Fast and Free SSO: A Survey of Open-Source Solutions to Single Sign-On

Craig Dickson, Software Engineering Manager
Naveen Nallannagari, Senior Consultant

Behr Process Corporation
www.behr.com

TS-4604

Goals of This Presentation

A survey of Open-Source Solutions to Single Sign-On

Present a sample of the different open source-based SSO solutions, critically compare and contrast them and provide tips on how to choose the right one to fit your needs.

Agenda

What Is SSO? (Briefly)

Survey of the Main Open Source Players

Head-to-Head Comparisons

Summary

Q&A

Agenda

What Is SSO? (Briefly)

Survey of the Main Open Source Players

Head-to-Head Comparisons

Summary

Q&A

What Is SSO?

It is definitely not…

"Every **Single** time you want to do something, you are going to have to **Sign-On**!"

—Your Sys Admin

What Is SSO?

This is more like it…

- Authenticate only once and access multiple resources
- Improved user productivity
- Improved developer productivity
- Ease of administration

What Is SSO?

But what about the downsides…

- Potentially creates a single point of attack
  - Malicious types only need 1 set of credentials and they can do a lot of damage
- Can be very difficult to retrofit existing applications and infrastructure with an SSO solution

Agenda

What Is SSO? (Briefly)

Survey of the Main Open Source Players

Head-to-Head Comparisons

Summary

Q&A

OpenSSO

Open Web SSO

- Mission of OpenSSO

  To provide an extensible implementation of identity services infrastructure that will facilitate Single Sign-On for web applications

- From the java.net community
- Focused on web-based single sign-on
  - A common starting point for many identity management projects

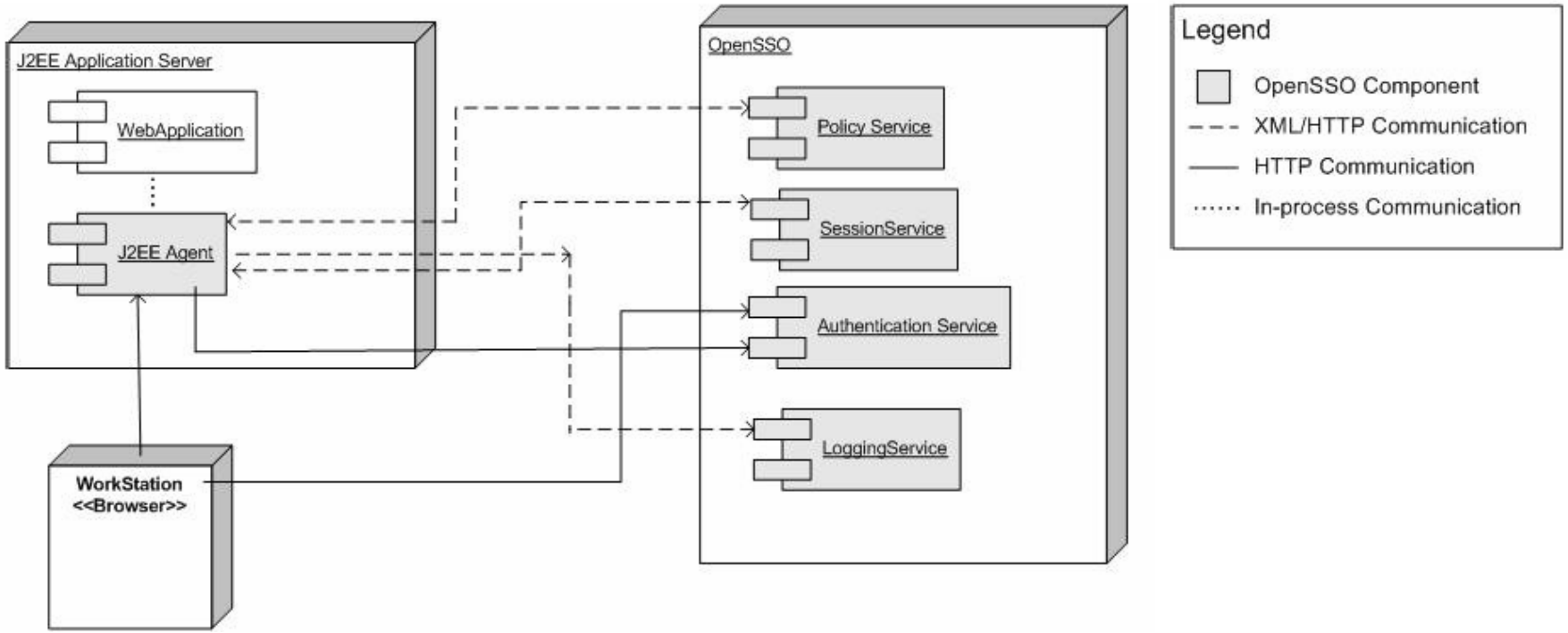
OpenSSO

Continued…

- Sun will make the following Sun Java™ System Access Manager modules freely available as part of OpenSSO
  - Authentication
  - Single-domain SSO
  - Web and Java 2 Platform, Enterprise Edition (J2EE™ platform) agents
  - Session management
  - Policy
  - Console
  - Administration tools
  - Federation
  - Policy agents

OpenSSO

OpenSSO Architecture



OpenSSO

OpenSSO Configuration

- Open SSO is deployed as only one application

  opensso.war

- After installation, configuration (name of host, protocol, etc.) can be done at:

  http://localhost:8080/opensso/configurator.jsp

- Realms have to be created

OpenSSO

OpenSSO Configuration

- Installation of Agent (e.g., Tomcat)
  - agentadmin - install
- Modify web.xml

```
<security-constraint>
   <web-resource-collection>
       <web-resource-name>Protected Resources</web-resource-name>
       <url-pattern>/secure/*</url-pattern>
   </web-resource-collection>
   <auth-constraint>
       <role-name>id=teste, ou=role, dc=opensso, dc=java, dc=net</role-name>
   </auth-constraint>
</security-constraint>
<login-config>
    <auth-method>FORM</auth-method>
    <form-login-config>
        <form-login-page>/authentication/login.html</form-login-page>
        <form-error-page>/authentication/accessdenied.html</form-error-page>
    </form-login-config>
</login-config>
<security-roll id="test">
    <role-name>id=test, ou=role, dc=opensso, dc=java, dc=net</role-name>
</security-role>
```

JOSSO

Java Open Single Sign-On

- Based on Java Authentication and Authorization Service (JAAS)
- Uses web services implemented with Apache Axis as the distributed infrastructure
- Uses Apache Struts and JavaServer Pages™ technology (JSP™ page) technology standards
- Comes with a Reverse Proxy component that can be used to create n-tier Single Sign-On configurations
  - Allows n-tier configurations using multiple strategies, including storing user information and credentials in LDAP, Databases, and XML files

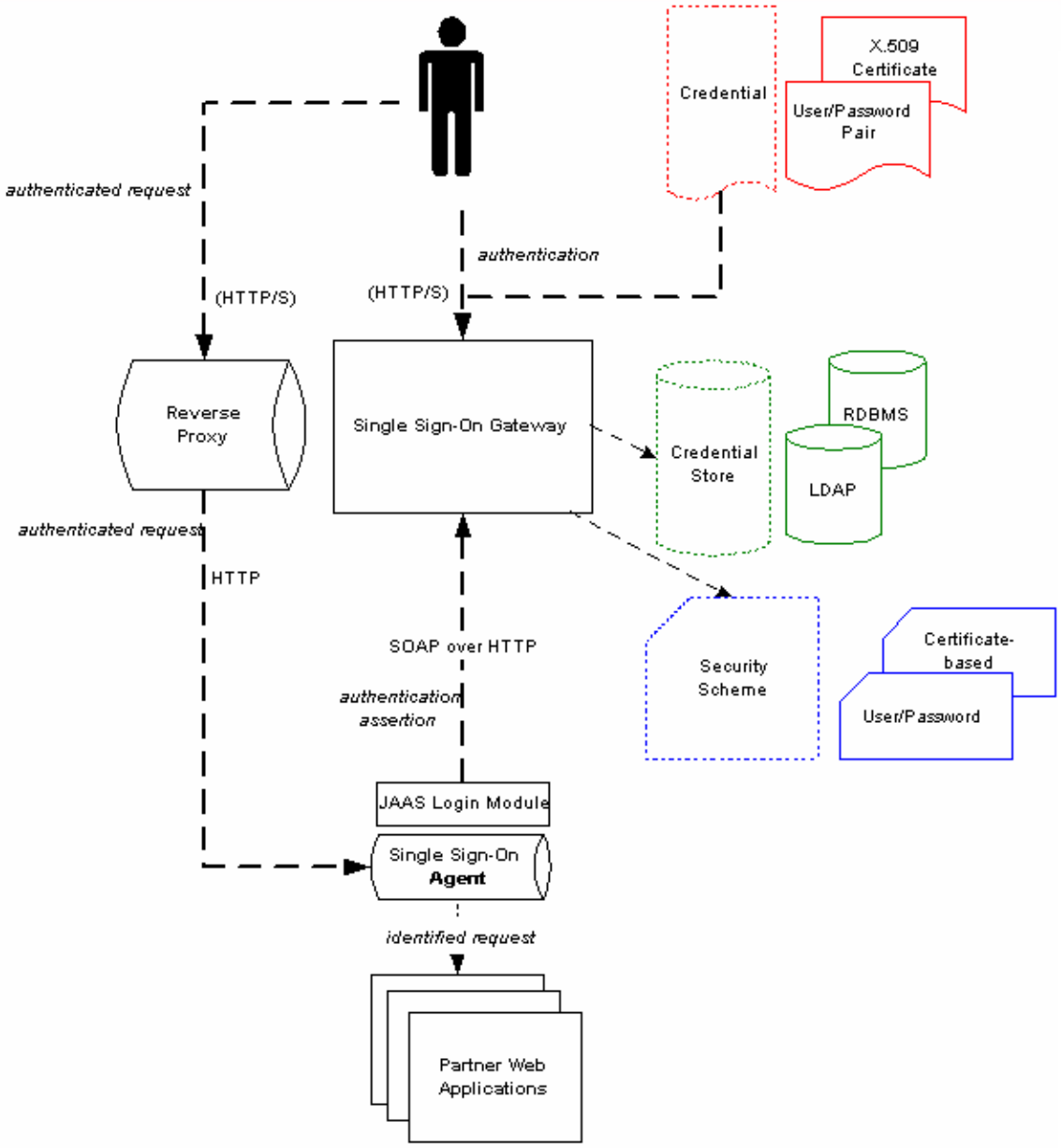
JOSSO

Continued…

- Implement and combine multiple authentication schemes with credential stores
- Credential Stores are repositories for user credentials, to be used during the user authentication transaction
- Can be configured to use (for example) certificate-based authentication scheme, obtaining user X.509 certificates from a database using Java DataBase Connectivity (JDBC™) software

JOSSO

JOSSO Architecture



JOSSO

JOSSO Configuration

- Integration of JOSSO with specific application Server (Tomcat or JBoss)
- Integrating Java Web Application with JOSSO

JOSSO

JOSSO Configuration—Integration with Tomcat or JBoss

- The Single Sign-On Gateway Configuration
  - Configuration file: **`josso-gateway-config.xml`**
    - Authenticator
    - Identity Manager
    - Session Manager
    - Audit Manager
    - Event Manager
- Single Sign-On Agent Configuration
  - To check that a previously user logged in is authorized to access a web context
  - Configuration file to declare the concrete configuration files:

    `$CATALINA_HOME/bin/josso-config.xml`

JOSSO

JOSSO Configuration—Integration with Tomcat or JBoss

- Protect a Web Application
  - Add to server.xml file

```
<Host>
    ...
    <Valve className="org.josso.tc50.agent.SSOAgentValve" debug="1"/>
    ...
</Host>
```

  - For each request to the `/partner` Web Context, the Single Sign-On Agent will intercept it, assert the Single Sign-On session, and obtain the user data from the Single Sign-On Gateway

JOSSO

JOSSO Configuration—Integration with Tomcat or JBoss

- Add a JAAS Realm
  - In order to integrate the Single Sign-On Agent with the Single Sign-On Gateway a JAAS Tomcat Realm entry must be added to the `server.xml`
- Configure a JAAS Login Module
  - `jaas.conf` file in the `$CATALINA_HOME/conf` directory with the following content:

```
josso {
   org.josso.tc50.agent.jaas.SSOGatewayLoginModule
   required debug=true;
};
```

  - The Login Module validates the session and obtains the corresponding user and role information by invoking the gateway identity management web services

JOSSO

JOSSO Configuration—Integration with Tomcat or JBoss

- Configure the Agent

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<agent>
    <class>org.josso.tc50.agent.CatalinaSSOAgent</class>
    <gatewayLoginUrl>http://localhost:8080/josso/signon/login.do</gatewayLoginUrl>
    <gatewayLogoutUrl>http://localhost:8080/josso/signon/logout.do</gatewayLogoutUrl>
    <sessionAccessMinInterval>1000</sessionAccessMinInterval>
    <service-locator>
            <class>org.josso.gateway.WebserviceGatewayServiceLocator</class>
            <endpoint>localhost:8080</endpoint>
    </service-locator>
    <partner-apps>
            <partner-app>
                <context>/partner</context>
            </partner-app>
    </partner-apps>
</agent>
```

JOSSO

JOSSO Configuration—Integration Java application with JOSSO

- Web application Security Constraints
- Configured using three elements in web.xml
  - <login-config> element
  - <security-constraint> element
  - <security-role> element

JOSSO

JOSSO Configuration—Integration Java application with JOSSO

- Integrating Enterprise JavaBeans™ (EJB™) with JOSSO
  - The security constraints should be declared in the `ejb-jar.xml` file of the partner components based on the Enterprise JavaBeans specification (EJB components)
  - For the user identity to be propagated to the EJB components tier, the `jboss.xml` file must set `java:/jaas/josso` as the security domain in the following way:

```
<?xml version="1.0" encoding="UTF-8"?>
<jboss>
   <security-domain>java:/jaas/josso</security-domain>
   <enterprise-beans>
      <session>
         <ejb-name>PartnerComponentEJB</ejb-name>
         <jndi-name>josso/samples/PartnerComponentEJB</jndi-name>
      </session>
   </enterprise-beans>
</jboss>
```

JA-SIG CAS

Central Authentication Service

- An open and well-documented protocol
- A library of clients for Java technology, .NET, PHP, Perl, Apache, uPortal and others
- Integrates with uPortal, BlueSocket, TikiWiki, Mule, Liferay, Moodle, and others
- Community documentation and implementation support
- An extensive community of adopters

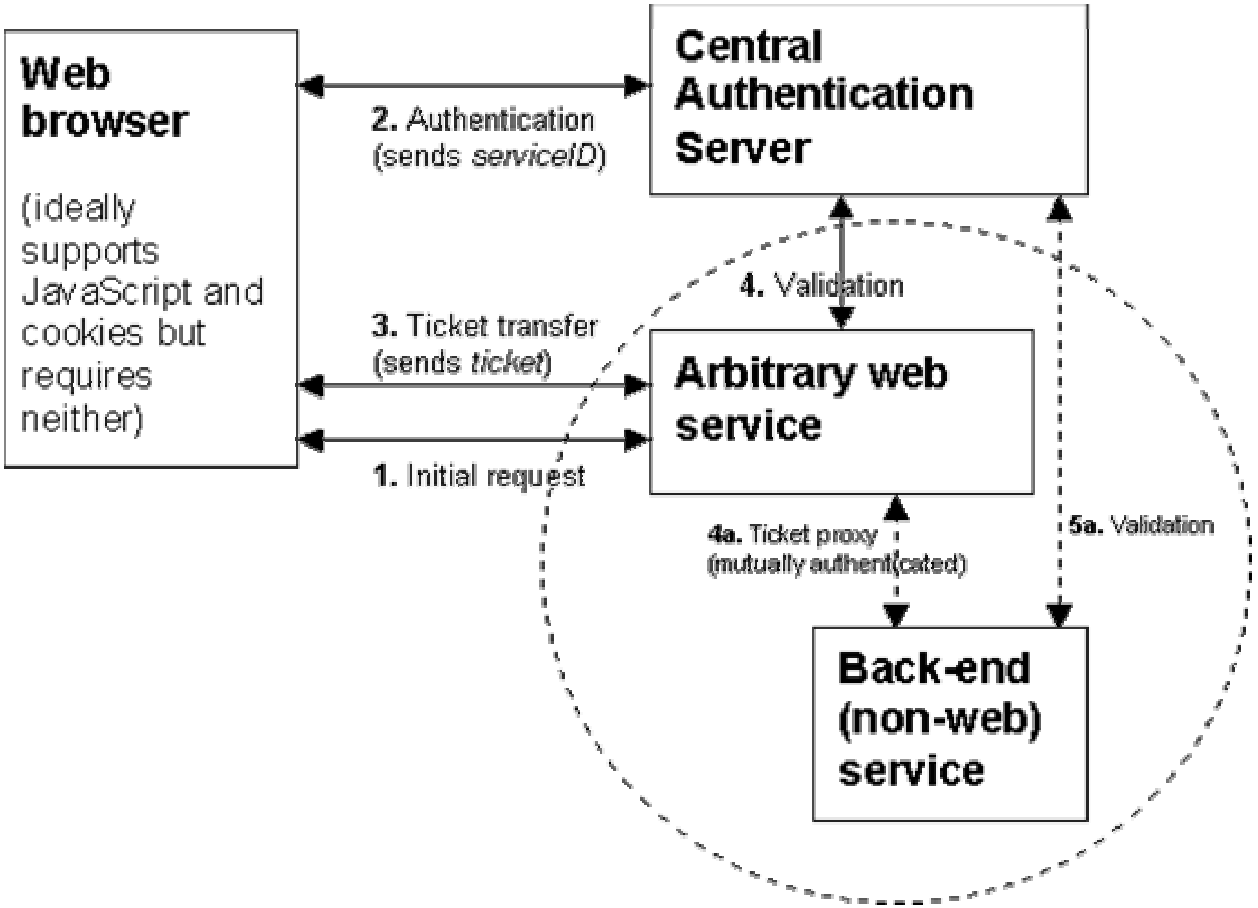
JA-SIG CAS

Continued…

- The players involved
  - CAS (The Central Authentication Service)
  - Service
  - Proxy
  - Target (or back-end service)
- CAS authentication make use of tickets, or opaque strings that prove some assertion to CAS
- CAS 2.0 uses the following tickets:
  - Ticket-granting cookie (TGC)
  - Service ticket (ST)
  - Proxy-granting ticket (PGT)
  - Proxy-granting ticket IOU (PGTIOU)
  - Proxy ticket (PT)

JA-SIG CAS

CAS Architecture



JA-SIG CAS

CAS Configuration

- Server Deployment
- Client Configuration

JA-SIG CAS

CAS Configuration—Server Deployment

- Based on authentication scheme used
  - Password based
  - Certificate based
- Need to implement Authentication Handler interface

JA-SIG CAS

CAS Configuration—Server Deployment

- Example: password based

```
public class UsernameLengthAuthnHandler implements AuthenticationHandler {

    public boolean authenticate(Credentials credentials) throws
            AuthenticationException {
        UsernamePasswordCredentials upCredentials =
            (UsernamePasswordCredentials) credentials;
        String username = upCredentials.getUsername();
        String password = upCredentials.getPassword();
        String correctPassword = Integer.toString(username.length());
        return correctPassword.equals(password);
    }

    public boolean supports(Credentials credentials) {
        // we support credentials that bear usernames and passwords
        return credentials instanceof UsernamePasswordCredentials;
    }
}
```

JA-SIG CAS

CAS Configuration—Server Deployment

- Customizing views
  - The existing views can be changed (i.e., JSP pages to match the look and feel of the applications)
- Using LDAP for authentication
  - Install the CAS LDAP authentication handler .jar file - `cas-server-ldap-{SOMETHING}.jar`
  - Include an LDAP library (“LdapTemplate” or “Spring LDAP”) into CAS server

JA-SIG CAS

CAS Configuration—Server Deployment

- Using X.509Certificates
  - CAS provides customizations to the CAS webflow to retrieve certificates from the HttpServletRequest, package the certificates into Credentials CAS can understand and pass them into the CentralAuthenticationService service
  - Provides an authentication handler to determine the validity of a certificate and if the credentials are authentic or not
  - Provides sample resolvers to translate the credentials into a principal that client applications will understand

JA-SIG CAS

CAS Configuration—Client

- Various clients
  - Java technology client
  - JSP software client
  - Uportal client
  - Acegi as CAS client
  - Perl, ASP.NET client, etc.

JA-SIG CAS

CAS Configuration—Client

- Java technology Client Configuration
- CASFilter configuration—Example

```
<web-app>
...
    <filter>
        <filter-name>CAS Filter</filter-name>
        <filter-class>edu.yale.its.tp.cas.client.filter.CASFilter</filter-class>
        <init-param>
            <param-name>edu.yale.its.tp.cas.client.filter.loginUrl</param-name>
            <param-value>https://secure.its.yale.edu/cas/login</param-value>
        </init-param>
        <init-param>
            <param-name>edu.yale.its.tp.cas.client.filter.validateUrl</param-name>
            <param-value>https://secure.its.yale.edu/cas/serviceValidate</param-
value>
        </init-param>
        <init-param>
            <param-name>edu.yale.its.tp.cas.client.filter.serverName</param-name>
            <param-value>your server name and port (e.g., www.yale.edu:8080)</param-
value>
        </init-param>
    </filter>

    <filter-mapping>
        <filter-name>CAS Filter</filter-name>
        <url-pattern>/requires-cas-authetication/*</url-pattern>
    </filter-mapping>
    ...
</web-app>
```

Agenda

What Is SSO?

Survey of the Main Open Source Players

Head-to-Head Comparisons

Summary

Q&A

Head-to-Head Comparison

Retrofitting an existing application

- JOSSO
  - No support for certain application servers
  - Does provide a plugin infrastructure to facilitate integration with other containers; you can base your own plugin on existing samples
- OpenSSO
  - Can fit into a multitude of application servers because of the availability of agents
  - These agents include Apache, Sun Java System Web Server, Microsoft IIS, Domino

Head-to-Head Comparison

Integration of non-Java applications

- JOSSO
  - Uses web services for asserting user identity via SOAP
  - Allows the integration of non-Java applications (e.g., PHP, .NET, etc.)
- CAS
  - There are many client libraries to assist in “CASifying” applications
  - Examples include AuthCAS for Apache, a uPortal client, a Java technology Client, a PHP client, and a Perl client

Head-to-Head Comparison

Customizability

- JOSSO
  - If your application server is not supported, need to customize by writing plugins
- CAS
  - Basic implementation includes only HTTPS
  - Can be easily customized to be HTTP enabled
  - Look and feel of login pages can be changed
  - Comes with pluggable authenticators to validate against LDAP, etc.
- OpenSSO
  - Customizations can be done by writing Authentication modules
  - Authentication User Interface JSP pages can be customized by Realm, Locale, Client type, or any Service of the SSO system

Head-to-Head Comparison

Ease of deployment

- CAS
  - Involves deploying CAS Server (downloadable as a pre-built WAR file or can be customized) and a CAS client with each application
- JOSSO
  - Involves Configuration of:
    - Single Sign-On Gateway
    - The Authenticator
    - The Identity Manager
    - The Session Manager
- OpenSSO
  - Deployable as a WAR file

Head-to-Head Comparison

Authentication for non-browser-based clients

- CAS
  - Has Proxy Authentication support
- OpenSSO
  - Does not have out-of-the-box support for CAS-like proxy authentication; however, there are authentication APIs available to build one
- JOSSO
  - Comes with a Reverse Proxy component that can be used to create n-tier Single Sign-On configurations

Head-to-Head Comparison

Support for web service security

- JOSSO
  - Can be used to secure web services but is limited due to the level of application server support
- CAS
  - Supports web service security by protecting URLs
- OpenSSO
  - Secure web services using SAML

Head-to-Head Comparison

Community support

- As all three are Open Source solutions, the support is in the form of project websites, community generated documentation, user forums and mailing lists
- CAS, OpenSSO, and JOSSO all have well-managed user groups

Agenda

What Is SSO?

Survey of the Main Open Source Players

Head-to-Head Comparisons

Summary

Q&A

How to Choose

Which horse for which course…

- There are multiple factors to consider when deciding on the SSO solution you need
- All three are Open Source solutions, so licensing issues are removed
- OpenSSO is a good choice if:
  - Using XML-based file formats and language independent APIs is important
  - Clustered environment support is required
  - SSL mutual authentication is required
  - You want to leverage all of the features of the Sun Java System Access Manager
- CAS is a good choice if:
  - Your using a Spring-based infrastructure with acegi
  - Your using simple DB-based credential management
- JOSSO is a good choice if:
  - It supports your particular application server; otherwise, additional development effort will be required

Alternative Open Source Solutions

Some other horses to consider

- Atlassian Seraph
  http://opensource.atlassian.com/seraph
- Shibboleth
  http://shibboleth.internet2.edu
- CoSign
  http://www.umich.edu/~umweb/software/cosign
- Enterprise Sign On Engine
  http://esoeproject.org

For More Information

- OpenSSO Home Page
  https://opensso.dev.java.net/
- JOSSO Home Page
  http://www.josso.org/
- CAS Home Page
  http://www.ja-sig.org/products/cas/
- Wikipedia
  http://en.wikipedia.org/wiki/Single_sign-on
- SAML
  http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security
- Acegi
  http://www.acegisecurity.org/

Agenda

What Is SSO?

Survey of the Main Open Source Players

Head-to-Head Comparisons

Summary

Q&A

Q&A

Craig Dickson—cdickson@behr.com
Naveen Nallannagari—nnallannagari@behr.com

Fast and Free SSO: A Survey of Open-Source Solutions to Single Sign-On

Craig Dickson, Software Engineering Manager
Naveen Nallannagari, Senior Consultant

Behr Process Corporation
www.behr.com

TS-4604