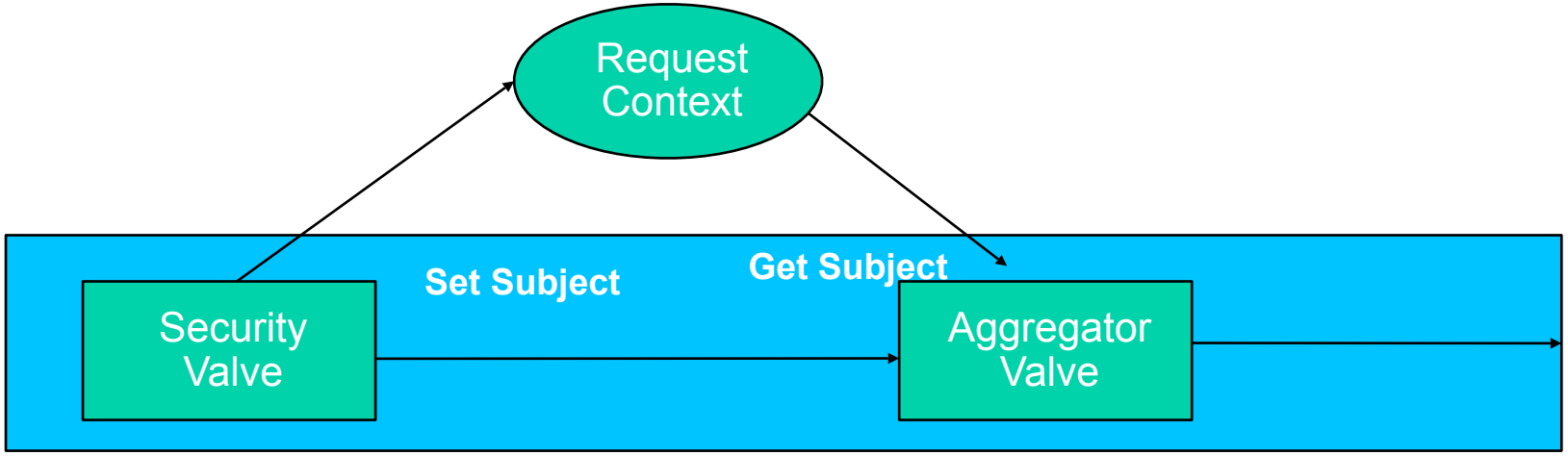
Pipeline-Level Integration

- Using Pipeline-level integration, integration points are at the servlet filter and Jetspeed security valve



Security Valve

- **Valves** are units of work along the pipeline workflow. Typically valves represent access to a Jetspeed feature or component, such as aggregation, security, action processing, or device capabilities.The Jetspeed Pipeline has a request context associated with the entire request pipeline. Using the Request Context API, valves can add or retrieve bits of information to the pipeline request process.



The Subject

- Contains all principals and credentials about an authenticated entity
- A Java Security standard class
- Can be used to execute protected and privileged operations in unison with a security policy (Jetspeed provides a security policy):
  - `Subject.doAsPrivileged(subject, new PrivilegedAction()`

Service-level Integration

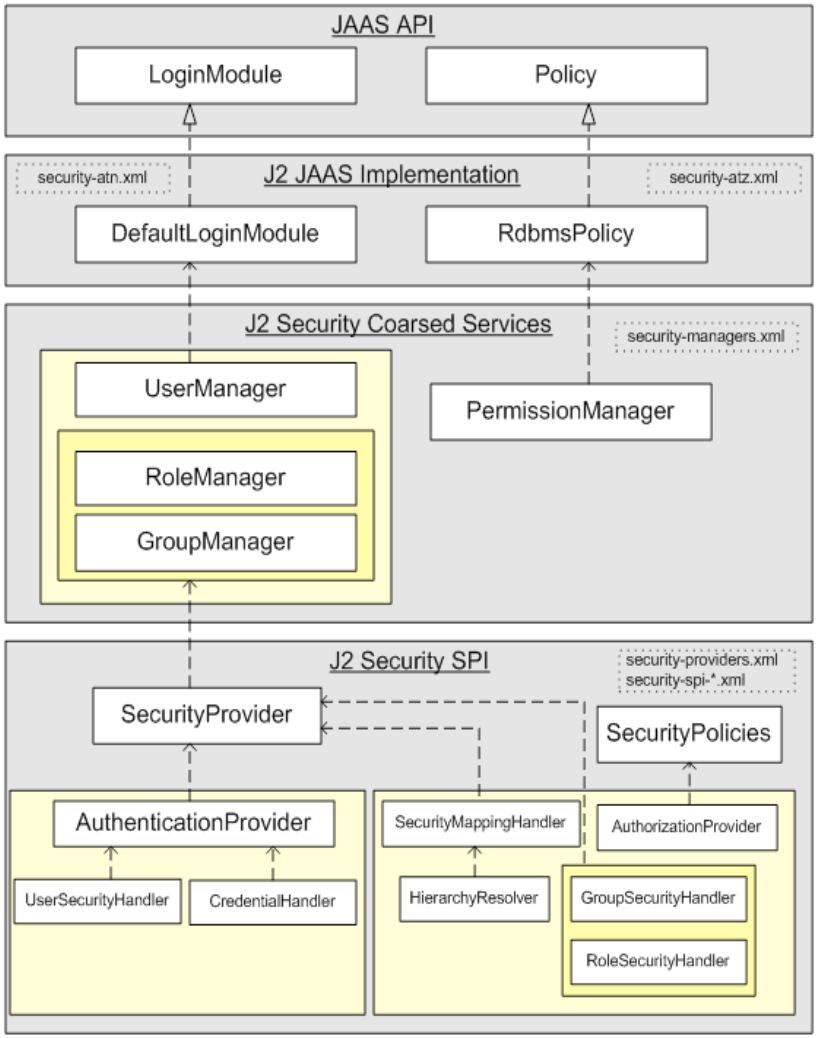
- Service-level integration is required when you need to integrate with Jetspeed Security services, or the Jetspeed administrative portlets for administering users
- Requires more programming, need to implement one or more Jetspeed Security services SPIs
- Gives finer integration with Jetspeed's administrative API and console.

Jetspeed Security Services

- Jetspeed Provides as both Services and Administrative Portlets
  - User and Credential Management
  - Role and Group Management
  - Permission Management (Access Control)
  - Constraints Management (Access Control)
  - SSO Management
  - User Profile Management
- When integrated with a federated provider, consider if your solution will require full support for all services

Layered Services

- **Jetspeed Security** can be accessed at the Java API layer: the JAAS API
- We implement both a Login Module and Security Policy
- Jetspeed Security Managers are configured in security-managers.xml
- Security managers are high-level APIs, used to manage users, roles, groups and permissions
- Jetspeed provides a Security Service Provider Interface (SPI) for layered handlers of security services. See the security-providers.xml and security-spi-*.xml



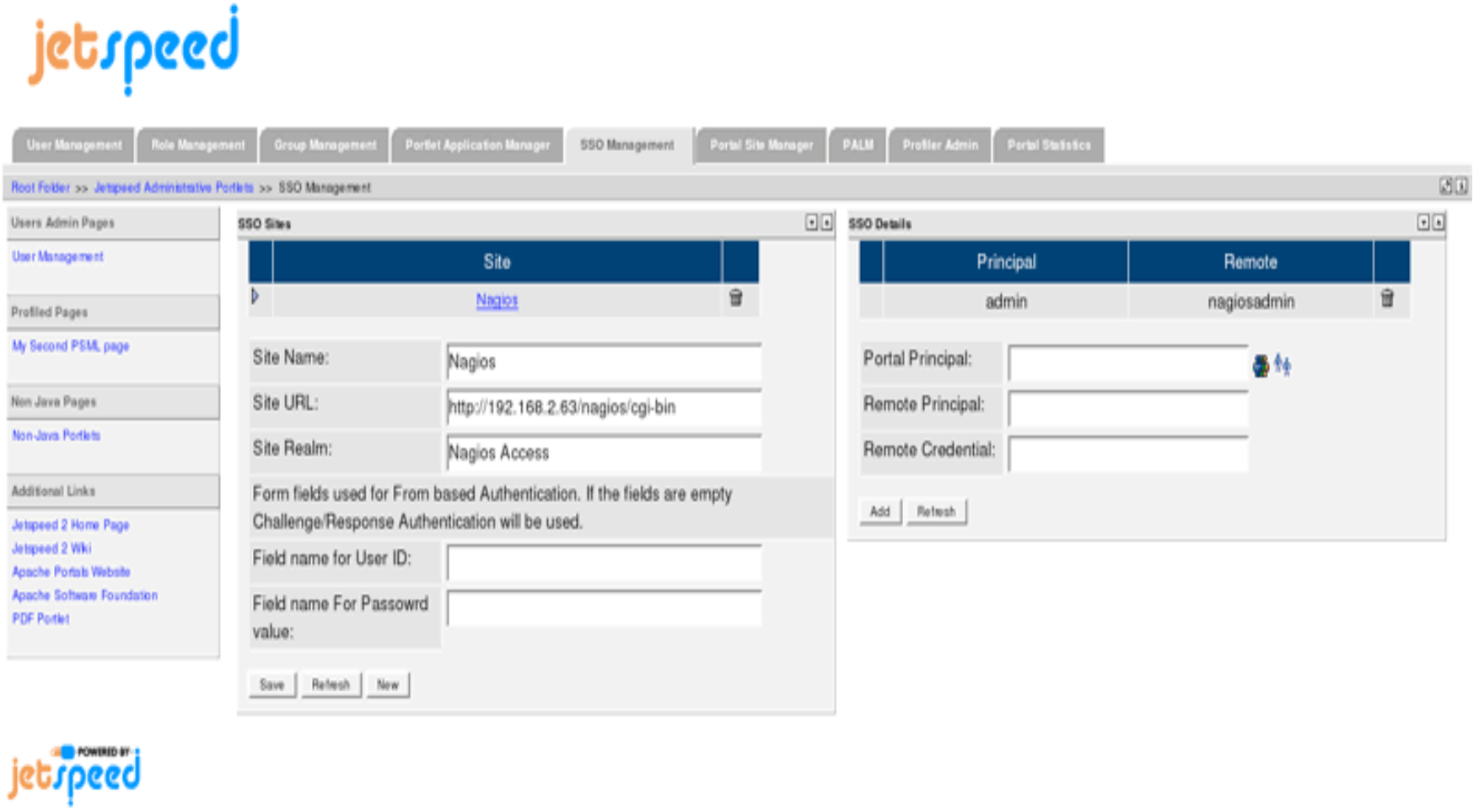
Jetspeed “Simple” SSO

If you are not able to integrate existing applications with your federated solution, you can always fall back on Jetspeed’s “Simple” SSO feature...

Jetspeed-2 Single Sign-on (SSO) feature is a credential store implemented as a component. It uses J2 security implementation for storing credentials. A management portlet allows the editing of SSO sites and remote credentials. It supports Basic Authentication and Form Based authentication and supports cookies.

The SSO Management administrative feature enables you to configure "single sign-on" access interactively.

Configuring Jetspeed SSO



Future Directions

- Provide Full Service-Level Support for federated interoperability with
  - Shibboleth
  - Triplesec
  - possibly OpenSSO
  - ...