IBM

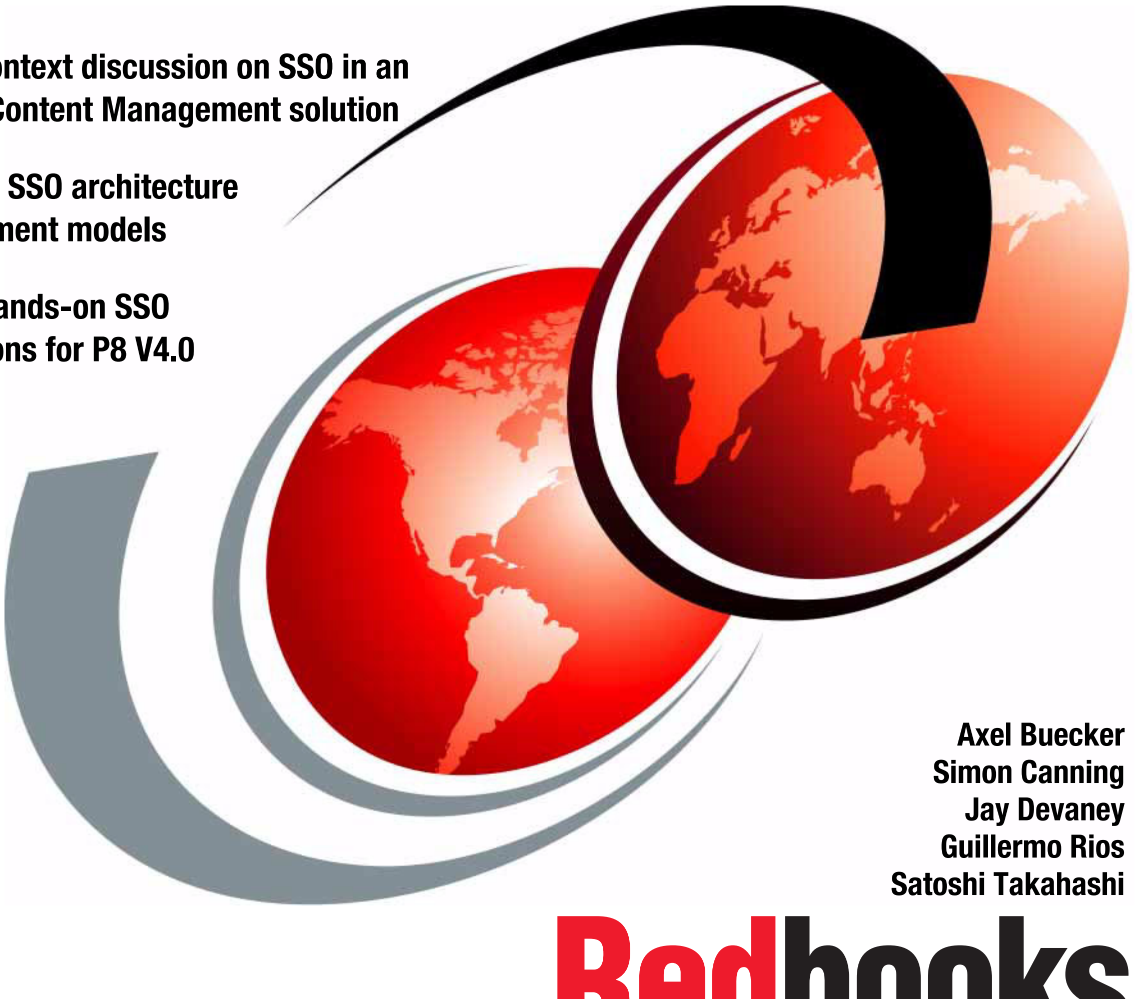
Single Sign-On Solutions for IBM FileNet P8

Using IBM Tivoli and WebSphere Security Technology

Business context discussion on SSO in an Enterprise Content Management solution

Overview of SSO architecture and deployment models

Complete hands-on SSO configurations for P8 V4.0



Axel Buecker
Simon Canning
Jay Devaney
Guillermo Rios
Satoshi Takahashi

Redbooks

ibm.com/redbooks

International Technical Support Organization

Single Sign-On Solutions for IBM FileNet P8 Using IBM Tivoli and WebSphere Security Technology

June 2009

SG24-7675-00

Note: Before using this information and the product it supports, read the information in "Notices" on page vii.

First Edition (June 2009)

This edition applies to Version 4.0 of IBM FileNet P8. We point out the individual versions of the different products used in the single sign-on scenarios in their respective chapters.

Contents

Notices

Trademarks

Preface

Authentication is the act of verifying a user's identity based on the credentials that they have presented. Establishing each user's identity is a critical first step in any client/server based system. In this IBM® Redbooks® publication, we present an overview of the set of authentication options in the IBM FileNet® P8 V4.0 release.

The two standards at the core of the authentication process in IBM FileNet P8 V4.0 are the Java™ Authentication and Authorization Service (JAAS) standard and the Web Services Security standard (WS-Security). The JAAS standard forms the framework for security interoperability in the J2EE™ world, while the WS-Security standard forms the framework for security interoperability in the heterogeneous world of clients and servers that communicate through Web services interfaces. IBM FileNet customers rely on a variety of authentication technologies to secure their corporate intranets. By implementing and adhering to these two standards, IBM FileNet P8 V4.0 enables a wide range of authentication integrations.

In this IBM Redbooks publication we discuss and demonstrate the IBM FileNet P8 integration with IBM Tivoli Access Manager for e-business, IBM Tivoli Federated Identity Manager, and the SPNEGO mechanism provided in IBM WebSphere Application Server.

This book is a valuable resource for security officers, access management administrators, and architects who wish to better understand single sign-on options for the IBM FileNet P8 V4.0 solution.

A word of caution

Performing a single sign-on (SSO) integration with IBM FileNet P8 V4.x using IBM Tivoli® Access Manager and a supported J2EE application server can be a very complex undertaking. The Tivoli Access Manager product supports many different types of SSO scenarios. This product also comes with many pages of product documentation. If you are planning to configure P8 with this SSO solution, you need to either have the appropriate SSO expertise on staff, or engage a services group to assist with the integration. The IBM FileNet ECM team does not have the in-depth knowledge of these complex products, and are not in a position to provide support for them.

In general, the following are the guidelines for identifying the appropriate resources in order to plan and perform a successful SSO integration with IBM FileNet P8 V4.x:

- Configuring an SSO solution (Tivoli Access Manager) to work with the J2EE application servers (WebSphere® and WebLogic) that are hosting the P8 Content Engine (CE) and Application Engine (AE).

  This task requires in-depth knowledge of the SSO solution and the application server environment. Most likely, a combination of your IT resources, a Tivoli Access Manager administrator, and an application server administrator needs to be involved in this effort.

- Configuring the reverse proxy server (WebSEAL) to work with the application server hosting the P8 AE

  HTTP requests coming into a reverse proxy server are not a simple pass through. The behavior of WorkPlace and other AE applications can vary when requests and responses are being translated by a reverse proxy server. Reverse proxy servers can be quite difficult to configure, particularly on the AE tier. A combination of resources from the customer, Tivoli Access Manager/WebSEAL administrator, and an application server administrator needs to be involved in this effort.

- Setting up the appropriate trust between the AE and CE servers

  This is required in order for the authenticated JAAS subject to be propagated to P8 CE. A combination of the IBM FileNet TC/personnel and an application server administrator needs to be involved in this effort.

- Enabling SSO for custom J2EE applications using the P8 CE Java APIs

  These custom applications need to perform a JAAS login and establish trust to the P8 CE server in order for the authenticated JAAS subject to be propagated to CE. A combination of your resources and an application server administrator needs to be involved in this effort.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

Axel Buecker is a Certified Consulting Software IT Specialist at the International Technical Support Organization (ITSO), Austin Center. He writes extensively and teaches IBM classes worldwide on areas of software security architecture and network computing technologies. He holds a degree in Computer Science from the University of Bremen, Germany. He has 22 years of experience in a variety of areas related to workstation and systems management, network computing, and e-business solutions. Before joining the ITSO in March 2000, Axel worked for IBM in Germany as a Senior IT Specialist in Software Security Architecture.

Simon Canning is a Software Engineer in the Australia Development Laboratory, Gold Coast location. He has 2 years of experience in the Tivoli security field. He holds a degree in Engineering (Computer Systems) from the Queensland University of Technology. His areas of expertise include Web-based single sign-on and Web services security. He has written Tivoli security related articles on IBM developerWorks®.

Jay Devaney is a Solution Architect on the Worldwide ECM team. He has 9 years of experience in the ECM field. Jay joined IBM with the FileNet acquisition and has worked at IBM for 2 years. His areas of expertise include content management, business process management, and software security. Jay holds a Bachelor of Science degree from Pepperdine University. He has also done post-graduate study in Computer Science.

Guillermo Rios is a Solution Architect on the Americas ECM team, based in Costa Mesa, California. He has 15 years of experience on the FileNet ECM platform. He holds a degree in Computer Science from Universidad Anahuac, Mexico. Before joining the Architects team in November 2006, Guillermo worked for FileNet in the Latin America team for 8 years as a Senior IT Specialist helping customers and partners in the design and implementation of complex FileNet solutions for the banking and insurance industry. He has written several P8 platform technical notes on single-sign on.

Satoshi Takahashi is a senior IT specialist in IBM Japan. He has 10 years of experience in the IT security field. He is mainly working on consultation and technical support for Tivoli Security products, such as Access Manager for e-business, TDS, and TDI. Before joined the technical support team, he worked with health care industry customers for 5 years. He wrote an IBM network software related IBM Redbooks publication in 1999.

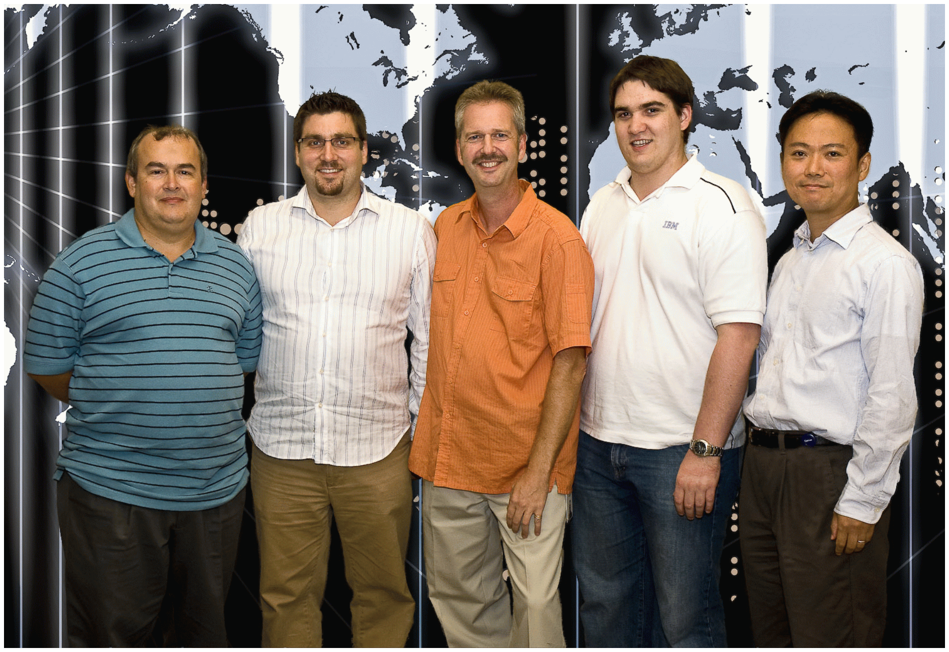


Figure 1 From left: Guillermo, Jay, Axel, Simon, and Satoshi

Thanks to the following people for their contributions to this project:

Wade Wallace
International Technical Support Organization, Austin Center

Roger Vuong, Joseph Raby, Darik Siegfried, Jackie Zhu, Evangeline Fink, Neil Readshaw, Davin Holmes, Shane Weeden, Pat Wardrop, Xingdong Ji, Yuan-Hsin Chen, Jingli Chen, Matthew Vest
IBM US

Become a published author

Join us for a two- to six-week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

`ibm.com/redbooks/residencies.html`

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  `ibm.com/redbooks`

- Send your comments in an e-mail to:

  `redbooks@us.ibm.com`

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Part 1

Architecture and design

In this part, we discuss the business context of the IBM FileNet P8 single sign-on solutions. We then describe how to technically architect different solutions into an existing environment and introduce the necessary components.

1

Business context for single sign-on in an Enterprise Content Management environment

In this chapter, we introduce single sign-on and discuss the benefits of integrating single sign-on with an Enterprise Content Management (ECM) solution. We approach this both from a general security perspective and an ECM standpoint. Security within an ECM environment is particularly important because of the sensitive nature of the content managed within ECM solutions.

1.1 What single sign-on is

Today, in many organizations, there is an abundance of applications that require authentication and the number of applications is increasing. Users use a number of different user IDs and passwords to log in to applications. If there are too many passwords, people are often unable to keep track of all of them. This can generate unnecessary service calls to the help desk. A single sign-on solution can alleviate this problem both for the user and the IT organization. With a single sign-on solution, users only need to log in once and then they can use all applications without having to provide user credentials again. This is the basic function of single sign-on (SSO).

In our view, there are three classes of single sign-on solutions.

- Web single-sign on
- Desktop single sign-on
- Federated single sign-on

1.1.1 Three classes of single sign-on

Let us consider SSO in the context of the evolution of computing models, as shown in Figure 1-1.

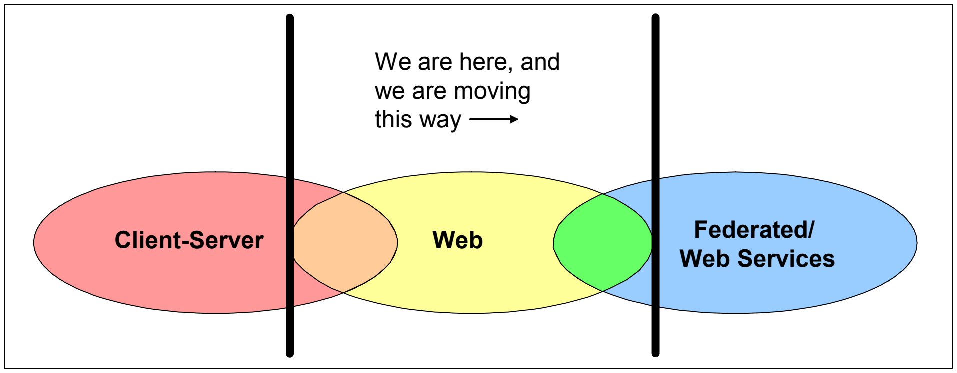


Figure 1-1 Evolution of computing models

As the computing models evolve from the insular client-server model to the open Web services model, the importance of security increases dramatically. In Table 1-1, we demonstrate how this evolution is impacting user authentication.

Table 1-1 Single sign-on models and their authentication characteristics

Model	Network exposure	Communication protocol	Authentication
Client-Server	Private	Proprietary	Authenticates user against an application-specific repository. User population contained within the enterprise.
Web	Private/Public	Standards-based	Authenticates user against an application-specific or enterprise repository. User population expands to the Internet.
Federated/ Web Services	Public	Standards-based	Authenticates user against an enterprise repository. Also authenticates users and other network entities (for example, Web services) originating from foreign companies. User population expands to the Internet and other enterprises.

The *client-server model* achieves a certain level of security because it operates within the corporate network and communicates over a propriety protocol. However, being a network-based architecture, the client-server model requires client authentication. In this model, each application tends to have its own user repository. This requires users to keep track of separate accounts for separate applications.

The *Web model* is just a special case of the client-server model that uses a standard client and communications protocol (HTTP/S). Companies find this model more cost effective, since only one client needs to be deployed to the corporate desktops. Many traditional Web applications were developed (such as the client-server model) using their own user repository. In addition to having the same sign-on problem as the client-server model, the Web model compounds the problem by exposing corporate applications directly to the customers through the Internet. This means that companies face a large increase in the number of users that must be supported. Also, these users are not the traditional corporate users, but rather customers who the company must vet before assigning accounts.

The emergence of the Web as the platform of choice for corporate applications and the exposure of the corporate applications to the users created the opportunity for services to be linked between corporations over the Internet. For example, a corporate Web portal may link to the health benefits provider and the financial services partner. These links lead to an external Web site that requires authentication. Thus, the user is once again faced with additional account data to manage. The *federated model* requires identity information to be carried securely over the Internet so that users may consume services at various companies.

1.2 Benefits of a single sign-on solution

In this section, we discuss some of the most prevalent benefits of single sign-on solutions from a security perspective.

- Cost reduction for the user help desk

  If users are required to maintain different accounts for each application they access, it can quickly become difficult for them to keep track of passwords. Not remembering passwords can result in system lockouts, lost productivity, and it can generate unnecessary help desk calls. It is well known that a high percentage of help desk cost is related to password reset service requests.

- Greater user productivity and experience

  SSO allows users to access business systems faster, which enables them to get more done. And users who can sign in once feel better about their transaction experience than users who must log in multiple times with many different IDs and passwords.

- Mitigate the risk around password handling

  Users that are required to keep track of numerous passwords tend to write down the passwords, which can create an additional security exposure. In the worst case, users have been known to write their passwords on post-it notes and paste them to their displays. This situation can cause a critical information breach. To mitigate this security risk, you need to reduce the number of passwords that a user is required to remember.

- Centralized access control and centralized audit log management

  Well implemented single sign-on solutions provide centralized access control and logging functions. These functions are useful to the overall organization for the purposes of adhering to regulatory compliance initiatives. Without these functions, more workload is required to maintain the appropriate levels of security and compliance control.

- Keep confidentiality and integrity on the communication channel between client and server

  This benefit only applies to the Web single sign-on solution. Typical Web single sign-on products use a reverse proxy that is placed between the browser client and the Web application server. Even if the Web application server has no SSL capability, a reverse proxy component can provide SSL capability between the client and the reverse proxy. This function provides secure communication between the client and the data center.

- Faster application deployment

  When organizations deploy a superior SSO and security system that allows application developers to call out to external security services, security no longer has to be coded into each application. As a result, a company can get new applications to market quickly, and can later update application business logic and enhance security much more efficiently.

1.3 Single sign-on and Enterprise Content Management

In this section, we give an overview of ECM and security at the business level. In addition, we talk about how SSO integrates into an ECM solution and give a high level overview of how authentication and authorization take place within the IBM FileNet P8 platform. Further detailed descriptions about the FileNet P8 architecture and integration with LDAP and SSO solutions is provided in subsequent chapters.

1.3.1 The ECM point of view

Enterprise Content Management is a broad area that takes in a number of technology spaces, such as content storage, business process management, and general search, and so on. For the purposes of this book, we discuss the IBM FileNet P8 Platform for Enterprise Content Management. The P8 platform delivers a wide array of services and features through our single Enterprise Reference Architecture (see Figure 1-2).

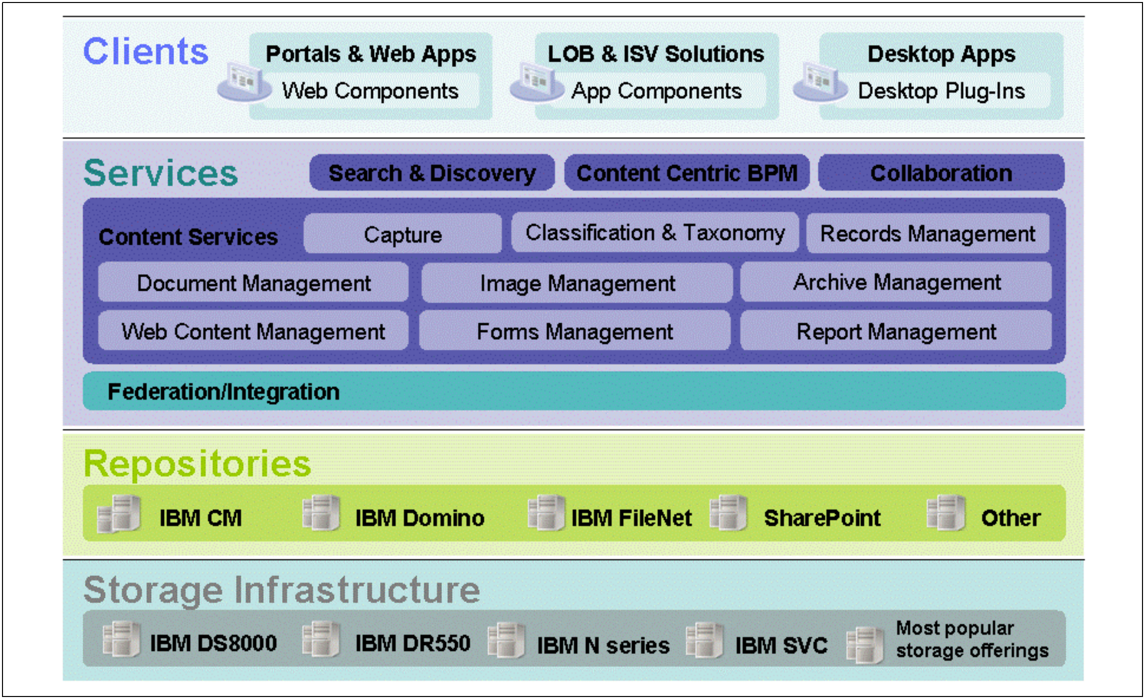


Figure 1-2 ECM Enterprise Reference Architecture

In Figure 1-2 we show the architecture of IBM ECM solutions at a very high level, but each of these elements are critical for Enterprise Content Management.

- Clients

  In the first layer of the diagram, we present the client interfaces. Client interfaces are important to knowledge workers; good, clean, and efficient interfaces are critical for user adoption and increased productivity in the ECM domain. Building upon Web 2.0 technologies and look and feel, we continue to modernize and bring a highly usable and efficient experience to users.

- Services

  In the middle layer are the set of services that are delivered by the ECM platform. These are the core capabilities that manage all information types, including images, reports, documents, e-mail, Web content, forms, audio, video, XML, and so on.

  – First, we need to connect to all the repositories where information resides. To accomplish this task, we utilize information integration and content federation services. This is more than just a connection to one content repository or another; it provides active management of content *in place*, so customers do not have to migrate or move content. This helps customers retain, find, use, reuse, and manage information, regardless of where it resides. And with IBM Information On Demand (IOD), this set of technologies, products, and services helps bring together information from other sources across the enterprise, whether it is traditional applications, SAP®, Siebel®, or PeopleSoft® applications. IOD helps bring this information together and put it in context so it returns greater value to the enterprise. ECM fits right into the IBM IOD strategy.

  – Second, we add a complete set of information management capabilities to bring life to the content, from capture and management to presentation and archival, and everything in between. Some of the exciting things we are working on include taxonomy and classification of content. Adding these capabilities help organizations more efficiently and effectively create, categorize, organize, manage and find information, and it allows us to integrate more completely with other parts of IBM for enterprise search, content analytics and business intelligence, information servers, and master meta data management.

  – Third, collaboration brings people together to create, review, and use information. We are constantly enhancing our content services to provide new ways for extended and dispersed teams to collaborate and come together in new and efficient ways, whether it is through SharePoint®, Quickr™, or other collaboration solutions. Content is good, but we need to put it to work. Business Process Management (BPM) remains very central to our ECM strategy to enable you to build content centric solutions and to execute content centric processes to bring value to your users.

- Repositories

  At the bottom of this chart, you see a number of different repositories, or places where content and data are stored. Information can be stored anywhere: in IBM repositories, in non-IBM repositories, on file shares and e-mail systems, in databases, and in storage devices. A typical company may have more than 10 different content management systems and repositories, with the common result that users do not know about content in other parts of the enterprise that might be useful, or they cannot access it. ECM aims to connect all of these sources and to bring all of the information and data together in an open manner so that the appropriate user can get access to it as needed to perform their functions within the organization.

As you can see, Enterprise Content Management covers a broad range of application spaces, such as BPM, document management, imaging, records management and compliance, team collaboration, forms management, e-mail management, and more. The point here being that the introduction of an ECM platform such as FileNet P8 is not simply the introduction of a new application to your user environment; it can mean the introduction of a number of new applications. Some of those applications may or may not be visible to the user. Additionally, ECM capabilities may be delivered through integration with other Enterprise Systems, such as SAP, Siebel, or portals that provide the user interface, such as when the ECM serves as the content repository for documents associated with records within these environments. One of the benefits of the P8 platform is the complete and seamless integration of its components. This manifests itself in the security space in its reliance on an LDAP provider to store and maintain user names and passwords. At a high level, authentication, which is covered in detail in 1.3.2, “P8 authentication overview” on page 10, is handled by validating that the credentials received from the user, or the application accessing ECM services, is found in one of the Directory Servers to which P8 has access. Regardless of the number of ECM applications to which the user has access or services accessed by Line of Business applications he employs, the user has a single identity across the platform, and by leveraging an SSO solution we can then seamlessly extend that to provide the user a single identity across the organization. With Federated Identity, we show how that can be extended beyond organizational boundaries.

1.3.2 P8 authentication overview

Authentication is the process of confirming that a user accessing an application is who they purport to be. This is done by validating their credentials, in our particular case, by challenging them for a user name and password. On the IBM FileNet P8 platform, the users’ identities are defined and maintained in a user repository (LDAP directory). Their ability to access objects and services on the ECM platform is determined by their individual identity and their group memberships, which is contained in the topic of authorization.

The two standards at the core of the authentication process in IBM FileNet P8 V4.0 are the *Java Authentication and Authorization Service* (JAAS) standard and the *Web Services Security* standard (WS-Security).

The JAAS standard forms the framework for security interoperability in the J2EE world, while the WS-Security standard forms the framework for security interoperability in the heterogeneous world of clients and servers that communicate through Web services interfaces.

IBM FileNet P8 customers rely on a wide variety of authentication technologies to secure their corporate intranets. By implementing and adhering to these two standards, IBM FileNet P8 enables a wide range of authentication integrations.

1.3.3 P8 authorization overview

Authorization is the process of determining the access rights that an authenticated user has for a particular application. In the case of our ECM application, this may include access to data, documents, forms, and other formats of electronic content. Authorizations also grant permissions to perform certain actions within an application, such as approve a new loan or insurance policy in a BPM managed process.

IBM FileNet P8 implements and manages authorization within the platform through the use of *access control lists* (ACLs). An access control list defines the rights (or permissions) a user or group has to a resource or object. Those rights can typically only be changed by modifying the ACL defined on each individual resource or object. Each object stored within the IBM P8 Content Engine has an associated ACL attached that determines the users and groups that have permissions to access the information it contains or perform actions on the objects. These permissions may range from a user only having the ability to view the information contained in the object to the user having full access to the object, which may include the ability to update the information or delete the object in its entirety.

1.3.4 SOA identity propagation overview

A *service-oriented architecture* (SOA) connects loosely coupled services to construct new applications. These services use their own user registries, which are often administered in isolation from those of other services in the SOA environment. Users and service entities in such a homogeneous environment are likely to have different identities in the various services that make up a composite application, as shown in Figure 1-3.

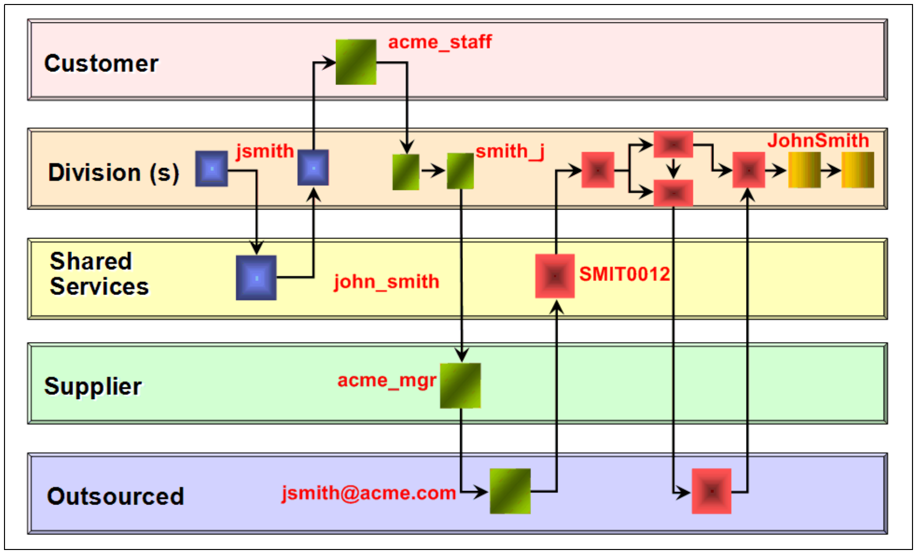


Figure 1-3 Different identities required in different services of a composite application

Establishing the identity of the service requester in each service request is a fundamental step in ensuring that business requirements such as authorization, audit, and compliance can be implemented.

Identity services are required in the SOA infrastructure so that services can be easily interconnected with the correct identities being propagated.

A solution for the challenge of SOA identity propagation must be:

- Capable of understanding and operating with a variety of formats for representing identity
- Capable of translating between different identities
- Based on SOA principles itself to deliver a flexible, infrastructure-based solution de-coupled from application business logic
- Constructed using open standards to provide maximum interoperability with the platforms and systems on which SOA solutions are constructed

In the ECM context, SOA identity propagation is one method to tackle the problem of authenticating external users, such as a third-party appraiser that provides a report in the context of a mortgage origination process for your bank.

Traditionally, in ECM implementations, these external users are required to be stored and maintained in the corporate directory server, which creates an additional administrative burden and costs for IT that is often undesirable to the organization.

The SOA principal of identity propagation provides a mechanism by which users have their identities verified using an identity provider in a way that allows the partner's identity assertion to be trusted by your organization. This allows, for example, a group of individual appraisers in an organization external to your bank to be authenticated against their own corporate directory and participate in a business process to underwrite a mortgage loan.

In order for this scenario to take place, your bank must have a business agreement with this third-party firm to regulate the level of trust as well as the technical IT terms about who manages the user identities and how an authenticated user may access certain application and data resources on your IT infrastructure.

By using a SOA identity propagation approach, you can trust the identity assertion provided by your partner through a SAML token, for example, so that the individual can participate in tasks in your business processes.

If you want to learn more about the powerful world of service-oriented architectures, especially about SOA security, refer to the IBM Redbooks publications *Understanding SOA Security Design and Implementation*, SG24-7310 and *Propagating Identity in SOA with Tivoli Federated Identity Manager*, REDP-4354.

2

Single sign-on architecture and component design

In this chapter, we want to cover more details about the three single sign-on scenarios that we introduced in 1.1, “What single sign-on is” on page 4 and how these scenarios can be architecturally mapped onto four product related solutions.

For a better general understanding, we begin with a look at the overall IBM FileNet P8 architecture and its core components. This gives you an initial view of what a P8 implementation may look like in the absence of a SSO solution. In the rest of the chapter, we then continue to discuss the architectural overview of three product solutions and how they integrate with P8 to deliver SSO in an ECM implementation.

2.1 IBM FileNet P8 architecture overview

There are two primary components in the IBM FileNet P8 Platform: the *Content Engine* server, which provides core content management capabilities, and the *Process Engine* server, which provides core business process management (BPM) capabilities. A third component, the *Application Engine*, hosts the Workplace Web application, Workplace Java applets, and Application Programming Interfaces. For the purposes of security and SSO, we are going to focus more closely on the Content Engine because, for purposes of authentication, it provides the *Directory Service Provider layer* for the rest of the platform. This will be explained in detail in later sections of this chapter. Now we will give you an overview of the P8 platform to create a general understanding of all the components and how they interoperate.

2.1.1 Architecture of an IBM FileNet P8 system

IBM FileNet P8 is the foundation for the overall IBM FileNet family of products, allowing the individual solutions to seamlessly interoperate so that their powerful capabilities can be fully utilized. There are three main *engines* that comprise the IBM FileNet P8 platform. We define an engine as a collection of services and components, which together perform a set of related functions. Although each engine is comprised of various parts, we view it as a single functional unit. Understanding the complexity of how each engine works is not necessary, but it is important to know what each engine does. Let us shed some light into the primary engines of the P8 platform.

- Content Engine (CE)

  The Content Engine provides software services for managing different types of business-related content, which we refer to as *objects*. The Content Engine provides the active content capability, so that events involving content objects can trigger corresponding actions. The Content Engine handles database transactions required for managing one or more *object stores*. An object store is a repository for storing objects in a P8 environment. Each object store manages a database (Content Engine database) for meta data and one or more storage areas that represent *physical storage locations* for content. A storage area can be in a database, a file system, a fixed content device, such as an *image services repository*, Network Appliance™ SnapLock®, EMC Centera, IBM DR550, or a combination of these options. For the latest integration information, contact your local sales team. The Content Engine uses the latest J2EE technology standards and is deployed inside of an application server that spans a Java Virtual Machine.

- Process Engine (PE)

  The Process Engine allows you to create, modify, and manage automated business processes. It provides software services such as business process execution and routing, integration of external rules engines, process analysis, and process simulation. The processes can be performed by applications, enterprise users, or external users, such as partners and clients. The Process Engine uses the Process Engine database where all process related data is stored. Processes run inside of an isolated region that acts as an individual processing space.

- Application Engine (AE)

  The Application Engine hosts the Workplace Web application, Workplace Java applets, and Application Programming Interfaces (APIs). It is the presentation tier for both content and process. It also handles *user authentication* against the directory service. The Application Engine consists of an application server with one or more deployed applications. It includes a deployment of the JSP™ based Web client Workplace as well as the standard Web client Workplace XT, which is implemented with AJAX using Java Server Faces (JSF) technology. For simplification, we refer to these applications as *Workplace*. A third deployed application is the Global Help system. These applications run in the Java Virtual Machine context of the application server.

Both the Content Engine and Process Engine have their own databases. There is also a Global Configuration database (GCD) that stores a global system.

An overview of the P8 component architecture is shown in Figure 2-1. In the remainder of the P8 architecture overview section, we focus on the following components:

- “Content Engine architecture” on page 18
- “Process Engine architecture” on page 20
- “Directory service integration” on page 21
- “Java based client authentication through JAAS” on page 22
- “Web service based client authentication through WS-Security” on page 25
- “Application Engine authentication” on page 28
- “Process Engine authentication” on page 29

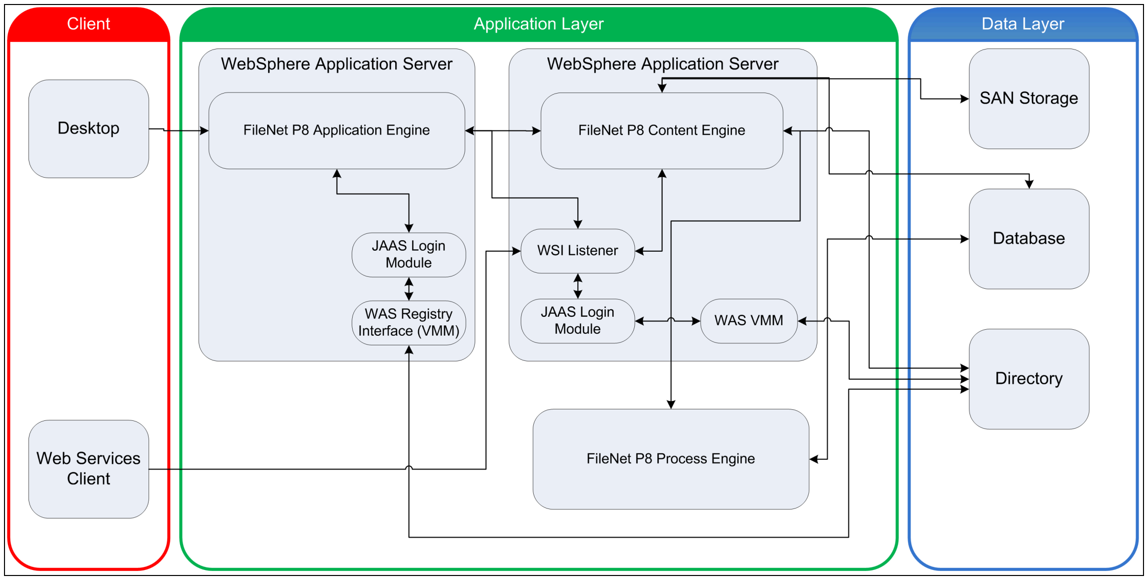

Client
Application Layer
Data Layer
Desktop
Web Services Client
WebSphere Application Server
FileNet P8 Application Engine
JAAS Login Module
WAS Registry Interface (VMM)
WebSphere Application Server
FileNet P8 Content Engine
WSI Listener
JAAS Login Module
WAS VMM
FileNet P8 Process Engine
SAN Storage
Database
Directory

Figure 2-1 IBM FileNet P8 architecture overview

2.1.2 Content Engine architecture

Figure 2-2 on page 20 shows a high level view of an IBM FileNet-P8 4.0 Content Engine server and some of the various types of client applications that access it. The Content Engine is packaged as a J2EE application and deployed on one or more J2EE application server instances.

The key components of this application are:

- The Content Engine Web service listener

  This component is packaged as a Servlet based application that resides in the Web container of the application server. This listener implements the *Content Engine Web service*. It supports IBM FileNet P8 V4.0 Web service clients, as well as IBM FileNet P8 3.5 Web service clients. It exposes the full functionality of the Content Engine server through a standard Web services API. Requests that arrive at this Web service are authenticated based on the credentials in their WS-Security headers, and then passed on to the Content Engine EJB™ layer.

- The Content Engine EJBs

  These J2EE session beans reside in the EJB tier of the application server, and implement the same set of low level Content Engine server primitives as the Content Engine Web service, exposing them through an Enterprise Java Bean interface, rather than a Web services interface. All clients of this EJB layer must perform a JAAS login prior to sending a request to one of the EJBs.

  The Content Engine EJBs are strictly for use within the CE Java API and are not exposed for general use.

The core content management logic resides in the resource adapter tier of the application server. It provides the same enterprise repository services, content storage services, and caching services, as the IBM FileNet P8 3.x Content Engine.

The Web Service listener and EJB layers are referred to as the two transport layers of the Content Engine. All client requests enter the Content Engine through one of these two transport layers.

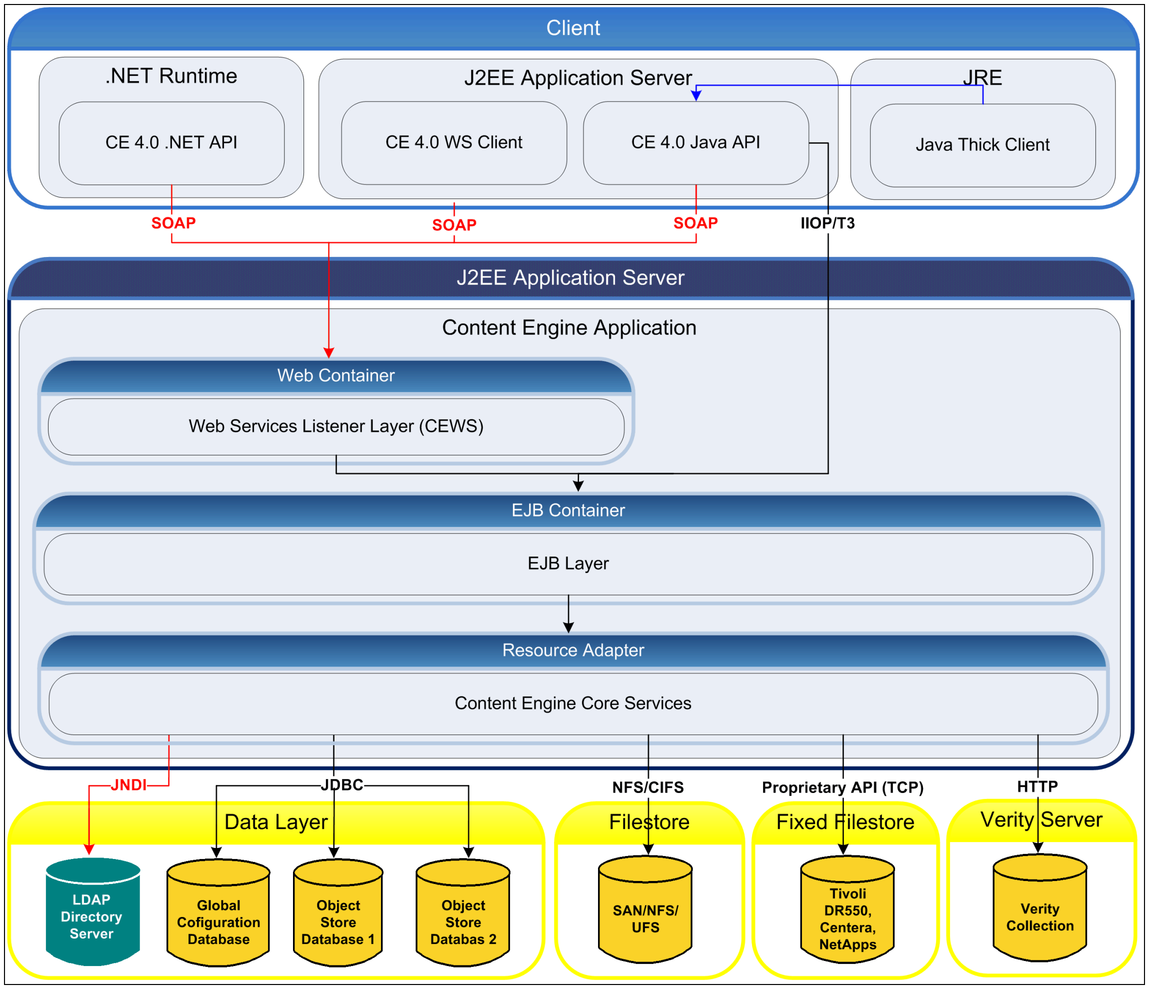


Figure 2-2 Content Engine component architecture

The IBM FileNet P8 V4.0 Content Engine accepts requests through a standard J2EE EJB, or through the Content Engine Web service. Authentication options will differ significantly, depending on which of these two transport layers an application uses.

2.1.3 Process Engine architecture

The IBM FileNet P8 Process Engine does not run on a J2EE application server in Version 4.0. Therefore, it is not able to directly leverage the JAAS standard to support authentication and single sign-on (SSO) solutions. The Process Engine works around this limitation in Version 4.0 by delegating authentication operations to the Content Engine server. This arrangement is discussed in further detail in 2.1.5, “Java based client authentication through JAAS” on page 22.

2.1.4 Directory service integration

It is important to note that the authentication process for IBM FileNet P8 V4.0 occurs before any Process Engine or Content Engine server logic is invoked. After authentication has occurred, the user's identity is represented by a *JAAS Subject* (or by an *Identity Token*, in the case of the Process Engine).

Authorization in IBM FileNet P8 uses a fine-grained model whereby individual documents, folders, process queue items, and so on, are protected by Access Control Lists (ACLs). An ACL contains a list of Access Control Entries (ACEs) that either grant or deny a certain type of access to an individual user or a group of users. In order to perform these authorization operations, a user must be mapped to an entry in a directory service, so that their security identifier may be obtained, and their group membership evaluated. This process of resolving a user to a directory service entry occurs by extracting a principal name from their JAAS Subject (or Identity Token) and issuing queries against the directory service to obtain user and group information. In IBM FileNet P8 V4.0, IBM FileNet software does not interact directly with a directory service for authentication purposes. Authentication is always performed by a JAAS LoginModule, which may or may not interact with a directory service. IBM FileNet software needs to retrieve user and group information from a directory service for authorization purposes.

As discussed in the previous section, the Process Engine delegates this user and group lookup to the Content Engine.

The Content Engine implements a *Directory Service Provider* layer. Directory Service Providers are implemented underneath this layer for each directory service supported by IBM FileNet P8. The directory services supported in Version 4.0 include:

- IBM Tivoli Directory Server
- Microsoft® Active Directory®
- Sun™ ONE Directory Server
- Sun Java™ System Directory
- Novell® eDirectory
- Microsoft ADAM (also known as ADLDS)

This list will likely be expanded in future P8 releases. Always refer to your product documentation for the latest accurate information of supported product versions.

2.1.5 Java based client authentication through JAAS

As shown in Figure 2-2 on page 20, clients of the Content Engine V4.0 Java API have the option of choosing to use either of the transport layers, *SOAP* or *EJB*. Clients of other Content Engine APIs do not have the option of using the EJB transport layer.

One of the advantages inherent to the EJB transport layer is the ability to leverage JAAS-based authentication. The JAAS standard provides a mechanism that allows third-party authentication / single sign-on solutions to be integrated with any J2EE compliant application. If a single sign-on provider writes a JAAS LoginModule for a given application server, then clients of applications hosted in that application server can leverage that SSO solution.

The way that JAAS-based authentication is used can differ in different client and server scenarios. The sections below illustrate some, but not all, such scenarios.

Browser-based client of J2EE application server

Browser-based clients of J2EE-based application servers interact with Servlets and Java Server Pages (JSPs). These are Java-based Web components, managed by the Servlet Container of a J2EE application server. Servlets interact with browser-based clients (and other Web clients) through a request/response paradigm managed by the Servlet Container. The authentication process for these types of clients is defined by the Java Servlet specification.

To access an IBM FileNet P8 server, J2EE Servlet-based applications must obtain a JAAS Subject that is valid in the J2EE EJB container that hosts the Content Engine EJB. There are two separate high level paradigms that may be used to obtain this JAAS Subject:

- Application managed authentication
- Container managed authentication

These two options are covered in more detail in the following sections.

Application managed authentication

A Servlet may make JAAS calls to perform its own JAAS login programmatically. This approach may involve application server specific idiosyncrasies and configuration issues, but it is fairly standardized for the common user name and password case. To use application managed authentication, each Servlet deployed in the container must include logic that determines if the user is authenticated. If the user is authenticated, then its identity is determined by examining information in the user's session. Each Servlet must perform some action (such as redirecting unauthenticated users to a logon page) that collects and verifies their credentials, handles errors, and manages the encoding of the authenticated identity into the user's session.

Container managed authentication

One of a standard set of Servlet authentication options may be specified in the Web application's deployment descriptor. In this case, the Servlet container performs the JAAS login, based on the credentials that are supplied, relieving the application of this burden.

Each Web application deployed in a Servlet container can specify one of the following options, as defined in the Servlet specification, which the container should use to authenticate users.

- HTTP Basic Authentication (BASIC)

  The container requests the client browser to prompt the user for a user name and password, using a browser specific dialog box. This option is defined by the HTTP specification. The user-supplied credentials are sent to the server for authentication. A secure transport must be used, as credentials are unencrypted. Few applications use this method, because the application’s look and feel are not preserved, and the login prompt cannot be integrated into the application in a cohesive presentation.

- HTTP Digest Authentication (DIGEST)

  Similar to HTTP Basic, but a digest (a one-way hash) of the password is sent instead of the password. This option is somewhat more secure, as the actual password is not compromised, but requires that the authentication mechanism that manages identities accept the password hash in a particular form, rather than the actual password. This is not possible in most real world cases.

- Forms Based Authentication (FORMS)

  Instead of asking the client browser to prompt for a user name and password, the caller is redirected to an application-specific form to provide this information. This option allows customization of the look and feel of the login page and any error pages. It also requires a secure transport to protect the password.

- HTTPS Client Authentication (CLIENT-CERT)

  This option requires each user to have a unique Public Key Certificate (PKC), and requires the use of an encrypted HTTPS (SSL) connection between the client and the server.

Note that while all four of the options above may be executed over an HTTPS connection (and, in fact, that is a recommended best practice), only CLIENT-CERT actually *requires* an HTTPS connection. SSL is engaged through the configuration in the Servlet descriptor of `<transport-guarantee>` as `CONFIDENTIAL` or `INTEGRAL`.

All of these technologies are forms of *container managed authentication*, where the J2EE Servlet Container performs the JAAS authentication based on credentials obtained by a standard mechanism. The specification of one of these authentication mechanisms is a standard part of a Servlet deployment descriptor. The specification and configuration of how the J2EE application server validates these credentials, however, is application server dependent. In an enterprise environment, an authentication mechanism must be provided to validate credentials against the enterprise identity management solution (either a directory service or a single sign-on solution).

Once a caller has been authenticated by a J2EE Servlet container, if the Servlet subsequently calls an EJB, the Servlet Container is required to propagate the caller's identity (JAAS Subject) to the EJB. Figure 2-3 shows the container managed authentication case, using forms-based authentication to authenticate the caller against an Active Directory service.

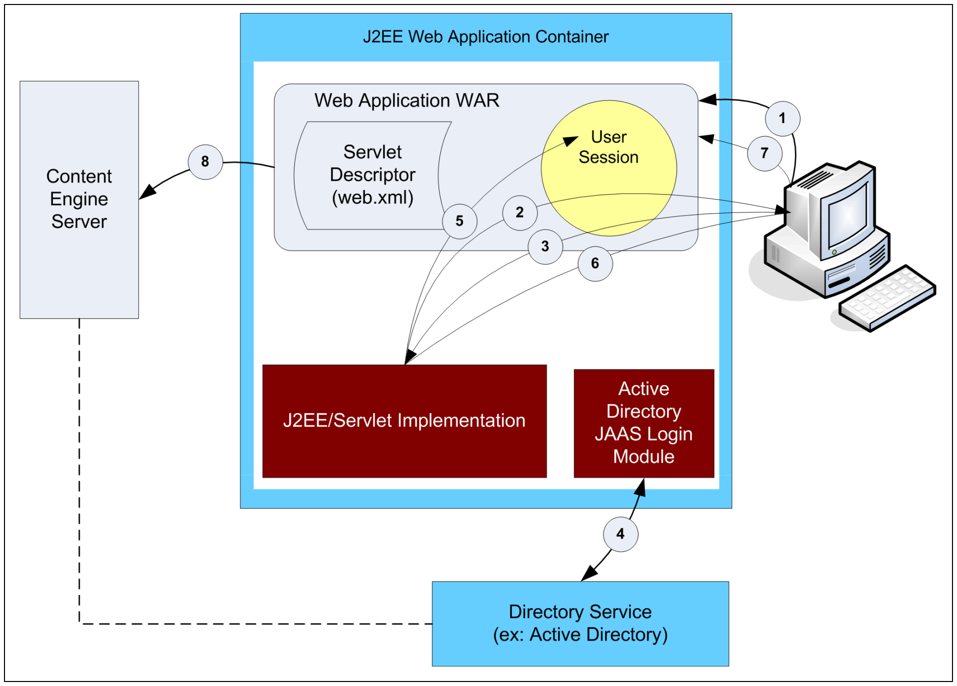

J2EE Web Application Container
Web Application WAR
Servlet Descriptor (web.xml)
User Session
Content Engine Server
J2EE/Servlet Implementation
Active Directory JAAS Login Module
Directory Service (ex: Active Directory)
1
2
3
4
5
6
7
8

Figure 2-3 JAAS container managed authentication

We also provide a step-by-step walkthrough for the authentication scenario.

1. A user attempts to access a Servlet based application.
2. The J2EE application server redirects the user to a page that challenges for credentials.
3. The user enters credentials and submits them to the server.
4. The J2EE server validates the user's credentials through JAAS.
5. The J2EE server creates JAAS Principal and Subject objects using the Active Directory JAAS LoginModule, and places them in the user's session.
6. The J2EE server redirects the user back to the application page that was originally requested.
7. The Servlet container looks for a user Principal available on the incoming request.
8. Once invoked, the Servlet makes a call to the P8 Content Engine server, and the user's JAAS Subject is propagated to the EJB container in which the Content Engine resides.

2.1.6 Web service based client authentication through WS-Security

In this section, we discuss how two specific WS-Security profiles (the *Username Token profile* and the *Kerberos profile*) are supported in IBM FileNet P8 V4.0, and how support for additional standard profiles, as well as nonstandard WS-Security compliant approaches, may be integrated with IBM FileNet P8 V4.0 Web services, using the IBM FileNet P8 V4.0 *Web Service Extensible Authentication Framework*.

Clients of a Web service must produce WS-Security compliant headers to use any of the P8 Web services. Most Web service-based applications are created using a toolkit, such as IBM Rational® Application Developer or Microsoft Visual Studio®, which handles the creation of WS-Security compliant headers.

Username Token credentials

The *Web Services Security Username Token Profile* specifies how user name and password based credentials can be passed in a WS-Security header. All Web service clients that adhere to this profile should be able to interact with any Web service that implements the profile. The XML `<wsse:UsernameToken>` and `<wsse:PasswordToken>` elements are defined, along with rules for how these fields must be used.

The key concept is, when Username Tokens are sent in a WS-Security header, a secure, private channel, such as an HTTPS connection, must be used between the client and the server, to prevent compromising the client's password.

Two types of passwords are defined by this standard: a text password and a password digest. IBM FileNet P8 V4.0 supports only the text password option. The digest version sends a SHA-1 digest of the password, instead of the text password. The digest version requires that the servers have access to the user's clear text password for verification purposes, which is typically not the case.

When a Web service request containing a Username Token arrives at an IBM FileNet P8 V4.0 Content Engine Web service, the Web service listener extracts the credentials from the WS-Security header, and uses them to perform a JAAS login using an application server specific user name and password LoginModule. Once the JAAS login has successfully completed, the IBM FileNet P8 Web service listener is now in possession of a JAAS Subject, and can pass the call along to the Content Engine server through the EJB transport.

Kerberos credentials

The *Web Services Security Kerberos Token Profile* specifies how Kerberos-based credentials can be passed in a WS-Security header.

Kerberos overview

Kerberos is an authentication protocol that was developed at MIT during the 1980s. It allows the authentication process to occur securely over untrusted networks. It is a widely used standard for authentication, both in the UNIX® and Microsoft Windows® worlds. In a Kerberos environment, clients obtain *tickets* that grant them access to interact with a particular server for a particular period of time. Servers are able to verify the validity of these tickets, and of the user's identity. Symmetric encryption is used to secure the tickets and keys that are exchanged in a Kerberos environment.

A Kerberos environment requires the presence of one or more Kerberos *Key Distribution Centers* (KDCs). A KDC generates the encryption keys and tickets that are used in the environment.

Kerberos and P8 authorization scenario

Figure 2-4 on page 27 shows the exchanges that occur in a typical Windows Kerberos-based client/server interaction (a UNIX-based Kerberos interaction would be conceptually identical).

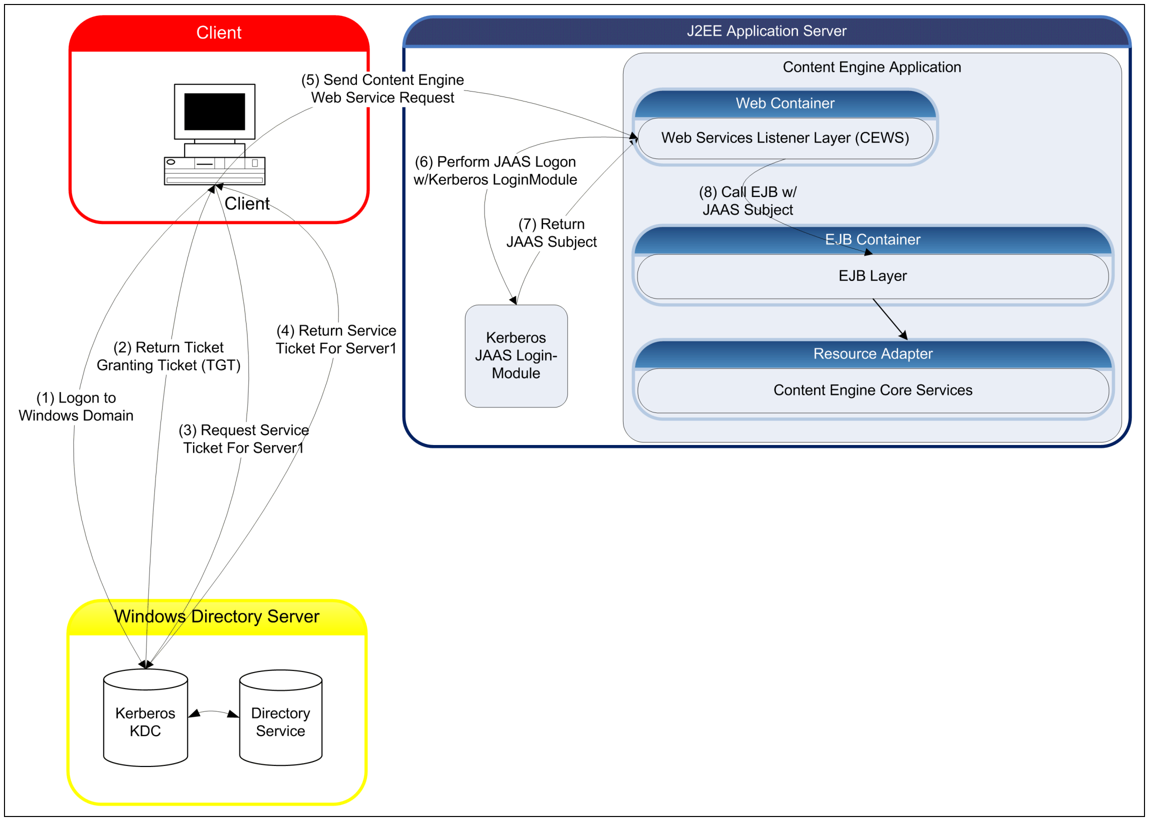

Client
(5) Send Content Engine Web Service Request
Client
(2) Return Ticket Granting Ticket (TGT)
(4) Return Service Ticket For Server1
(1) Logon to Windows Domain
(3) Request Service Ticket For Server1
Windows Directory Server
Kerberos KDC
Directory Service
J2EE Application Server
Content Engine Application
Web Container
Web Services Listener Layer (CEWS)
(6) Perform JAAS Logon w/Kerberos LoginModule
(7) Return JAAS Subject
(8) Call EJB w/ JAAS Subject
EJB Container
EJB Layer
Kerberos JAAS Login-Module
Resource Adapter
Content Engine Core Services

Figure 2-4 Kerberos architecture overview

Where:

1. Authentication: Client logs in to the Windows domain. The operating system collects the users name and password and sends them to a KDC.
2. The KDC works with a directory service to validate the user's credentials, and returns a Ticket Granting Ticket (TGT) to the caller.
3. Ticket granting exchange: The client wishes to access a particular server. It sends a second request to the KDC, specifying the service that it wishes to access. (Note that if the client had previously been granted a service ticket and it has not yet expired, then steps 3 and 4 would be skipped.)
4. The KDC issues a Service Ticket, which grants the caller access to that service.
5. The client sends its request, with the Service Ticket encoded in a WS-Security Kerberos token, to the server.

6. The Web Service Listener on the Content Engine server performs a JAAS login using credentials supplied in the incoming WS-Security header.
7. The JAAS LoginModule validates the client's ticket. No round-trip to the KDC is necessary. The server is able to use its own key (obtained from the KDC with the first use of this LoginModule) to validate service tickets. The LoginModule produces a valid JAAS Subject, based on the successful validation of the ticket. This JAAS Subject is returned to the Web Service Listener.
8. The Content Engine Web Service Listener passes the call along (with valid JAAS Subject) to the Content Engine EJB layer.

When Kerberos tokens are sent in a WS-Security header, a secure, private channel, such as an HTTPS connection, is not required between the client and the server, as the tickets sent in a Kerberos request are already encrypted. Kerberos also provides protection against replay attacks. (Customers may wish to use an HTTPS connection anyway, to ensure privacy for information contained in the request and response bodies.)

2.1.7 Application Engine authentication

The IBM FileNet P8 V4.0 Application Engine (AE) server hosts the Workplace Web application, Workplace Java applets, and Process Engine applets, as well as application development tools. It is the presentation layer for both process and content. There are a number of different components that run on the Application Engine. The sections below discuss how each of the components deal with authentication and single sign-on integrations.

Workplace Web application

This user Web application provides access to the document management and business process management capabilities of IBM FileNet P8. Workplace also supports extended IBM FileNet P8 capabilities, such as forms management, records management, and portals.

Workplace runs within a Web container on a J2EE application server, positioning it to participate in the JAAS-based authentication framework of IBM FileNet P8 V4.0.

The following sections discuss how each of the high level authentication options apply for the Workplace application.

Application managed authentication

This is the mode that is supported in Version 3.5 of Workplace. It is basically a forms-based authentication, but the Workplace application performs the redirection of unauthenticated user requests to a logon page, and encodes the credentials supplied to the logon page, in the user's JSP session. This mode supports user name and password credentials only. The credentials collected from the Application Engine custom logon page are used to perform a JAAS logon programmatically.

This mode will continue to be the default behavior of Workplace in Version 4.0.

Container managed authentication

In this mode, the Workplace application does not control the authentication process. The deployment descriptor for the Workplace application specifies the security constraints required to access Workplace pages. When an unauthenticated user accesses a Workplace page, the J2EE Web container initiates a user challenge, obtains user credentials, and performs a JAAS login based on those credentials.

The deployment descriptor specifies the authentication method that should be used. The following standard methods defined by the Servlet specification are supported.

- Forms Based Authentication

  The container redirects the user to an HTML page, where the user's credentials are collected.

- Basic Authentication

  The container uses standard HTTP options to direct the user's browser to pop up a dialog box and prompt for user name and password credentials.

- HTTPS Client Authentication

  This mechanism requires each user to have its own Public Key Certificate (PKC), and requires the use of an HTTPS (SSL) connection.

2.1.8 Process Engine authentication

In the IBM FileNet P8 V4.0 release, the Process Engine (PE) relies on the Content Engine (CE) for authentication and directory service access operations. The rationale for this design is that the Content Engine server is implementing certain single sign-on and directory service enhancements in the Version 4.0 time frame. The implementation cost for these enhancements is eased by virtue of the integration capabilities provided by running in a J2EE application server and using JAAS as the authentication framework in the Content Engine server. The Process Engine server needs to implement the same set of single sign-on and directory service enhancements, but Process Engine will not be hosted in a J2EE server in the Version 4.0 time frame. The Process Engine is able to leverage the enhancements made within the Content Engine by deferring to the Content Engine for authentication and directory service operations.

The following sections describe the steps that occur as a client of the Process Engine authenticates with the Content Engine server, and then sends the credentials obtained from the Content Engine to the Process Engine server with each request.

The PE Java API client (using CE EJB transport)

Figure 2-5 on page 31 shows a Process Engine Java API client in an IBM FileNet P8 server environment. Note that both the Process Engine Java API and the Content Engine Java API must be present on the client machine (the installer for the Process Engine Java API will install the Content Engine Java API components as well). Note also that Figure 2-5 on page 31 assumes that the Content Engine Java API is configured to work over the EJB transport. See “The PE Java API Client (using CE Web service transport)” on page 33 for a discussion of how this will differ if the Content Engine Web service transport is in use.

Figure 2-5 on page 31 represents a two level authentication process, wherein the client first authenticates with JAAS. The client then authenticates with the Content Engine server, with the application server validating the JAAS credentials by invoking one of the configured LoginModules to confirm that the client's credentials are of a type accepted by this server, and that they are valid. This establishes the caller's identity from an IBM FileNet P8 point of view.

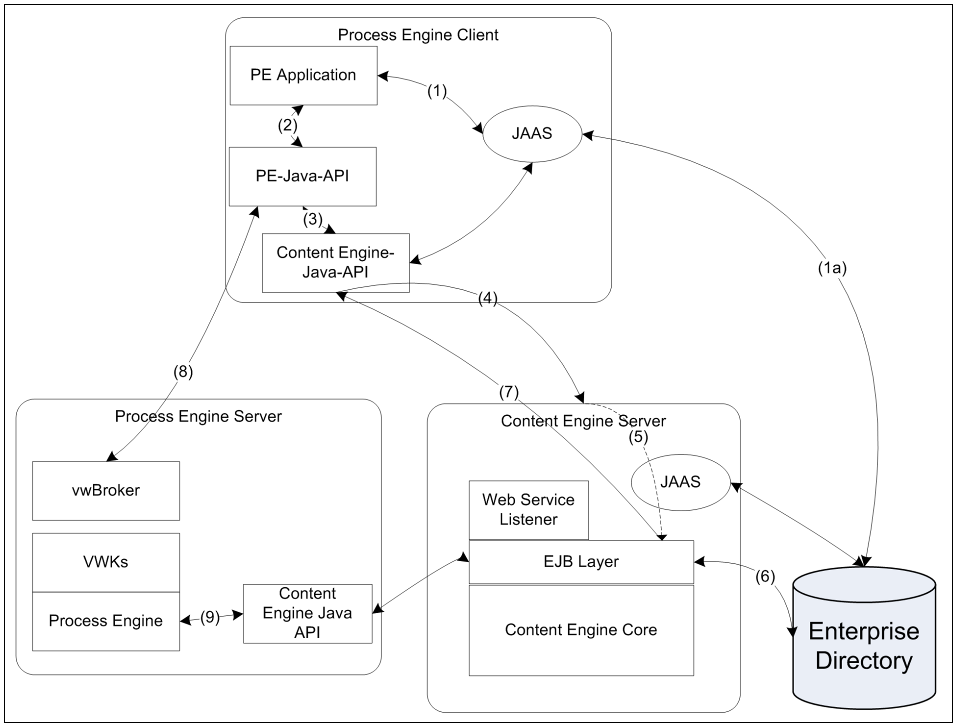


Figure 2-5 PE Java API client

The following steps occur as the Process Engine client application initializes and then makes a call to the Process Engine server.

1. The Process Engine client performs a JAAS login, using the LoginModule configured in the client environment. This login probably occurs before any interaction occurs with the Content Engine or Process Engine APIs. It may happen automatically as a result of a J2EE configuration, or the client may invoke the LoginModule programmatically. For the user name and password case, a helper method to facilitate performing the JAAS login is provided in the Process Engine Java API.

   Depending on what LoginModule is in use, the LoginModule may make a call to the enterprise directory service, possibly through a proxy or some intermediate SSO solution.

2. The Process Engine application makes a call to the Process Engine Java API.
3. The Process Engine Java API sees that the client has not been authenticated to IBM FileNet P8 yet, and therefore makes a call to the Content Engine Java API to obtain an IBM FileNet P8 identity token.

4. The Content Engine Java API makes a call to the server. At the Content Engine server, the call arrives at the J2EE application server's EJB container with the caller's JAAS Subject. The application server examines its security policy configuration, as well as how the Content Engine EJB is configured, to determine what security policy is in place. It determines if the JAAS Subject associated with the incoming request matches one of the authentication providers that are configured and enabled for use with this EJB. If so, it makes a call to this authentication provider, which performs any necessary checks on the JAAS Subject (this may involve contacting a directory service or SSO provider to validate the Subject). If any part of this application server authentication process fails, an appropriate error is returned to the caller (note that this would occur before any logic within the Content Engine EJB is invoked).
5. If the incoming JAAS Subject is validated by the application server, then the call is passed through to the Content Engine EJB, and the JAAS Subject is available to the Content Engine EJB.
6. Within the Content Engine EJB, the Principal name is extracted from the JAAS Subject, and the Principal is looked up in the Content Engine user cache. If this user is not found, then the Content Engine makes a call to the enterprise directory service to expand the caller's identity information.
7. The EJB processes the incoming request. In this case, an IBM FileNet P8 identity token is created, and returned to the caller (the Process Engine Java API).
8. If all of the above steps are successful, the Process Engine Java API now has an IBM FileNet P8 identity token, obtained from a Content Engine server. It now makes a call to the Process Engine server, passing this identity token as a parameter. When this call arrives at the Process Engine server, the identity token is examined, and the signature on it is validated. If the signature is valid, then the Process Engine server trusts that the token is valid, and that the caller is therefore who it claims to be (the user identified by the token).
9. This optional step is actually unrelated to the authentication process, but will occur in many cases. In this step, the Process Engine server wishes to retrieve the detailed user information (full name, DN, e-mail address, group memberships, and so on). It does so by calling the Content Engine server to retrieve the requested User or Group objects.

The P8 identity token used to convey the identity of the caller from the Content Engine server back to the Process Engine server is protected by a pair of cryptographic keys. One of these keys is a secret shared between a Process Engine and Content Engine server, while another key is dynamically generated for use with each token.

The identity of Process Engine clients is confirmed by a Content Engine server, and transmitted to the Process Engine in the form of an IBM FileNet P8 Identity Token.

The PE Java API Client (using CE Web service transport)

For the V4.0 time frame, Process Engine authentication through Content Engine over the Web service transport is only supported for the user name and password case. Alternate credential types over the Web service transport may be added in the post-Version 4.0 time frame. If the EJB transport is used, then other types of JAAS credentials can be supported for Process Engine Java API clients.

The diagram for this scenario is the same as shown in Figure 2-5 on page 31, with the exception that the arrow in step 4 goes to the Web Service Listener, rather than going to the EJB Listener. The Web Service Listener then extracts the user's credentials from the WS-Security header, and performs a JAAS login using a LoginModule that has been configured at the server. Once that step is complete, the Web service listener forwards the call along to the EJB listener, and everything is the same as in the previous section.

The PE Web service API Client

The Process Engine Web service client sends a Web service request to the Process Engine Web service listener. The Process Engine Web service listener is then responsible for performing the JAAS login. Once the Process Engine Web service listener has obtained a valid JAAS Subject, it makes a call to the Process Engine Java API.

Note that the only credential type that will be supported for this scenario in IBM FileNet P8 V4.0 are user name and password credentials. The Process Engine Web service does not support Kerberos credentials, nor does it support the Web Services Extensible Authentication Framework (WS-EAF). It is IBM’s stated intention to support this function in a subsequent release.

Figure 2-6 shows the Process Engine Web service listener running on a separate server from the Content Engine server. Although this is correct from a conceptual point of view, it is likely that the Process Engine Web service listener will actually be co-resident with the Content Engine server (this is the default configuration).

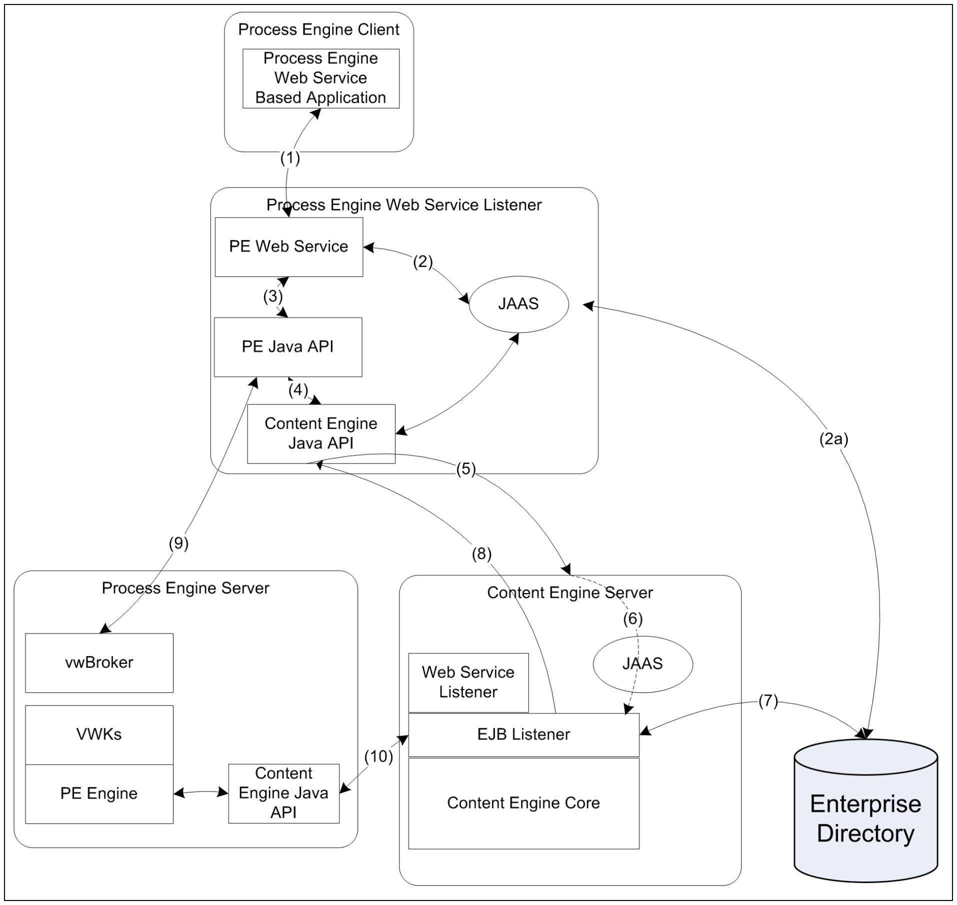


Figure 2-6 PE Web service API Client

This concludes our overview of the IBM FileNet P8 base architecture. In the following sections, we introduce different single sign-on options based on explicit product examples. We introduce the architectural concepts of these individual solutions so that you can better understand the integration scenarios in Part 2, “Technical single sign-on implementations” on page 53.

2.2 Simple Protected GSSAPI Negotiation Mechanism architecture overview

In this section, we introduce the basic *S*imple *P*rotected GSSAPI *Nego*tiation Mechanism (SPNEGO) function, as well as the Kerberos authentication protocol, which is used in the SPNEGO and Windows domain authentication processes.

2.2.1 What SPNEGO is

In this section, we begin with an explanation of the Kerberos protocol. This is because SPNEGO depends heavily on the Kerberos protocol in its authentication process.

SPNEGO is used to provide secure single sign-on functionality from an Internet Explorer® browser, running on a Microsoft Windows desktop, to a Web application. When a user tries to access an SPNEGO enabled Web application server, the SPNEGO authentication token is automatically exchanged between the browser and the Web application server. Hence, the user can log in to the Web application server without being challenged with another login window.

The SPNEGO mechanism is described in RFC 4559[1], even though it is in an informational status. Because its mechanism and protocol are open, we can also use other browsers, such as Firefox and Mozilla.

SPNEGO is tightly coupled with the Windows authentication mechanism. In other words, SPNEGO uses authentication information provided by the Windows OS and trusts it. So in order to use the SPNEGO mechanism, you need an Active Directory domain. In your Active Directory domain, Microsoft Windows can use two kinds of authentication mechanisms. One is called *NTLM authentication*, which is used in Windows NT® based domains, while the other is *Kerberos authentication*, which is used in Windows 2000 and later. If you are using WebSphere Application Server V6.1, only SPNEGO based on Kerberos authentication is supported.

Kerberos

Kerberos is an authentication protocol for distributed environments based on the RFC 1510[2] standard. It is designed for handling sensitive data (authentication information) in untrusted networks.

[1] More information about RFC 4559 SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows can be obtained at http://www.rfc-archive.org/getrfc.php?rfc=4559

[2] More information about RFC 1510 The Kerberos Network Authentication Service (V5) can be obtained at http://www.rfc-archive.org/getrfc.php?rfc=1510

Kerberos makes use of a trusted third party, called a *key distribution center* (KDC), which consists of two logically separate parts: an *Authentication Server* (AS) and a *Ticket Granting Server* (TGS). Kerberos works on the basis of *tickets*, which serve to prove the identity of users.

Components used in Kerberos authentication

The following logical components are used in a Kerberos implementation.

- Key Distribution Center (KDC)

  This component takes a trusted third-party role in Kerberos authentication. This component consists of a TGS and an AS.

- Authentication Service (AS)

  This component is for receiving initial authentication requests from a client. This component verifies the user ID and password from a client. If both the user ID and password are correct, this component issues the ticket that is called a Ticket Granting Ticket.

- Ticket Granting Ticket (TGT)

  The TGT is used to retrieve other *Service Tickets* along the way. A client needs to present a valid TGT to a TGS for being issued a Service Ticket.

- Ticket Granting Service (TGS)

  This component distributes Service Tickets when a user wants to access a secured service. The user needs to send a TGT to the TGS for being issued a Service Ticket.

- Service Ticket

  A client has to present this ticket when accessing a secured service. Actually, the secured service possesses a ticket for itself, albeit different from the client's ticket). Using the embedded encryption key information in the ticket on the service side, the service can verify if the ticket presented by client is valid or not.

How Kerberos authentication works

Figure 2-7 on page 37 shows a generic Kerberos based authentication scenario. which we walk through step-by-step.

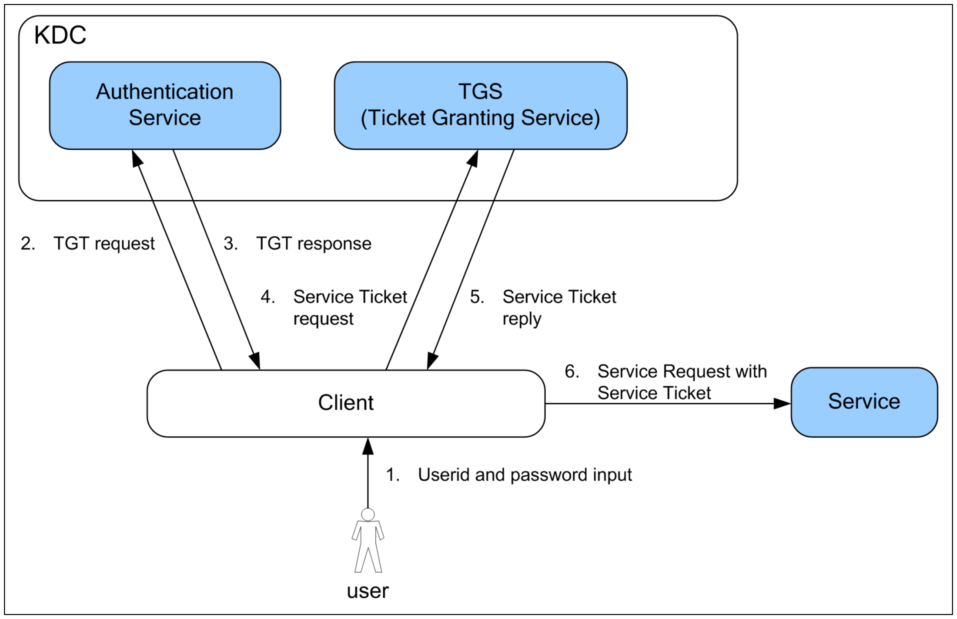

KDC
Authentication Service
TGS (Ticket Granting Service)
2. TGT request
3. TGT response
4. Service Ticket request
5. Service Ticket reply
Client
6. Service Request with Service Ticket
Service
1. Userid and password input
user

Figure 2-7 Kerberos basic workflow diagram

1. A user presents a user ID and password for Kerberos Authentication Service.
2. The Kerberos client sends a TGT request to the Authentication Service.
3. The Authentication Service verifies the client's credential using a repository (when using a Windows domain controller, the database is Active Directory). If it is valid, the AS sends a TGT back to the client.
4. When a user wants to access a secured service, they need to present a Service Ticket. A component that is tightly coupled with the Kerberos client (in our later example, an SPNEGO enabled Web browser) automatically creates a request for getting a Service Ticket from the TGS.
5. TGS validates the request from the client. If it is valid, the TGS sends a Service Ticket back to the client.
6. With the Service Ticket attached, the client sends a request to the Service. The Service verifies if the Service Ticket is valid or not. If it is valid, the Service grants access to the client and provides the Service. The Service can also retrieve identity information from the Service Ticket.

2.2.2 How SPNEGO works

Let us now take a closer look at how the information flows in an SPNEGO authentication scenario using a browser and WebSphere Application Server. This scenario is shown in Figure 2-8, and then we use another step-by-step walkthrough.

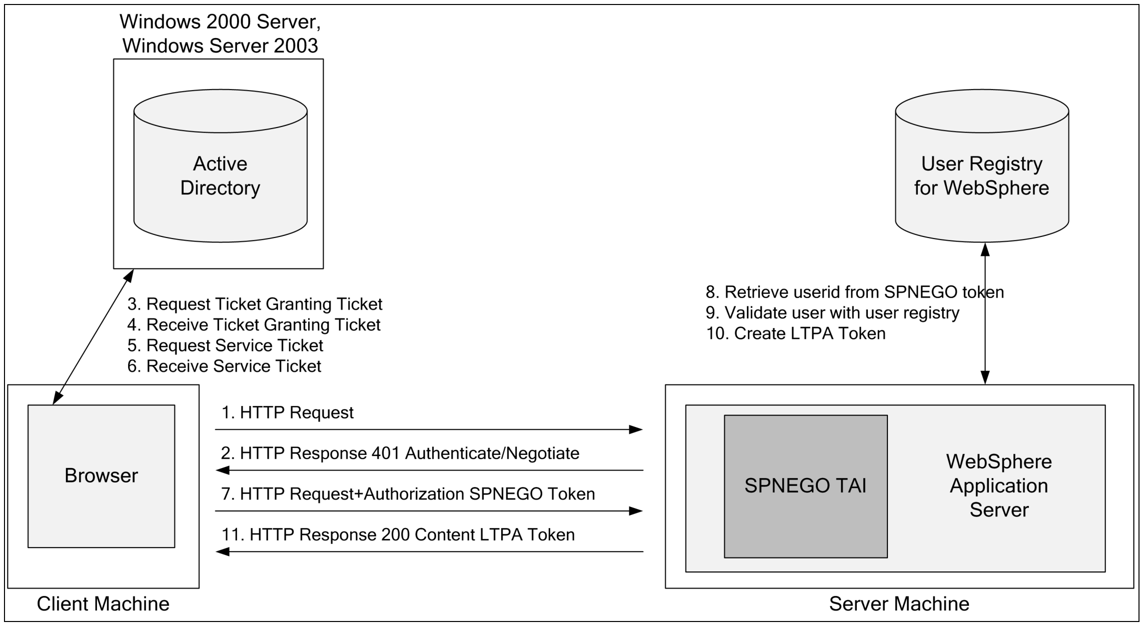

Windows 2000 Server,
Windows Server 2003
Active Directory
User Registry for WebSphere
3. Request Ticket Granting Ticket
4. Receive Ticket Granting Ticket
5. Request Service Ticket
6. Receive Service Ticket
8. Retrieve userid from SPNEGO token
9. Validate user with user registry
10. Create LTPA Token
1. HTTP Request
2. HTTP Response 401 Authenticate/Negotiate
7. HTTP Request+Authorization SPNEGO Token
11. HTTP Response 200 Content LTPA Token
Browser
SPNEGO TAI
WebSphere Application Server
Client Machine
Server Machine

Figure 2-8 SPNEGO work flow diagram

1. The user opens a browser and submits an HTTP request to the Web server. In our case, the request is aimed at the FileNet P8 Application Engine.
2. WebSphere Application Server receives a request from a user and returns an HTTP 401 status response. This response includes a *WWW-Authenticate: Negotiate* header. This header indicates that an SPNEGO token is needed for the client to be authenticated.

Note: Steps 3 to 6 typically happen once for each SPNEGO authentication session in a real Windows environment. Once the client PC receives the ticket, it can cache and re-use it for succeeding requests. If the tickets have timed out, a new one has to be requested.

3. The Windows OS on the client PC requests a Ticket Granting Ticket from the domain controller, which provides the Kerberos *Ticket Granting Server* function and *Authentication Server* function. The domain controller uses

mapping information to connect the Kerberos principal and Windows user account. Using this information, the domain controller validates the Ticket Granting Ticket request from client PCs.

4. The domain controller returns a Ticket Granting Ticket to the client PC.
5. Next, the Windows OS on the client PC requests a Service Ticket from the domain controller.
6. The domain controller returns a Service Ticket to the client PC.
7. Now the Web browser on the client PC can respond to the authentication challenge sent by the WebSphere Application Server. This response includes the SPNEGO token based on the Kerberos tickets.
8. WebSphere Application Server now receives an SPNEGO token from the client PC, and it attempts to validate the token with the secret key for the WebSphere Application Server's service principal. After successful validation, WebSphere Application Server retrieves the user's name.
9. Next, WebSphere Application Server verifies if the user exists in its user registry. For this task, you can utilize the same Active Directory on the domain controller as the registry. If you use Active Directory on the domain controller, you do not need a separate directory in your environment. Otherwise, you need a separate directory, and you also need to implement a directory synchronization mechanism between Active Directory and the user registry for WebSphere Application Server.
10. WebSphere Application Server creates the session for the user.
11. Finally, it returns an HTTP-response and session information in cookies.

The concludes the general overview for an SPNEGO authentication architecture.

2.3 Tivoli Access Manager architecture overview

In this overview, we want to briefly introduce the general application and deployment architecture of Tivoli Access Manager for e-business. We do not go into any great detail, but we thought it important to include this section for a better understanding of the integration scenario. For more information about the Tivoli Access Manager architecture, refer to *Enterprise Security Architecture Using IBM Tivoli Security Solutions*, SG24-6014.

2.3.1 Basic architecture of Tivoli Access Manager for e-business

Tivoli Access Manager is an authentication and authorization solution for corporate Web, client/server, and existing applications. Tivoli Access Manager enables you to control user access to protected information and resources. By providing a centralized, flexible, and scalable access control solution, Tivoli Access Manager enables you to build secure and easily managed network-based applications and e-business infrastructure. Tivoli Access Manager supports authentication, authorization, audit and logging, data security, and resource management capabilities. Tivoli Access Manager for e-business accommodates a broad range of possible user-authentication mechanisms, including user IDs and passwords, client-side certificates, RSA SecurID tokens.

Figure 2-9 shows an overview of the Tivoli Access Manager logical components.

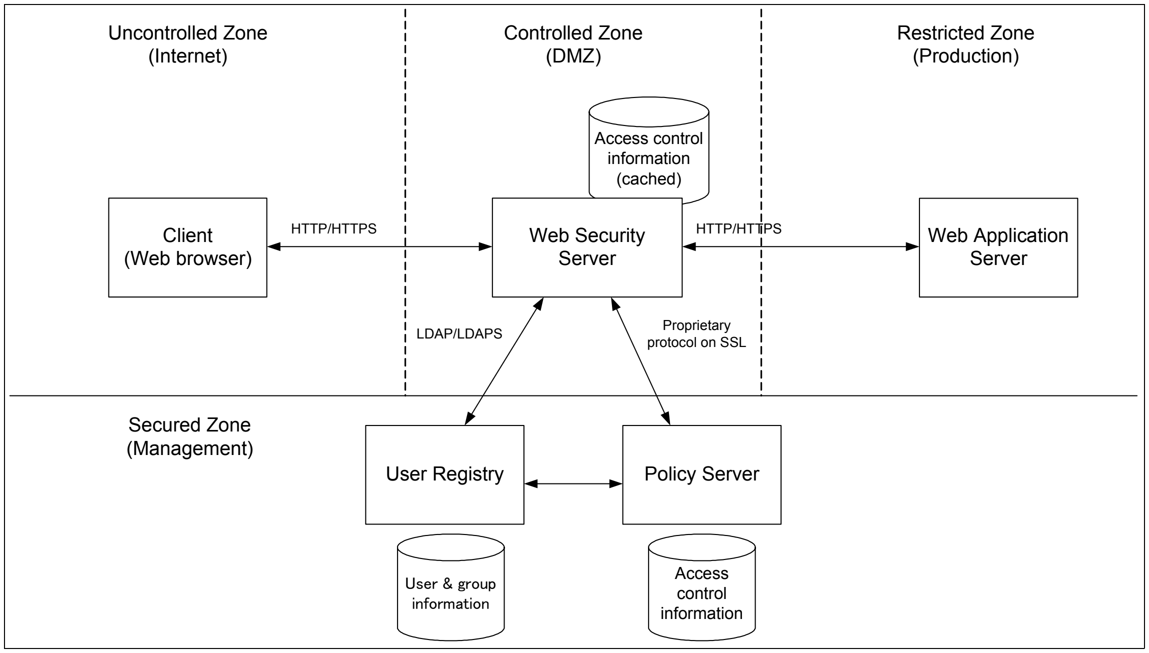

Uncontrolled Zone (Internet)
Controlled Zone (DMZ)
Restricted Zone (Production)
Access control information (cached)
Client (Web browser)
HTTP/HTTPS
Web Security Server
HTTP/HTTPS
Web Application Server
LDAP/LDAPS
Proprietary protocol on SSL
Secured Zone (Management)
User Registry
Policy Server
User & group information
Access control information

Figure 2-9 Tivoli Access Manager for e-business architecture diagram

Tivoli Access Manager for e-business consists of the following three major components:

- User registry
- Policy Server
- Resource manager

As a user registry server, various directory servers, such as Tivoli Directory Server, Microsoft Active Directory on Windows Server® 2003 and 2008, Novell eDirectory, Lotus® Domino®, and Sun Java Directory Server, are supported.

The Policy Server is the central management server of an Access Manager security domain. It maintains the master access control information database and the Access Manager portions of the user and group information in the user registry. It handles administration request commands and maintains consistency with replicated access control information databases that are used by distributed resource managers.

Access Manager uses different resource managers in order to enforce the centrally managed access control policies, such as the Web security server shown in Figure 2-9 on page 40. One of the Web security servers is called WebSEAL. WebSEAL is a multi-threaded *reverse proxy* server that can be placed between browser based clients and Web application servers. The other Web security server to manage authentication and authorization for Web applications is implemented as a *plug-in module* for Web servers. Both Web security servers make use of a locally replicated copy of the access control information database. Highly sophisticated mechanisms ensure proper security and replication mechanisms for this vital data. In the following sections, we focus on the WebSEAL component.

2.3.2 Providing single sign-on functionality

Providing single sign-on is just one part of Tivoli Access Manager for e-business. It allows us to integrate a multitude of Web application servers, which can be deployed within the enterprise, into a centralized access control security architecture. Tivoli Access Manager for e-business supports a wide variety of single sign-on methods, as shown in Table 2-1.

Table 2-1 Single sign-on methods supported by Access Manager

SSO method	**Description**
HTTP basic authentication header	Passing credential information in HTTP basic authentication header to back-end server.
Form based single sign-on	Fill in the login forms and post them to a back-end server.
LTPA	LTPA stands for Lightweight Third Party Authentication. Although Its name includes *third party*, it uses an IBM proprietary single sign-on token that is supported by WebSphere Application Server and Lotus Domino.

SSO method	Description
Trust Association Interceptor	Extensible authentication delegation mechanism of WebSphere Application Server. TAI is the abbreviation for Trust Association Interceptor.
HTTP header	WebSEAL can pass credential information fields in specific HTTP headers. They are iv-user, iv-groups, and iv-creds.
Kerberos ticket	Passing a Kerberos ticket in SPNEGO protocol form. This kind of token passing is supported for the Microsoft Internet Information Server (IIS).

You can use different single sign-on methods for each back-end Web server on a single WebSEAL instance. Each path to a back-end Web server is called a *junction* in Tivoli Access Manager terms. You need to create a *junction configuration* for a each back-end Web server. As shown in Figure 2-10, you can manage multiple back-end Web servers from a single WebSEAL instance. A user only needs to provide his credentials to WebSEAL once and can then use multiple back-end servers without having to provide login information again.

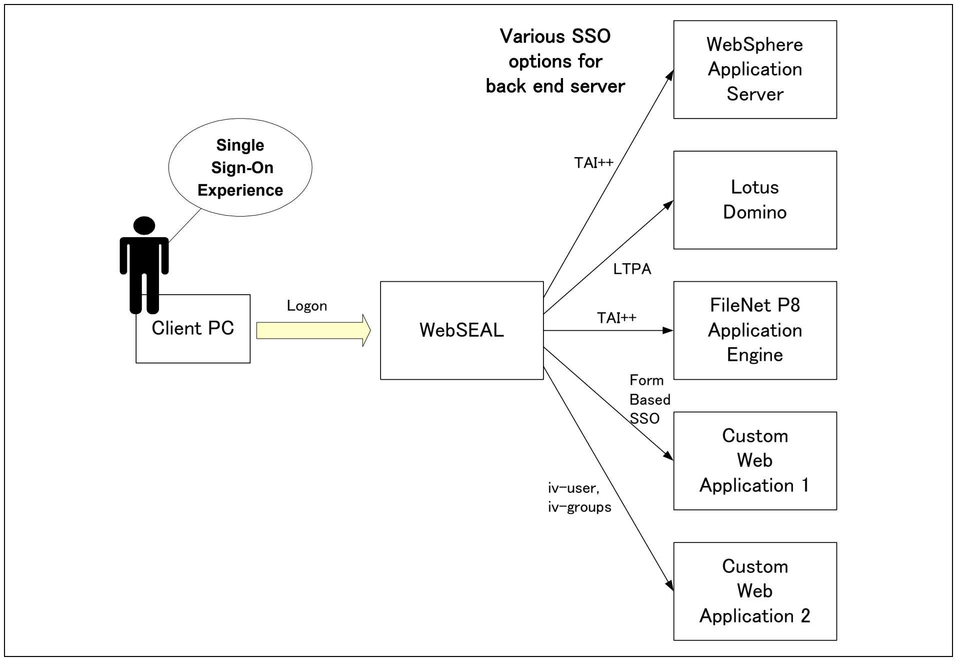


Figure 2-10 WebSEAL as focal point of single sign-on to back-end servers

2.3.3 Access Manager single sign-on flow

In order to implement the single sign-on mechanism using Tivoli Access Manager, it is necessary to pass a credential from the Web security server to the back-end Web application server. WebSEAL provides a number of ways to pass this credential; in our integration with FileNet P8, the TAI++ mechanism (*Trust Association Interceptor*) is the preferred way of passing the credential from WebSEAL to WebSphere Application Server, which runs the FileNet P8 Application Engine.

TAI++ is a function provided by WebSphere Application Server for integrating external single sign-on products. TAI++ allows us to implement a secure identity information transfer between WebSEAL and WebSphere Application Server. Using the diagram shown in Figure 2-11, we describe step-by-step how the TAI++ communication handles the credential information.

Note: A *Trust Association Interceptor* is used to connect reverse proxies, such as Access Manager WebSEAL, or any third-party offerings, to a J2EE application server. A TAI allows for single sign-on (SSO) and management privileges, for example, authentication, authorization, and policy-based security, within J2EE resources.

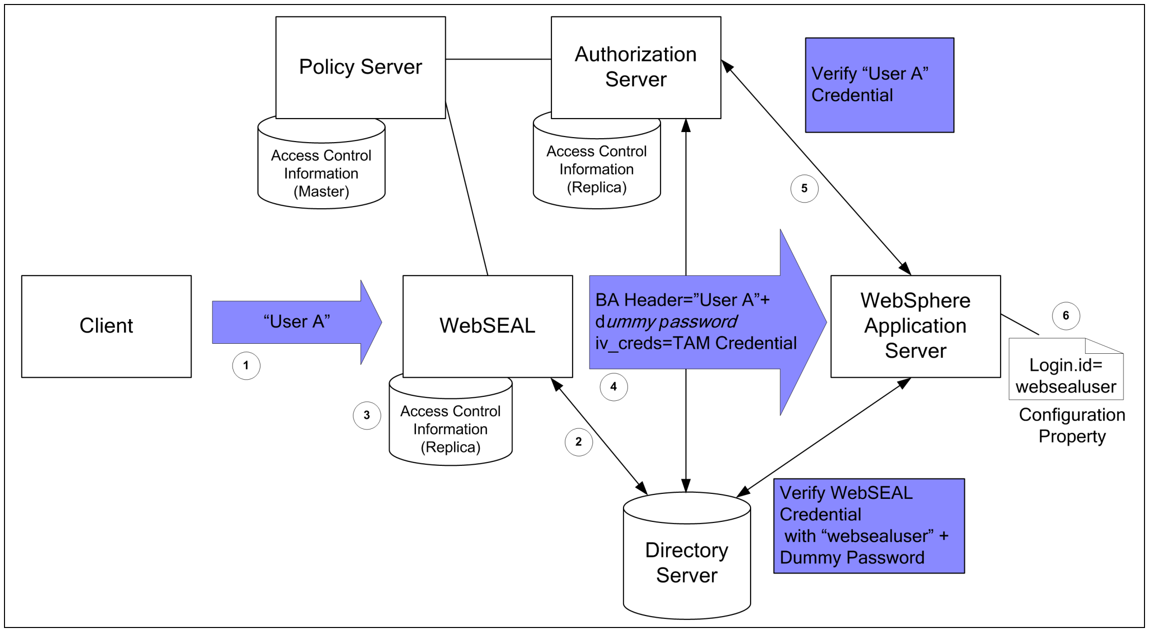

Policy Server
Access Control Information (Master)
Authorization Server
Access Control Information (Replica)
Verify "User A" Credential
5
Client
"User A"
1
WebSEAL
3
Access Control Information (Replica)
2
BA Header="User A"+ d*ummy password* iv_creds=TAM Credential
4
WebSphere Application Server
6
Login.id= websealuser
Configuration Property
Directory Server
Verify WebSEAL Credential with "websealuser" + Dummy Password

Figure 2-11 TAI++ flow diagram

Do these steps:

1. WebSEAL receives a set of user credentials from a client who wants to access a protected resource that is located behind WebSEAL.
2. WebSEAL verifies the user's credentials using directory server. If a password is used in the credential, WebSEAL sends a password compare request to the directory server. The directory server sends a result back to WebSEAL, and the password provided by the user either matches or not. If the verification was successful, WebSEAL creates a user session context in its memory.
3. WebSEAL makes an authorization decision using the access control information managed by the Access Manager Policy Server and user identity information. WebSEAL's decision is actually executed based on the local replica of the Access Control Information database.
4. If access is permitted, WebSEAL passes the original request to the back-end server. In the case of our TAI++ configuration, additional information is added to the original request. This information consists of the HTTP BA Header (basic authentication header) and Access Manager iv_creds credentials.

   The HTTP BA Header is a combination of the client's user ID and a password that has been specified in the WebSEAL configuration file. This password is referred to as a *dummy password*. This means that it is a common password that is used for all user requests that are passed on from WebSEAL to the back-end Web servers. This dummy password is used to verify the trust relationship between WebSEAL and the WebSphere Application Server. Note that this password is not used to verify the actual user.
5. Instead of using the HTTP BA Header, iv_creds is used for passing the actual user credentials. The iv_creds field uses a proprietary Tivoli Access Manager format, so WebSphere uses a built-in Tivoli Access Manager endpoint to decode iv_creds header by calling the Tivoli Access Manager Java Authorization API. This way, WebSphere Application Server can verify the transmitted credential of the original user.
6. If the credentials are positively verified, WebSphere Application Server allows access to the appropriate resources.

2.4 Tivoli Federated Identity Manager architecture overview

Tivoli Federated Identity Manager services use standardized means for allowing businesses to:

- *Engage in trust relationships* that facilitate direct integration of business processes in the most efficient fashion. The concept of business federations directly provides services for customers registered at other (Business Partner) organizations by establishing business trust relationships.
- *Share identity information and entitlements* in a trusted fashion between organizations. Current approaches to identity management generally rely on organizations incurring user life cycle management costs by maintaining redundant identities to manage employees, Business Partners, and customers. The relationship between the business and these individuals can change frequently. Each change requires an administrative action that can result in a high cost of user life cycle management.
- *Exchange*, in a secure and trusted manner, *tokens referring to a principal*, their attributes, privileges, and so on. These tokens are used to communicate information used for the authentication and authorization of a principal to a Business Partner.
- *Maintain security in a Web services oriented architecture*, allowing for secure standards based application-to-application inter-enterprise communication.

In order to better understand the integration between P8 and Federated Identity Manager, we provide an overview of the Federated Identity Manager functions in the following two sections covering the trust service and identity propagation patterns.

2.4.1 Trust Service

Tivoli Federated Identity Manager delivers a key functionality called *Trust Service* to enable identity federation solutions. This service is the basis for providing *federated provisioning*, *Web single sign-on*, and *Web service security management* solutions. Each of these solutions may be deployed independently, or they can be deployed together within an SOA environment to deliver a standards-based identity federation solution. In this book, we focus on the Web services security management abilities of Tivoli Federated Identity Manager and how it can be used to provide FileNet P8 with the ability to consume additional security tokens. This approach also allows us to address identity propagation in an SOA environment involving FileNet P8.

The IBM SOA identity propagation solution is built on open standards. The WS-Trust standard (part of the WS-Security family of standards) is the open mechanism by which:

- Security tokens can be validated, issued, and renewed
- Trust relationships can be established, assessed, and brokered

WS-Trust[3] is defined by a Web services interface. The service that implements the WS-Trust interface is known as a *Security Token Service* (STS).

In the IBM SOA identity propagation solution, the STS is a component of the Tivoli Federated Identity Manager product. Figure 2-12 shows the interaction between a WS-Trust client and the Tivoli Federated Identity Manager STS.

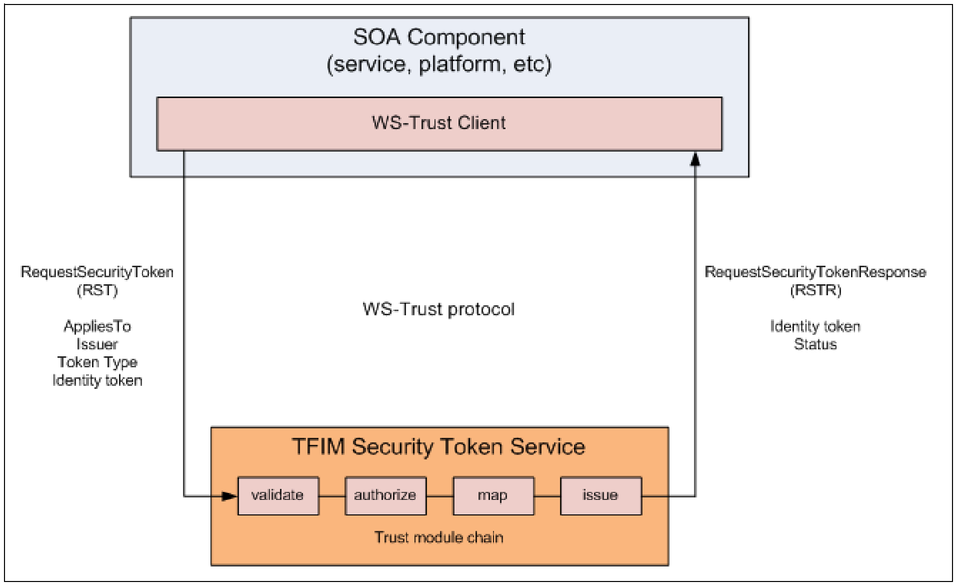


Figure 2-12 WS-Trust protocol and Federated Identity Manager Security Token Service

The STS configuration includes a set of *trust module chains*. The particular trust module chain selected to process a WS-Trust request is determined by matching the *AppliesTo*, *Issuer,* and *Token Type* parameters of the request with the same configuration properties of each trust chain. Usage of these parameters is described in Table 2-2 on page 47.

[3] The WS-Trust specification is available at http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf.

Table 2-2 Important parameters in a WS-Trust request

Parameter	Description	Example
AppliesTo	A representation of which service the WS-Trust relates to, or for what scope the requested security token is required. Typically in URL format.	`http://finance.itso.ibm.com/CreditService`
Issuer	Specifies the issuer of the security token that is presented in the WS-Trust message.	`urn:itfim:wssm:tokengenerator`
Token Type	URI describing the type of token requested in the response to the WS-Trust request.	`http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0`

A trust module chain consists of a sequence of module instances, as shown in Figure 2-13. Data from the WS-Trust request (RequestSecurityToken message) is placed into an XML document called the *STS Universal User* (STSUUSER). An STSUUSER document structures data as follows:

- A *Principal* element
- An *AttributeList* element
- Original data from the RequestSecurityToken message

The STSUUSER document is passed between modules in the trust module chain and transformed by the modules. At the completion of trust module processing, data from the STSUUSER is reformed into the WS-Trust response (RequestSecurityTokenResponse message).

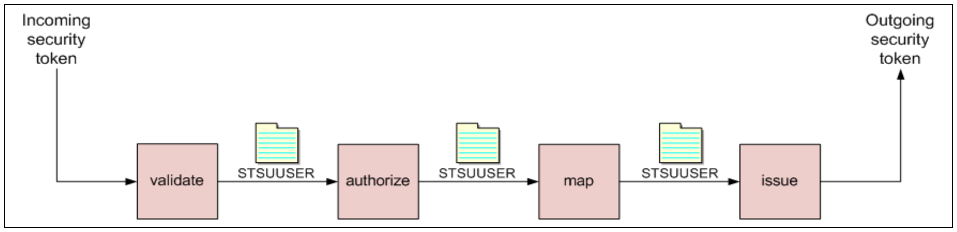


Figure 2-13 Processing a trust chain

Module instances can be configured in different modes, as described in Table 2-3. A minimally configured trust module chain will likely have modules configured in validate - map - issue modes. An authorization module may optionally be inserted before or after the map module. Multiple map modules may also be required in cases where identity mapping data is required to be retrieved from multiple data sources. The structure of the trust module chain shown in Figure 2-13 on page 47 is typical but does not represent the structure that *all* trust module chains must follow.

Table 2-3 Modes for module instances in a trust module chain

Module instance mode	Purpose
Validate	Validates an identity token. The specific validation will vary according to the token module type. For example, the SAML 2.0 token module validates the XML structure of the SAML 2.0 assertion, its compliance with the specification, and, if present, validates the signature on the assertion.
Map	Transforms the STSUUSER document passing through the trust module chain. Mapping modules can be defined by XSL transforms or perform lookups from data sources such as LDAP.
Other	General purpose processing mode. For example, modules that perform authorization would be configured in this mode.
Issue	Generates an identity token from the STSUUSER document.

The Tivoli Federated Identity Manager STS supports a variety of identity token types, including:

- User name
- SAML assertion (versions 1.0, 1.1, and 2.0)[4]
- LTPA
- Kerberos
- X.509 certificate

Additional token modules can be constructed in Java code when required for particular scenarios. Modules that represent an identity token type are typically configured in *validate* or *issue* mode.

[4] SAML assertions are particularly suited for use in identity propagation scenarios because they are based on an open standard widely implemented by vendors, do not require password synchronization, provide for arbitrary attribute lists, and offer selective protection of attribute data through digital signatures and encryption.

2.4.2 Identity propagation patterns

Integration with the SOA identity propagation solution follows one of three general patterns:

- Service Requester
- Service Provider
- Intermediary

These patterns are introduced in this section.

Service requester pattern

The *service requester pattern* recognizes the need for consumers of a service to send a service *what it expects*. This pattern represents the authenticated identity in the service component in an identity token and uses an STS to transform the authenticated identity to an identity token suitable for sending in the service request (Figure 2-14). The most common use of this pattern is to prepare an identity token that contains an identity in the domain of the receiving service component and in the format expected by the receiver of the service request, whether it is an intermediary such as an ESB or a service implementation itself.

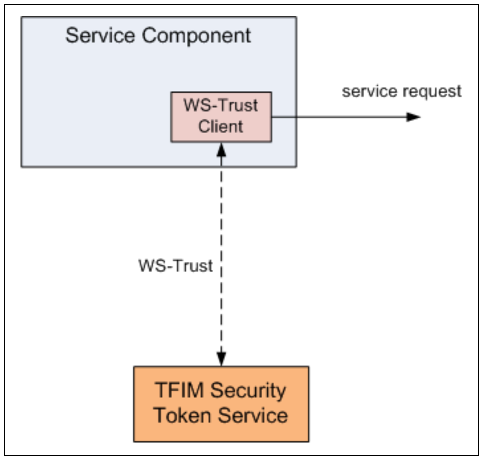


Figure 2-14 Service requester identity propagation pattern

Examples of the service requester pattern include:

- Tivoli Federated Identity Manager Web Services Security Management (WSSM) Token Generator component
- WS-Trust aware JAAS login module

Service provider pattern

The *service provider pattern* is shown in Figure 2-15. An incoming identity token is sent to the STS for validation and mapping to a local identity. This pattern is used in cases where the receiving service component is expected to accept an identity token that it is not able to natively support. The motivation for this might be that an enterprise-wide token standard has been employed or that the validation capabilities of the service component are insufficient for the requirements of the particular SOA environment.

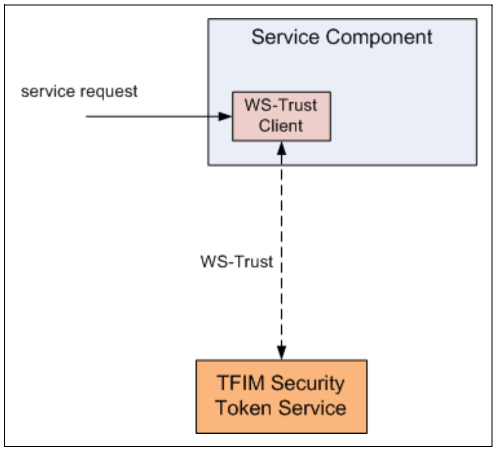


Figure 2-15 Service provider identity propagation pattern

An example of this pattern is the Tivoli Federated Identity Manager WSSM Token Consumer component.

Intermediary pattern

The *intermediary pattern* enables identity propagation through an intermediary, such as an enterprise service bus (ESB), as shown in Figure 2-16 on page 51. This pattern is a combination of the service requester and service provider patterns, where incoming identity tokens are required to be validated (as in the service provider pattern) before the identity token for an outgoing request is generated (as in the service requester pattern). ESBs are required to perform identity mediation, as they often sit at the boundary of different administrative domains. The intermediary pattern allows for solutions where different identity mediations may be required for connecting to each service referenced within a single mediation flow. The flexibility of performing identity mediation within a mediation flow is an advantage of using this pattern. This may reduce the need to use the service requester and service provider patterns in individual service components and reduce the overall complexity of the SOA identity propagation solution.

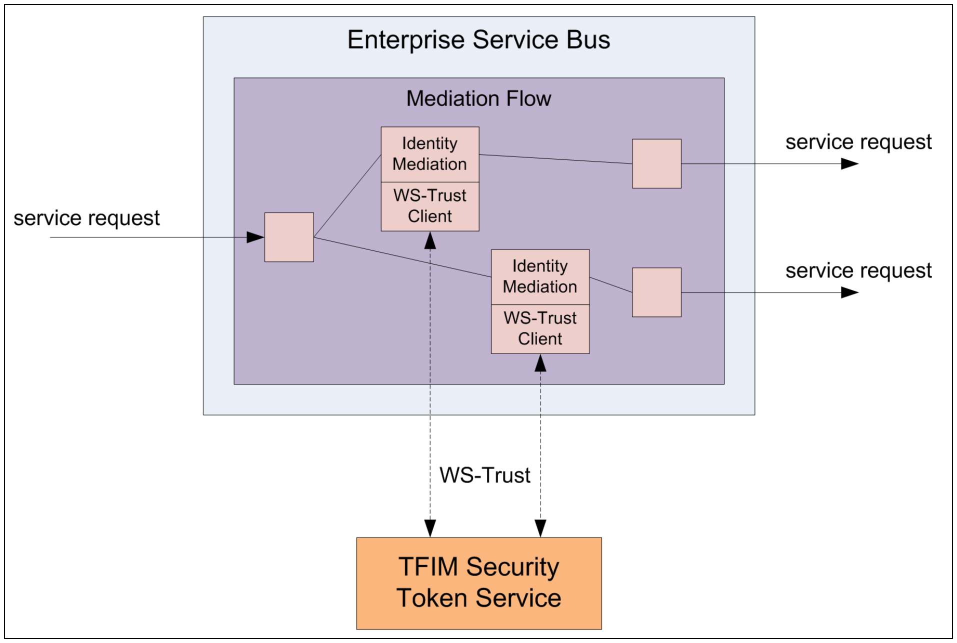


Figure 2-16 Intermediary identity propagation pattern

Examples of this pattern include the Tivoli Federated Identity Manager integration with:

- WebSphere Enterprise Service Bus
- WebSphere Message Broker
- WebSphere DataPower® SOA appliances

For more comprehensive information about the architecture for identity propagation within the IBM SOA Security Reference Model, refer to Chapter 2, “Architecture and technology foundation”, in *Understanding SOA Security Design and Implementation*, SG24-7310.

2.5 Conclusion

In this chapter, we prepared you to better understand the detailed single sign-on scenarios that we cover in the following chapters. We provided architectural introductions to all products and technologies that are involved in these scenarios, including IBM FileNet P8, IBM Tivoli Access Manager, IBM Tivoli Federated Identity Manager, and the SPNEGO and Kerberos technologies. Now is the time to get started and implement SSO.

Part 2

Technical single sign-on implementations

In this part, we discuss how you can deploy different approaches for single sign-on in your FileNet P8 environment. We introduce a fictional customer scenario with real-life business challenges and demonstrate various approaches in order to address the single sign-on challenges.

3 Customer overview

To illustrate the concepts that we discuss in this book, this chapter discusses a scenario about a fictional bank called Gold Coast Banking Inc. (GCBI). This chapter provides an introduction to the overall structure of Gold Coast Banking Inc., including its business profile, its current IT architecture and infrastructure, as well as their medium-term business vision and objectives.

Note: All names and references for the company and other business institutions used in this chapter are fictional. Any match with a real company or institution is coincidental.

3.1 Company profile

GCBI is a major financial services company in Australia that has been in operation since 1930. The bank offers many banking solutions, including retail banking, business banking, mortgage loans, personal loans, credit cards, and insurance. The following sections describe:

- The geographic distribution of GCBI
- The company organization
- The companies position within the marketplace

Note: The following section describes company information that is relevant to a FileNet P8 single sign-on solution and is not intended to be a complete description of the company.

3.1.1 Geographic distribution of GCBI

Gold Coast Banking Inc. is located at the Gold Coast, Queensland, with a central head office and data center located in Brisbane, Queensland. GCBI operates an office at the Gold Coast, known as the east office, and an office in Perth, Western Australia, known as the west office.

Each office contains a small group of IT professionals who perform functions such as help desk support and user administration. GCBI also maintains branches in the major capital cities of Australia so that customers can perform their banking needs. The branches do not contain any IT staff and rely on support from staff in the east and west offices for technical support. An overview of the geographic distribution is shown in Figure 3-1 on page 57.

The GCBI offices are:

Brisbane, Queensland
This is the IT center that houses the core IT infrastructure and staff.

Gold Coast, Queensland
This is the east office for the east coast of Australia. It includes technical support staff for the east coast. This office also has bank branch facilities.

Perth, Western Australia
This is the west office for central and the west coast of Australia. It includes technical support staff for the west coast. This office also has bank branch facilities.

The GCBI branches are:

Adelaide	This site includes branch banking facilities and is supported by the west office in Perth.
Canberra	This site includes branch banking facilities and is supported by the east office in the Gold Coast.
Darwin	This site includes branch banking facilities and is supported by the west office in Perth.
Hobart	This site includes branch banking facilities and is supported by the east office in the Gold Coast.
Melbourne	This site includes branch banking facilities and is supported by the east office in the Gold Coast.
Sydney	This site includes branch banking facilities and is supported by the east office in the Gold Coast.

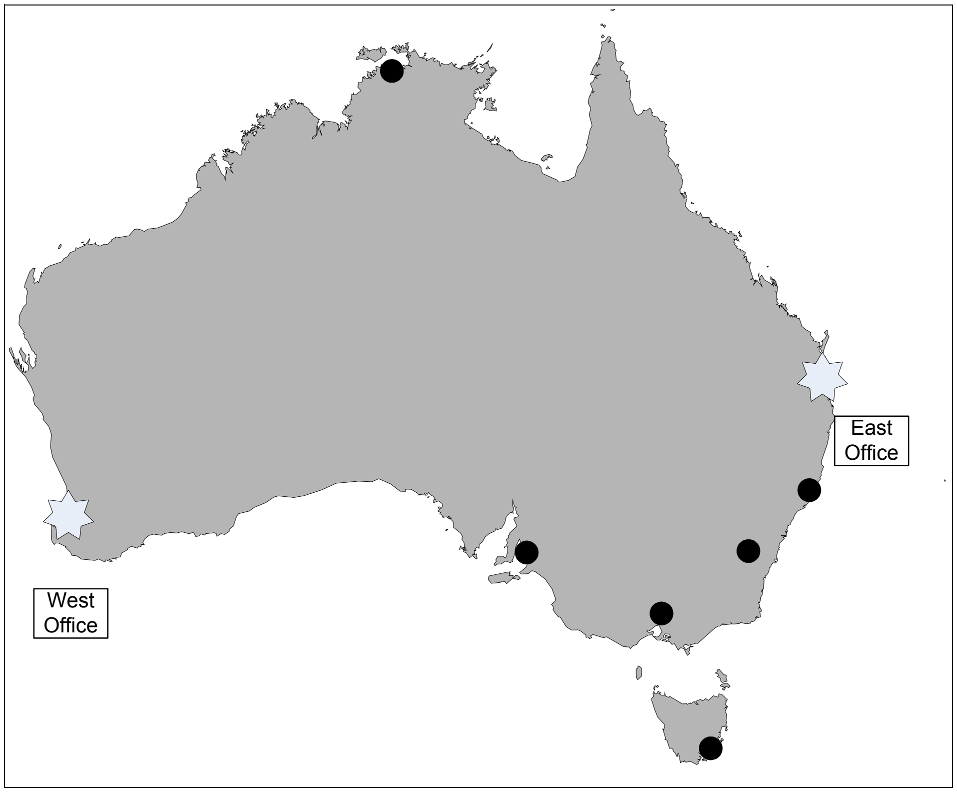


Figure 3-1 Geographic distribution

3.1.2 GCBI marketplace position

GCBI is currently positioned as a leading banking organization in Australia. Due to increasing competitive market demands, GCBI is interested in expanding its product portfolio offerings. GCBI executives feel that the best way to do this task is to partner with existing banks and insurance companies who share a similar corporate vision.

GCBI has identified the Queensland Insurance Solutions Agency (QISA) as a partner to provide insurance for the banking products that GCBI sell. In terms of their relationship, QISA is an independently run organization, but each of them need to be able to access business content based on specific business processes in order to perform their day to day business effectively.

3.2 Current IT architecture

In this section, we describe the current IT environment at GCBI. This includes:

- An overview of the GCBI network
- Current Filenet P8 infrastructure
- Windows server and desktop environment

3.2.1 Overview of the GCBI network

The GCBI network consists of a series of redundant links between the main data center and its offices. Each branch connects through a VPN tunnel to the office that supports them. A series of necessary routers, firewalls, and network appliances are used throughout the GCBI network to ensure that the network and servers are secure.

All links are leased services. GCBI relies on the service provider to ensure the necessary uptime, as agreed in the service level agreement. Therefore, the diagrams in Figure 3-2 on page 59 are logical and do not show redundant or standby links or triangulation of the network. GCBI relies on the service provider for this function.

3.2.2 Current Filenet P8 infrastructure

GCBI uses the Filenet P8 platform for all of its enterprise content management needs. This includes several BPM managed new loan processing applications, such as mortgage, small business, and automobile loans. In order to gain economies of scale and best leverage their investment in the P8 platform, GCBI would like to deploy these new loan applications to newly acquired banks. This can ensure consistent business processes and process visibility across the enterprise. Additionally, GCBI would like to extend the existing P8 infrastructure to include the single sign-on experience both for internal intranet/VPN users and business partners through the Internet.

The existing P8 deployment, shown in Figure 3-2, consists of a Content Engine, Process Engine, and Application Engine that are deployed in the central data center. The regional offices are using a browser to access the P8 content and business process management applications. Additionally, Capture Professional is deployed at each location to enable scanning and to capture faxes for import into the Content Engine.

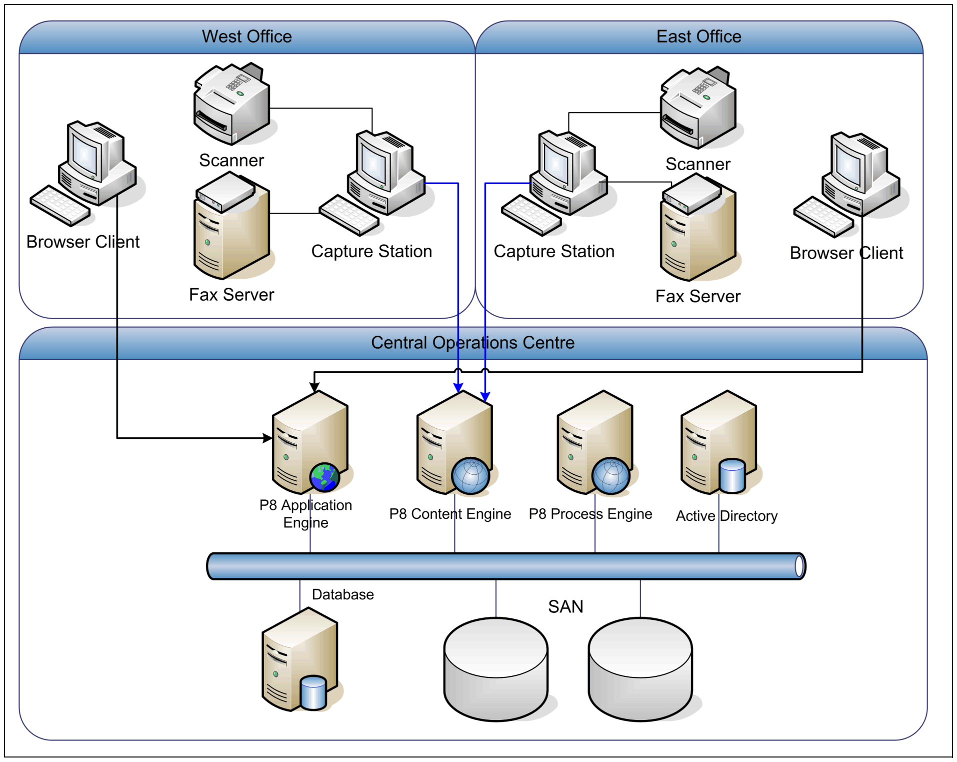


Figure 3-2 GCBI P8 infrastructure

3.2.3 Windows server and desktop environment

GCBI employs a standard operating environment that consists of Windows XP as its operating system. The company uses Active Directory as its user registry and every desktop machine is part of the AD domain. Users log in to their workstations as domain users and access a number of internal Web applications. A limited number of remote users connect to the corporate network through a VPN tunnel.

3.3 Corporate business vision and objectives

In the changing global enterprise, GCBI has realized that it can no longer continue performing its business the way it has in the past. Management has identified key criteria to ensure ongoing success for GCBI, particularly around IT management, security audit, and compliance. These criteria are:

- Reduction of costs around IT management of people, particularly passwords, within the organization.

  One of the main costs associated with the help desk service is the resetting of passwords for users. In order to reduce this cost, the introduction of a desktop based single sign-on solution is in order. If implemented, the P8 team must create a solution where users open a browser on their Windows workstation and browse to the FileNet Workplace application without having to enter any user credentials.

- Introduction of centralized security architecture for continued audit and compliance purposes.

  The introduction of a centralized security architecture means that the FileNet P8 team is challenged to integrate their SSO capabilities with this technology. This means that all users, whether they login through the Windows domain or through the VPN intranet, must be able to sign in to an application, including FileNet P8, so that centralized logging and audit of the event will take place.

- Provision access for Business Partners to internal applications through the use of Web services, which mutually benefit both the partner and GCBI.

  The provisioning of partner access to the FileNet P8 infrastructure through the use of Web services places additional security requirements upon the FileNet P8 team. Corporate policy dictates that the existing support for a Username Token is insufficient and a more appropriate token, such as SAML, must be used. This means that the FileNet P8 team must deploy a solution that can translate the incoming security token in the Web service request into a format that the FileNet P8 platform can use.

3.4 Business requirements

The following business requirements are derived from the business vision and objectives that we outline in 3.3, “Corporate business vision and objectives” on page 60.

1. The company needs to reduce the workload on employees with respect to password management. The CIO would like to see improvements in the time and effort required by employees who need to log into many different applications over the course of a work day.
2. Corporate security policy should be enforced for all user accounts and their attributes, access rights, and password rules. In particular, the new policy that passwords in different registries should be different needs to be addressed.
3. User and administrative historical data, such as application logins and policy changes respectively, must be kept for auditing purposes.
4. External partners need to be able to utilize the FileNet P8 platform through Web services without having to import all of their anticipated users into the FileNet P8 directory.

3.5 Functional requirements

We extract functional requirements by mapping business requirements to their underlying reasons. Our functional requirements then tie these reasons for a business requirement to the single sign-on capability that fulfill that business requirement.

Let us examine every business requirement and search for reasons and the functional requirements.

- Functional requirements that are associated with business requirement #1: Reduce the workload on employees with respect to password management.

  While employees access multiple applications, some of which utilize the existing Active Directory infrastructure, employees must still log in to each application. Therefore, the employees have to remember the passwords for the different applications.

To address these concerns, the new system should address the requirements that are listed in Table 3-1.

Table 3-1 Functional requirements

Requirement	Description
A	Employees need to memorize only their desktop password.
B	Employee is logged in to all applications that require authentication automatically.
C	Employees can reset their desktop Windows password from the desktop itself.

- Functional requirements that are associated with business requirement #2: Corporate security policy should be enforced for all user accounts.

  Each desktop application with user accounts stores the user passwords in its own way and has its own set of supported password policies. Some of these policies might not meet corporate standards and, even if all applications can meet the standards, managing the policies on all the various applications becomes expensive as the number of applications grow.

  GCBI requires the password policy to be managed easily and securely in an enterprise setting, as shown in Table 3-2.

Table 3-2 Functional requirements continued

Requirement	Description
D	Enforce corporate password policies, even for applications that do not enforce it themselves.
E	User credentials are encrypted in any persistent storage location.
F	Maintain user accounts and their attributes, access rights, and password rules from a central location.

- Functional requirements that are associated with business requirement #3: Availability of historical data.

  GCBI uses historical data to evaluate the effectiveness of security solutions as well as to gather usage statistics. Such data can also be used to troubleshoot problems. GCBI must be able to access this data quickly when necessary, as shown in Table 3-3.

Table 3-3 Functional requirements continued

Requirement	Description
G	Security logs from application access must be consolidated on a central system.

- Functional requirements that are associated with business requirement #4: Ability for external partners to utilize the FileNet P8 platform through Web services.

  GCBI has an increasing number of partners who are crucial to its ability to perform its day to day business. GBCBI must allow those partners secure access to content for processing without having to import all of the partners' users into the GCBI corporate directory, as shown in Table 3-4.

Table 3-4 Functional requirements continued

Requirement	Description
H	Multiple Web service security tokens, including custom tokens, must be used during inter-company Web services transactions.
I	Remote partner users should not have to be imported into the local application user repository.

3.6 Solution approaches

Based on the corporate business vision, GCBI has decided that there are three approaches that help support their new business objectives. Each approach addresses different subsets of the functional requirements. The three approaches are:

1. Introduction of desktop based SSO: WebSphere Application Server SPNEGO

   This approach addresses SSO from within GCBI's intranet desktop environment into FileNet P8 using WebSphere Application Server's SPNEGO implementation. This approach also addresses the addition of a separate independent Active Directory through the acquisition and use of WebSphere Application Server's federated repository virtual member manager.

2. Introduction of Web based SSO: Tivoli Access Manager for e-business

   This approach addresses SSO by using a centralized Web security server, such as Tivoli Access Manager for e-business. This approach also addresses SSO to the centralized Web security server from within the intranet desktop environment.

3. Introduction of Federated SSO for Web services: Tivoli Federated Identity Manager

   This approach addresses SSO from a Web services point of view by leveraging Tivoli Federated Identity Manager to perform security token exchange for Web service requests using WS-Security standards. This approach also addresses the mapping of external partner identities into local identities when accessing Web services.

Table 3-5 shows the functional requirements and the different approaches that address or fulfil a particular requirement.

Table 3-5 Functional requirements and approach fulfilment

Functional requirement	Desktop-based SSO	Web-based SSO	Federated WS Security
A	Yes	Yes	N/a
B	No	Yes	N/a
C	Yes	Yes	N/a
D	No	Yes	Yes
E	Yes	Yes	Yes
F	No	Yes	Yes

Functional requirement	Desktop-based SSO	Web-based SSO	Federated WS Security
G	No	Yes	Yes
H	No	No	Yes
I	No	No	Yes

Required personnel

Each approach requires different personnel, each with a different skill set. All of the approaches contain different roles that need to be fulfilled in order to implement the approach successfully. GCBI has identified the following roles:

- IT security leader
- FileNet P8 administrator or consultant
- WebSphere Application Server security and federated repository expert
- Tivoli Access Manager for e-business administrator or consultant
- Tivoli Federated Identity and Web service security administrator or consultant

3.7 Non-functional requirements

In this section, we consider the non-functional requirements of GCBI. While business and functional requirements are the main parts of the security design objectives, we also have to consider other non-functional requirements and constraints. These can include objectives that are necessary to meet general business requirements, or practical constraints on constructing security sub-systems. Single sign-on implementations often involve non-functional requirements relating to:

- Backup and recovery
- Performance and capacity
- Change management
- Budget and staffing

Because we focus on the multiple single sign-on architectures across multiple products in FileNet P8 environment, we do not look in detail at all of these non-functional requirements. We just assume that these requirements need to be addressed based on the single sign-on implementation chosen.

3.8 Conclusion

In this chapter, we introduced Gold Coast Banking Inc. We introduced the company profile, their geographic and marketplace position, their current IT architecture, including their FileNet P8 infrastructure, and their corporate business vision and objectives. This introduction lead to a set of business requirements and functional requirements that could be implemented using different single sign-on approaches.

The single sign-on approaches introduced (desktop single sign-on, Web based single sign-on, and federated Web services security) all have aspects that address the business needs of GCBI. These approaches are not considered as multiple phases of a project but more as independent stand-alone methods that independently fulfil a subset of the functional requirements. In this case, a combination of the Web-based approach and the federated Web services security approach would fulfil all of the functional and business requirements of GCBI. That being said, the remainder of this book addresses each approach independently and provides a number of possible solutions. These are organized as follows:

- Chapter 4, “Single sign-on using Tivoli Access Manager for e-business” on page 67
- Chapter 5, “Single sign-on using SPNEGO” on page 127
- Chapter 6, “Single sign-on using Tivoli Federated Identity Manager” on page 181

4

Single sign-on using Tivoli Access Manager for e-business

In this chapter, we describe the necessary configuration steps for the single sign-on integration between Tivoli Access Manager for e-business and the IBM FileNet P8 Application Engine. The chapter is outlined as follows.

4.1 Customer requirements

In this section, we briefly want to reiterate GCBI's requirements for the Web single sign-on solution that we formalized in 3.5, "Functional requirements" on page 61. We are adding additional challenges that are often encountered in larger organization's IT operations environments.

- Customer pain

  In addition to FileNet P8, GCBI has deployed many Web applications. These Web applications use different authentication mechanisms, and some applications even use their own user registry, many of which are not based on Active Directory. These user registries are not being synchronized with Active Directory, which is the main identity store for GCBI. Users have to log in to each application using separate user ID and password combinations.

- Solution

  GCBI wants to consolidate the authentication mechanisms for all Web applications. The Web single sign-on solution provides a single login page for back-end Web application servers and forwards user credential information to those back-end Web application servers on behalf of the users.

  In addition to providing a Web single sign-on solution, GCBI wants to integrate their Web authentication mechanism with the Windows desktop authentication process. Tivoli Access Manager for e-business provides the capability of integrating with the Windows desktop authentication. The Access Manager WebSEAL component, which is the reverse proxy resource manager of Tivoli Access Manager, utilizes an SPNEGO token that is based on the Kerberos token generated by the Windows domain controller.

- Additional benefits

  By using the Tivoli Access Manager for e-business for Web single sign-on solution, GCBI also gains additional benefits for its operational security. Tivoli Access Manager for e-business provides a single point of managing and enforcing URL based access control and a central audit mechanism that records all access to a back-end Web server.

4.2 Step-by-step configuration

In this section, we cover all the necessary steps to set up single sign-on between IBM FileNet P8 AE and Tivoli Access Manager. The individual sections are:

- "Prerequisites" on page 69
- "Tivoli Access Manager for e-business configuration" on page 74
- "Configure Access Manager Java runtime" on page 79

- “Enable the TAI++ interceptor on WebSphere Application Server” on page 83
- “Verify the TAI++ interceptor function” on page 90
- “Deploy the P8 AE application” on page 96
- “Additional configuration steps for SPNEGO authentication” on page 118

4.2.1 Prerequisites

In this section, we discuss the following topics:

- “Prerequisites for the installation and configuration” on page 69
- “Software components used in our example” on page 70
- “User registry synchronization” on page 71

Prerequisites for the installation and configuration

We assume that the Tivoli Access Manager for e-business environment is already installed and configured properly (Tivoli Access Manager for e-business installation and configuration should be done by following the product documentation *IBM Tivoli Access Manager for e-business V6.1 Installation Guide*, GC23-6502).

For this scenario, you need to have the following three components of Tivoli Access Manager:

- Policy Server
- WebSEAL
- Authorization Server

We assume that both the FileNet P8 Process Engine and Content Engine are already installed and configured properly. Note that in our scenario, depicted in Figure 3-2, “GCBI P8 infrastructure” on page 59, the Application Engine is installed on a different machine than the other P8 components. Typically the Application Engine is viewed as a presentation-tier resource and it is separated from the business/service-tier resource. For successful communication between the Application Engine and Process Engine/Content Engine, you need to configure LTPA SSO on the WebSphere Application Server where the FileNet P8 AE and P8 PE/CE applications run.

Figure 4-1 shows how LTPA SSO works between the P8 AE and P8 PE/CE components. Two LTPA keys are used in this case, but these keys have to be the same. To synchronize these keys, you should export the key from the WebSphere Application Server that runs P8 PE/CE and import it into the WebSphere Application Server that runs P8 AE.

You can also see how Tivoli Access Manager WebSEAL is providing access control into the P8 AE presentation-tier only. The other P8 resources behind the AE are being secured through regular WebSphere Application Server and P8 means.

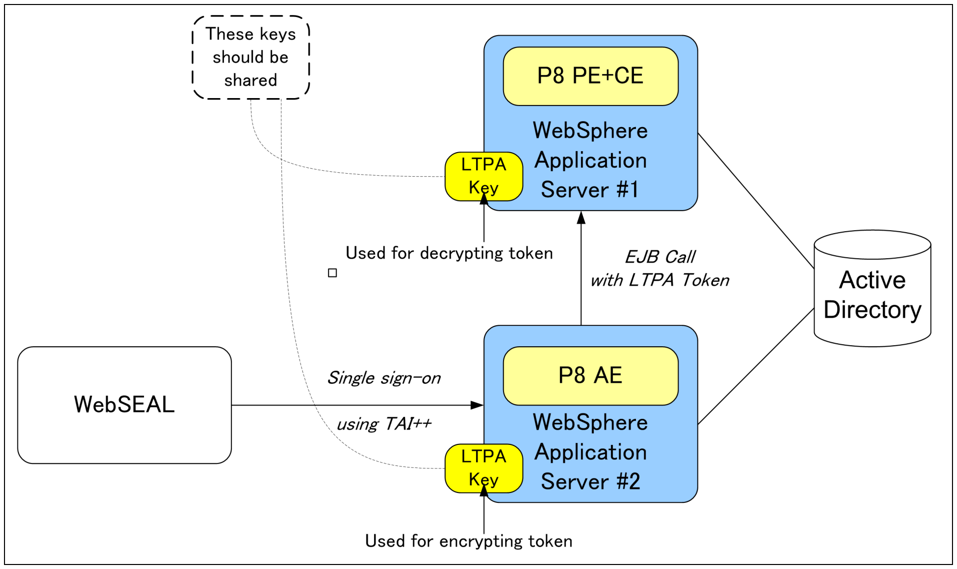

These keys should be shared
P8 PE+CE
WebSphere Application Server #1
LTPA Key
Used for decrypting token
EJB Call with LTPA Token
Active Directory
WebSEAL
Single sign-on using TAI++
P8 AE
WebSphere Application Server #2
LTPA Key
Used for encrypting token

Figure 4-1 Necessity of LTPA SSO between FileNet P8 components

Software components used in our example

We use the following software components in our lab setup:

- Windows Server 2003 Enterprise Edition Service Pack 1

  Access Manager servers, FileNet P8 AE/CE/PE servers, and Domain Controller servers

- Windows XP Service Pack 2

  All client machines

- Tivoli Access Manager for e-business V6.1 GA

  The following components are used in the example

  – Policy Server
  – WebSEAL

   - Authorization Server
- Tivoli Directory Server V6.1 GA
- WebSphere Application Server V6.1.0.11
- FileNet P8 Content Engine V4.01-004
- FileNet P8 Process Engine V4.0.2-001
- FileNet P8 Application Engine V4.0.1-004
- DB2® Enterprise Server V9.5

User registry synchronization

FileNet P8 supports various LDAP compliant directory servers. However, many customers typically choose Microsoft Active Directory, which often serves as the directory for the FileNet P8 solution in customer environments.

In the case of Active Directory, FileNet P8 can use the native representation of users and groups provided by Active Directory. In other words, customers do not need to modify the schema of Active Directory for the FileNet P8 solution.

Tivoli Access Manager for e-business also supports Active Directory as its user registry. However, Tivoli Access Manager for e-business requires schema extensions on Active Directory. Many customers prefer to avoid changing their Active Directory configuration in production. So it is usually a good idea to use IBM Tivoli Directory Server as a registry for Tivoli Access Manager for e-business and use Active Directory as a registry for FileNet P8 products. If there are two or more separate registries involved, administrative impact typically occurs. To reduce this impact, consider implementing an automated directory synchronization mechanism for Active Directory and Tivoli Directory Server.

This automated directory synchronization mechanism monitors for changes that are being administered in Active Directory. It can then administer these changes, for example, adding new users or changing user passwords, into other directories or repositories, such as Tivoli Directory Integrator.

Using the diagram shown in Figure 4-2, we discuss a step-by-step walkthrough about how IBM Tivoli Directory Integrator[1] can help synchronize the identity data between Active Directory and Tivoli Directory Server.

Note: We are taking a look at directory synchronization here because it is often required in a real-world deployment. However, we are not implementing this functionality in our lab setup.

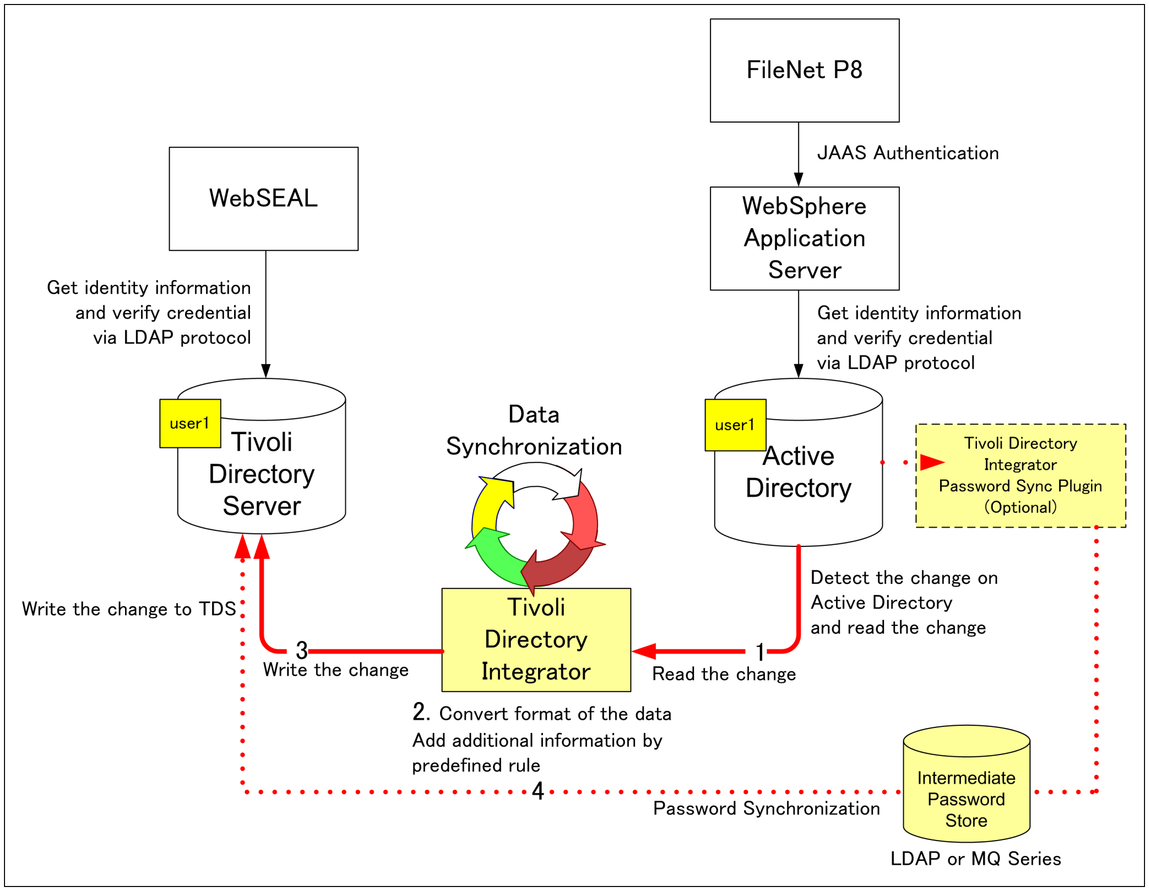

FileNet P8
JAAS Authentication
WebSphere Application Server
WebSEAL
Get identity information and verify credential via LDAP protocol
Get identity information and verify credential via LDAP protocol
user1
Tivoli Directory Server
Data Synchronization
user1
Active Directory
Tivoli Directory Integrator Password Sync Plugin (Optional)
Detect the change on Active Directory and read the change
Write the change to TDS
3
Write the change
Tivoli Directory Integrator
1
Read the change
2. Convert format of the data
Add additional information by predefined rule
4
Password Synchronization
Intermediate Password Store
LDAP or MQ Series

Figure 4-2 Directory synchronization by Tivoli Directory Integrator

[1] If you want to learn more about IBM Tivoli Directory Integrator, refer to the IBM Redbooks publication *Robust Data Synchronization with IBM Tivoli Directory Integrator*, SG24-6164.

The process is as follows:

1. Tivoli Directory Integrator connects to Active Directory using an *Active Directory Changelog Connector*. This connector has the ability to detect the changes happening on Active Directory by watching the *Update Sequence Number* (USN). When a change happens on Active Directory, USN is increased. When the update happens, for example, a user is created by an administrator action, the changelog connector retrieves the *change data* and *change type* (in this case, the change type is *add*) and sends this data to Tivoli Directory Integrator's flow engine, which is called an *AssemblyLine*.
2. Tivoli Directory Integrator now converts the format of data and checks if the data is needed in the AssemblyLine. This action is processed following your predefined Tivoli Directory Integrator configuration.
3. Tivoli Directory Integrator connects to Tivoli Directory Server using an *LDAP Connector*. It writes the data, which has been processed on the AssemblyLine, to the Tivoli Directory Server using the LDAP protocol.
4. Optionally, you can use the *Tivoli Directory Integrator Password Synchronization Plug-in for Active Directory* in order to synchronize passwords between Active Directory and Tivoli Directory server.

   In Active Directory, passwords are encrypted using a 1-way hash algorithm. This means that once you store your password in Active Directory, you cannot decrypt and retrieve a password in plain text format anymore. By using the Tivoli Directory Server *Active Directory Changelog Connector*, you cannot synchronize a password that is stored in Active Directory.

   The Tivoli Directory Integrator Password Synchronization Plug-in is a separate component provided by Tivoli Directory Integrator. If it is installed in an Active Directory environment, it can intercept the password change on a domain controller before the change (the new password) is written to Active Directory. This mechanism enables you to retrieve the password while it is still in plain text format.

   The Tivoli Directory Integrator Password Synchronization Plug-in does not have the capability to directly send the password change to the target data source, so you need to have an LDAP directory server available, such as Tivoli Directory Server or WebSphere embedded MQ series, which is bundled with Tivoli Directory integrator for intermediate data store. You also need to implement a synchronization between the intermediate data store and Tivoli Directory Server. This function can be easily implemented using Tivoli Directory Integrator itself.

4.2.2 Tivoli Access Manager for e-business configuration

The configuration for Tivoli Access Manager for e-business consists of the following steps:

- "Create trust identity for WebSEAL" on page 74
- "Create the junction" on page 78
- "Set dummy password in Webseald-default.conf" on page 79

Create trust identity for WebSEAL

To ensure that the credentials can be successfully passed on by WebSEAL, TAI++ requires trusted identities for WebSEAL on both directories for Access Manager and WebSphere Application Server.

Do these steps:

1. Create an identity for WebSEAL on Access Manager.

   Create a user using **pdadmin** commands, which can be run on the Windows command line. The first command we run creates the user "Websealuser". This user name has to match the user name in Active Directory, which is explained in the next step. The second command enables the user on Access Manager, because a user on Access Manager is disabled by default when it is created by an administrator.

   Example 4-1 shows the execution of the commands.

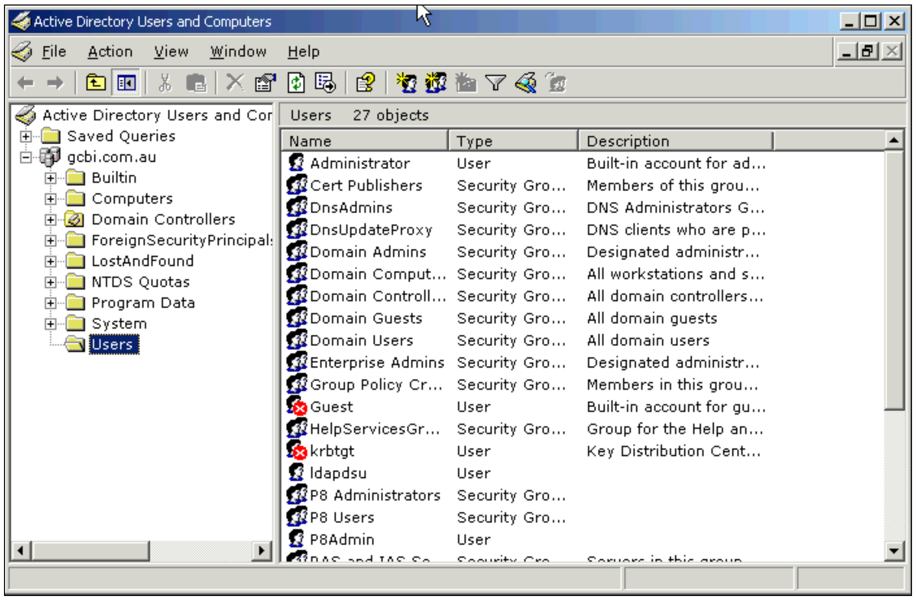
   Example 4-1 pdadmin commands to create a trust identity of WebSEAL

   ```
   pdadmin> user create Websealuser cn=Websealuser,o=gcbi,c=au
   Websealuser Websealuser password0
   pdadmin> user modify Websealuser account-valid yes
   ```

2. Create an identity for WebSEAL on Active Directory.

   Create a user using the Microsoft Management Console on the domain controller machine. In our case, the user logon name is "Websealuser", which matches the user name on Tivoli Access Manager.

To start the Microsoft Management Console, select **Start ∅ All Programs ∅ Administrative Tools ∅ Active Directory Users and Computers**. You should get the window shown in Figure 4-3.


Active Directory Users and Computers
File Action View Window Help

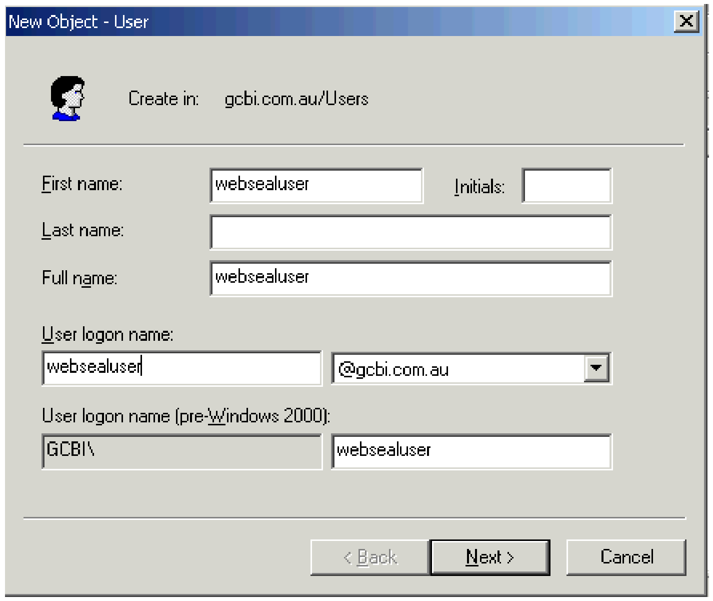
Active Directory Users and Cor
- Saved Queries
- gcbi.com.au
  - Builtin
  - Computers
  - Domain Controllers
  - ForeignSecurityPrincipal:
  - LostAndFound
  - NTDS Quotas
  - Program Data
  - System
  - Users

Users 27 objects

Name	Type	Description
Administrator	User	Built-in account for ad...
Cert Publishers	Security Gro...	Members of this grou...
DnsAdmins	Security Gro...	DNS Administrators G...
DnsUpdateProxy	Security Gro...	DNS clients who are p...
Domain Admins	Security Gro...	Designated administr...
Domain Comput...	Security Gro...	All workstations and s...
Domain Controll...	Security Gro...	All domain controllers...
Domain Guests	Security Gro...	All domain guests
Domain Users	Security Gro...	All domain users
Enterprise Admins	Security Gro...	Designated administr...
Group Policy Cr...	Security Gro...	Members in this grou...
Guest	User	Built-in account for gu...
HelpServicesGr...	Security Gro...	Group for the Help an...
krbtgt	User	Key Distribution Cent...
ldapdsu	User	
P8 Administrators	Security Gro...	
P8 Users	Security Gro...	
P8Admin	User	
RAS and IAS Se...	Security Gro...	Servers in this group

Figure 4-3 Microsoft Management Console window

3. Right-click the **Users** folder in the left pane of the window and select **New ∅ User**. You should get the window shown in Figure 4-4.



Create in: gcbi.com.au/Users

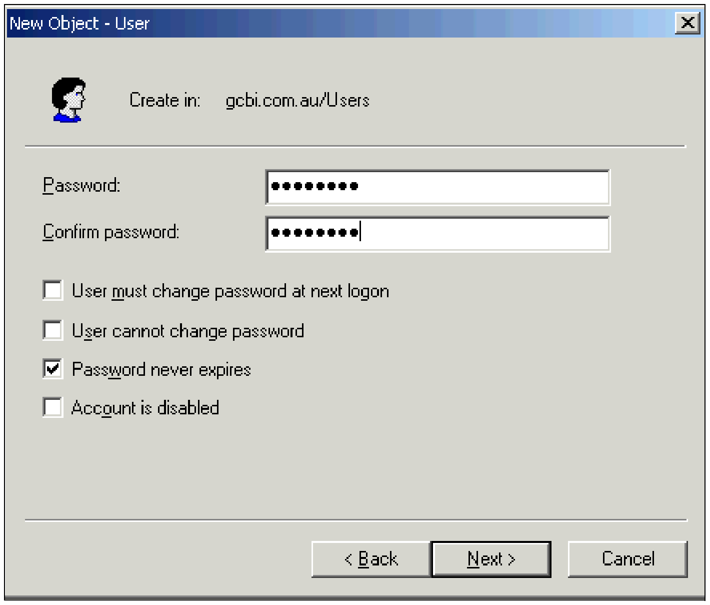
First name: websealuser
Initials:
Last name:
Full name: websealuser

User logon name:
websealuser @gcbi.com.au

User logon name (pre-Windows 2000):
GCBI\ websealuser

Figure 4-4   Create a user for WebSEAL on Active Directory 1

4. In our case, User logon name and Full name should be “Websealuser”. After filling in the user name, click **Next**. You should get the window shown in Figure 4-5.


New Object - User

Create in: gcbi.com.au/Users

Password: ●●●●●●●●●

Confirm password: ●●●●●●●●●

☐ User must change password at next logon

☐ User cannot change password

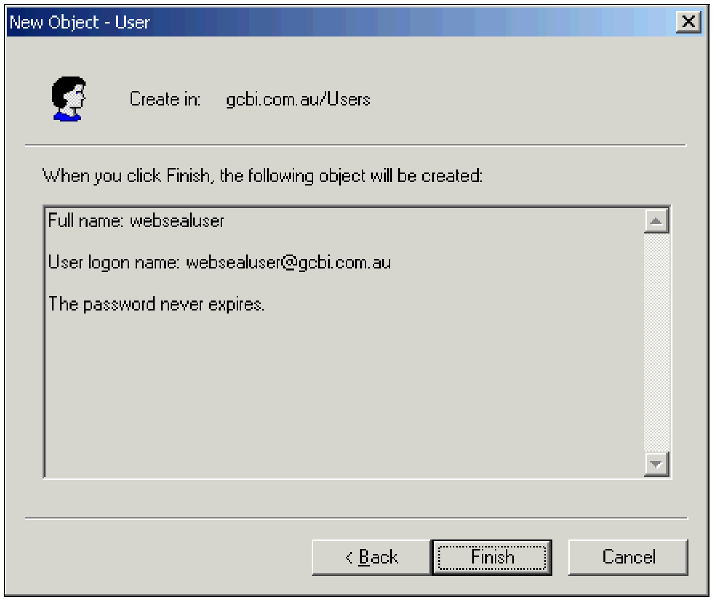
☑ Password never expires

☐ Account is disabled

< Back | Next > | Cancel

Figure 4-5 Create a user for WebSEAL on Active Directory 2

5. Type the appropriate password in both the Password field and the Confirm password field. This password should match that *dummy password* setting on Access Manager WebSEAL configuration file which is mentioned in the configuration step “Set dummy password in Webseald-default.conf” on page 79. Then uncheck **User must change password at next login** and check **Password never expires**, because this user is used by the system and not by a human. Then click **Next**. You should get the window shown in Figure 4-6.



When you click Finish, the following object will be created:

Full name: websealuser

User logon name: websealuser@gcbi.com.au

The password never expires.

Figure 4-6 Create a user for WebSEAL on Active Directory 3

6. Confirm the settings. If they are okay, click **Finish**.

Create the junction

Next, we need to create a junction to the P8 Application Engine using the server task sub command with the `pdadmin` command.

To work with the P8 AE, the *transparent path junction* is needed. For standard junctions, a link to a resource on a back-end junctioned server can only succeed if the URL in the request received by WebSEAL contains the identity of the junction. Junctions use both default and optional filtering solutions to force URLs found in HTML response pages to appear correct when viewed as a part of WebSEAL’s single host document space. We call this function, which corrects URLs in the contents, as *URL filtering*. Unfortunately, URL filtering does not work correctly and an improper rendering of the contents occurs. The *transparent path* *junction* allows us to avoid this error. To create a transparent junction, you need the `-x` option when you create a junction. For the TAI++ interceptor, you need to specify the -c iv_creds and -b supply options when creating the junction. The following is an example of the **`pdadmin`** command that creates the junction to the P8 AE:

```
pdadmin> s t create -t tcp -h rbp8ae -p 9080 -c iv_creds -b supply -x
/Workplace
```

Set dummy password in Webseald-default.conf

The *dummy password* is used to verify that the trusted identity passed by WebSEAL is correct. The junction option -b supply used in the previous step defines the use of the dummy password on the HTTP BA Header that is passed to the back-end Web server.

To set the dummy password, you have to modify the `Webseald-default.conf` file, which is the main configuration file for WebSEAL. On Windows, Webseald-default.conf is located in the `C:\Program Files\Tivoli\PDWeb\etc` directory. Open `Webseald-default.conf` using a text editor and find the *junction* section. There is an entry for the *basicauth-dummy-passwd* parameter in this section. Set the appropriate password string to the following value.

```
basicauth-dummy-passwd = password0
```

Save the change of configuration file. Then restart WebSEAL to make your change on configuration file valid.

4.2.3 Configure Access Manager Java runtime

You have to perform two steps in order to configure thee Access Manager Java runtime on WebSphere Application Server:

- "Access Manager Java runtime configuration" on page 79
- "Creating Access Manager server instance" on page 82

Access Manager Java runtime configuration

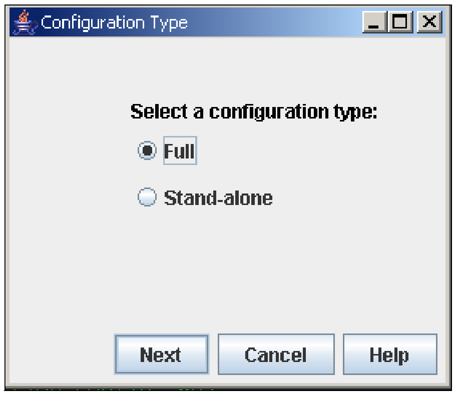
In order to call the Tivoli Access Manager authorization API within a Java Virtual Machine (JVM™), we need to configure the Access Manager Java runtime on the WebSphere Application Server JVM. In this step, we create a configuration file that specifies the Tivoli Access Manager Policy Server host name, port, and domain name.

Do these steps:

1. To begin the Tivoli Access Manager Java runtime configuration, run the following command:

```
C:\Program Files\IBM\WebSphere\AppServer\bin>java
-Dpd.home="%WAS_HOME%\java\jre\PolicyDirector" -cp
"%WAS_HOME%\java\jre\lib\ext\PD.jar" com.tivoli.pd.jcfg.PDJrteCfg
-action config -interactive -was
```

2. After you have entered the command, the following window is displayed (Figure 4-7). Confirm that **Full** is checked as the configuration type and click **Next**.

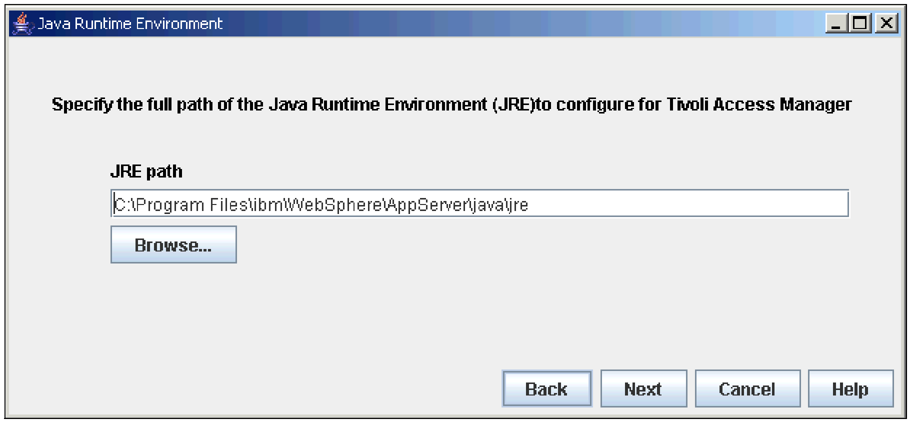

Configuration Type

Select a configuration type:

[✓ Full] Stand-alone

Figure 4-7 Access Manager Java runtime configuration type selection

3. Next, you are asked to provide the JRE™ path for the Tivoli Access Manager Java runtime. Confirm that the JRE path that is displayed matches the WebSphere Application Server's JRE path shown in Figure 4-8 on page 81. Click **Next** to continue.


Java Runtime Environment

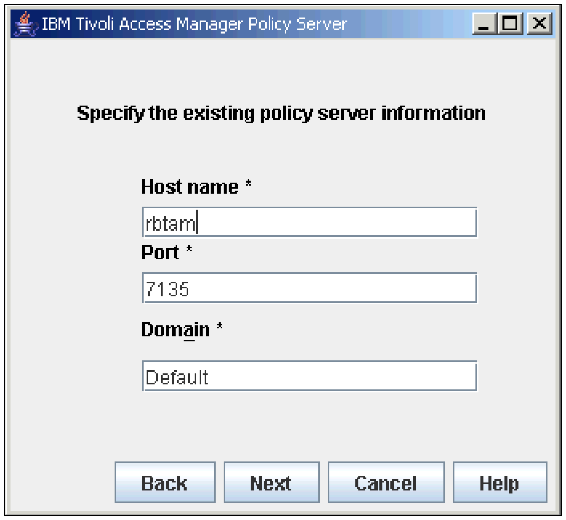
Specify the full path of the Java Runtime Environment (JRE)to configure for Tivoli Access Manager

JRE path

C:\Program Files\ibm\WebSphere\AppServer\java\jre

Figure 4-8 JRE path for Access Manager Java runtime

4. Next, you need to provide the Access Manager Policy Server information. According to your Access Manager installation, enter your Access Manager Policy Server host name, as shown in Figure 4-9. In our case, Access Manager Policy Server runs on an rbtam machine. By default, the port number is 7135 and the Access Manager domain is named Default. After setting the host name, port number, and domain name, click **Next**.


IBM Tivoli Access Manager Policy Server

Specify the existing policy server information

Host name *

rbtam

Port *

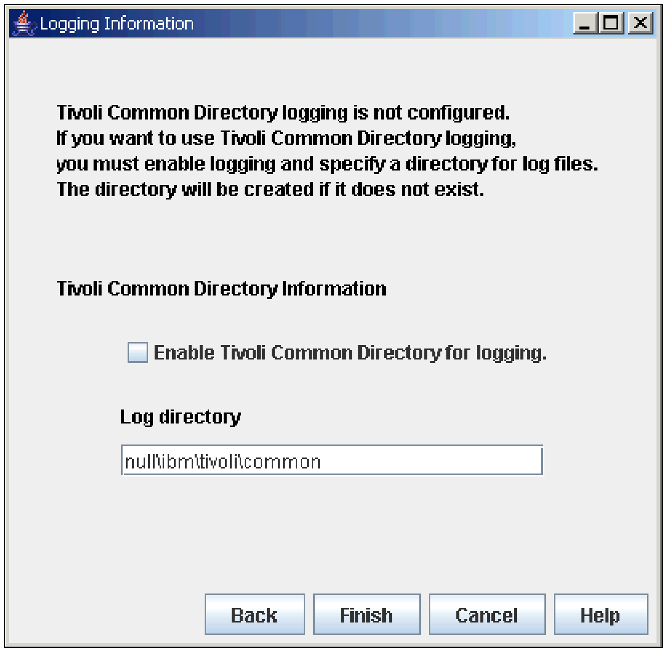
7135

Domain *

Default

Figure 4-9 Specify Access Manager Policy Server information

5. Tivoli Common Directory logging is an optional setting. In our example, we do not use Tivoli Common Directory logging. Click **Finish**, as shown in Figure 4-10.



Tivoli Common Directory logging is not configured.
If you want to use Tivoli Common Directory logging,
you must enable logging and specify a directory for log files.
The directory will be created if it does not exist.

Tivoli Common Directory Information

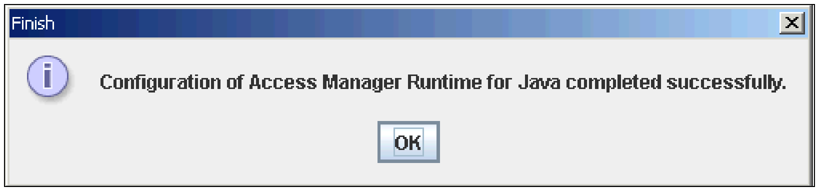
☐ Enable Tivoli Common Directory for logging.

Log directory

null\ibm\tivoli\common

Figure 4-10 Access Manager Java runtime configuration

6. You should see the window shown in Figure 4-11. The Access Manager Java runtime configuration is completed.



Configuration of Access Manager Runtime for Java completed successfully.

Figure 4-11 Access Manager Java runtime configuration completed

Creating Access Manager server instance

The `svrsslcfg` command is used to create an instance of a Tivoli Access Manager access enforcement point. To use the TAI++ function on WebSphere Application Server that runs the FileNet P8 AE, WebSphere Application Server must be configured as an access enforcement point. Here is a sample command to create an instance of Tivoli Access Manager for WebSphere Application Server:

```
C:\Program Files\IBM\WebSphere\bin> java com.tivoli.pdjcfg.SvrSslcfg
-action config -admin_id sec_master -admin_pwd filenet -appsvr_id
rbp8ae -appsvr_pwd password0 -port 7135 -mode remote -host rbp8ae
-policysvr rbtam:7135:1 -authzsvr rbtam:7136:1 -cfg_file
"%WAS_HOME%\java\jre\PdPerm.properties" -keyfile
"%WAS_HOME%\java\jre\lib\security\PdPerm.ks" -cfg_action create
```

Note: Make sure the command is a single line.

If the configuration successfully finishes, you should get the following message:

```
The configuration completed successfully.
```

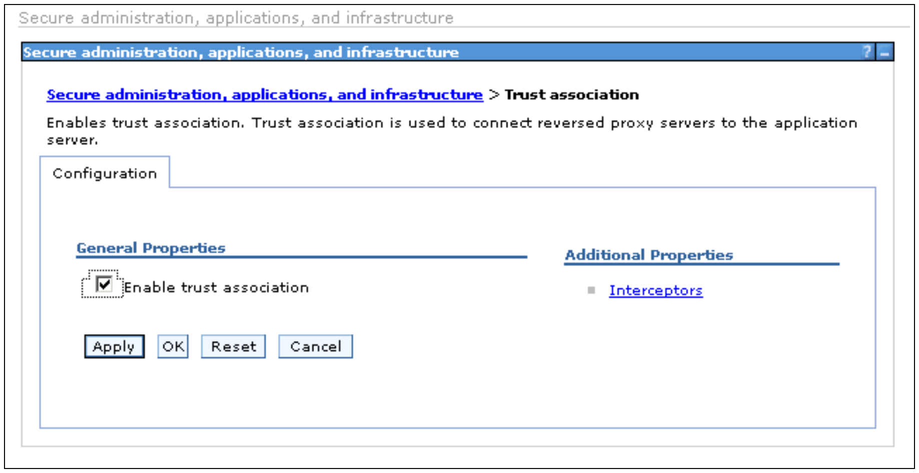
4.2.4 Enable the TAI++ interceptor on WebSphere Application Server

After finishing the Access Manager Java runtime configuration on WebSphere Application Server, we have to configure the TAI++ interceptor.

Do these steps:

1. First, access the WebSphere Application Server Integrated Solutions Console using `https://<your WebSphere server hostname:9043/ibm/console>`. Then select **Security tab ∅ Secure administration, applications, and infrastructure ∅ Trust Association**, as shown in Figure 4-12.

   Check **Enable trust association** and press **OK**.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association

Enables trust association. Trust association is used to connect reversed proxy servers to the application server.

Configuration

General Properties

[✓] Enable trust association

Additional Properties

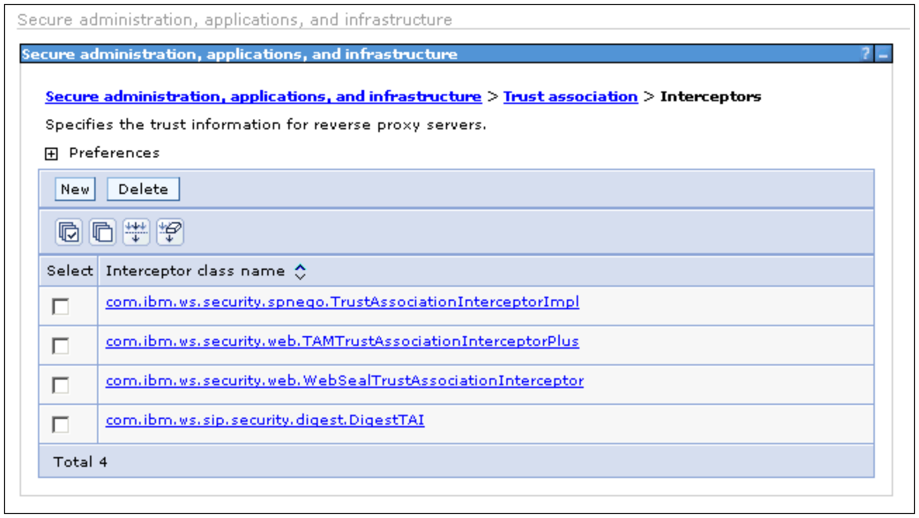
Interceptors

Figure 4-12 Enable trust association

2. After saving your configuration, click the **Interceptors** in order to modify the TAI interceptor setting.

   Figure 4-13 shows a list of TAI interceptors that is provided by default.

   We use com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus from the list, which is the interceptor module for TAI++.



Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association > Interceptors

Specifies the trust information for reverse proxy servers.

Preferences

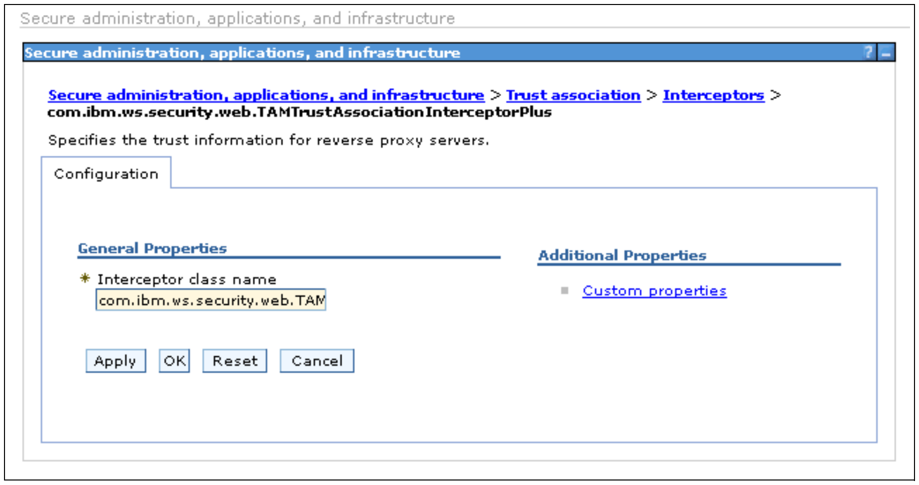
Select	Interceptor class name
☐	com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl
☐	com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus
☐	com.ibm.ws.security.web.WebSealTrustAssociationInterceptor
☐	com.ibm.ws.sip.security.digest.DigestTAI

Total 4

Figure 4-13 List of built-in interceptors

3. It is not mandatory, but it is a good idea to delete the interceptor modules other than the one for TAI++. Deleting unused TAI interceptor modules can reduce meaningless error messages in WebSphere Application Server log files.

   Now click **com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus** in the list. You should see the window shown in Figure 4-14 on page 85.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association > Interceptors > com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus

Specifies the trust information for reverse proxy servers.

Configuration

General Properties

* Interceptor class name: com.ibm.ws.security.web.TAM

Additional Properties

- Custom properties

Figure 4-14 General properties for TAI++

4. Select **Custom properties**. You now see the window for custom properties. At this moment, there are no properties defined on this window. You need to create each property by clicking the **New** button.

5. For TAI++ on WebSphere Application Server V6.1 there are nine custom properties, which are shown in Table 4-1). In order for TAI++ to work, you have to specify a value for com.ibm.websphere.security.webseal.loginId. This trusted ID for WebSEAL has been already created, as shown in “Create trust identity for WebSEAL” on page 74.

Note: In the case of your production system, you can let the TAI++ module verify the identity of WebSEAL using information in the HTTP header sent by WebSEAL. This function enables a more secure single sign-on context between WebSEAL and WebSphere Application Server. The verification is performed by using the HTTP through the header and host header.

Table 4-1 TAI++ custom properties

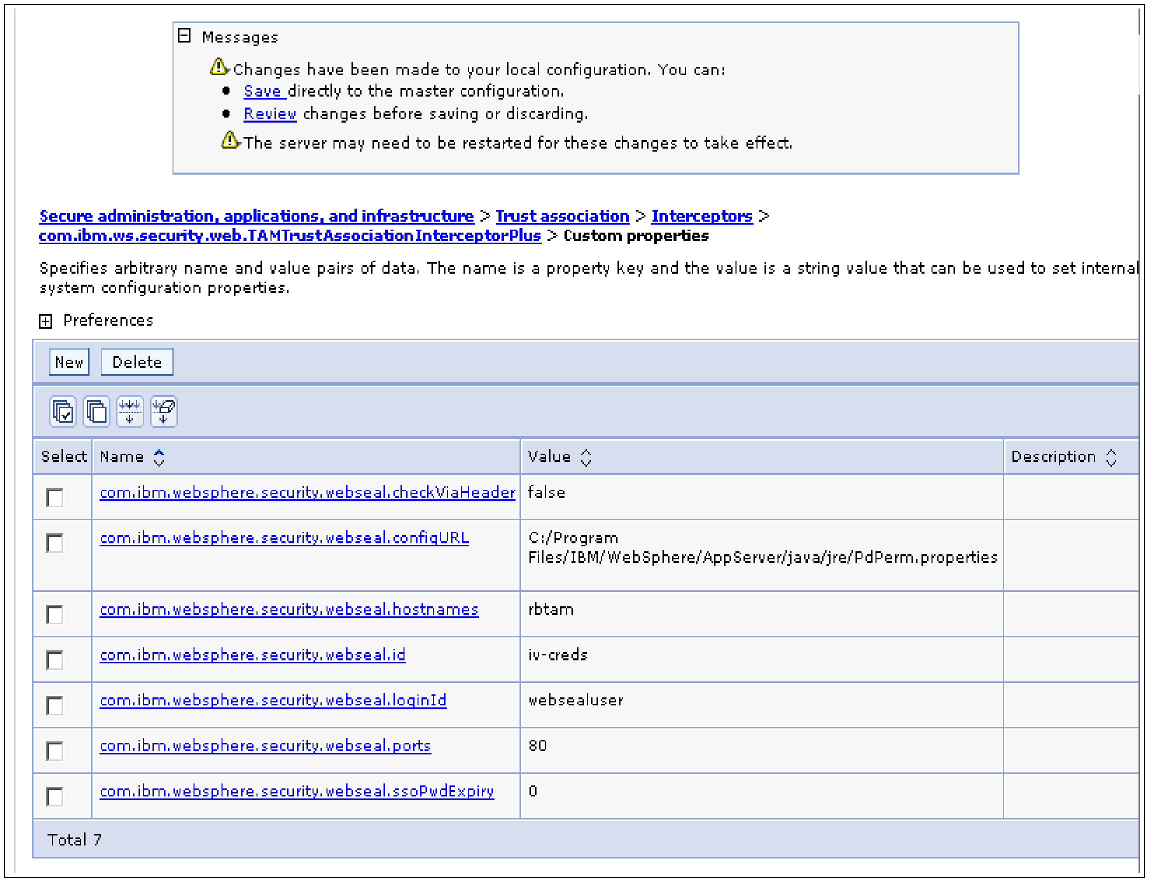
Option	Description	Mandatory?
com.ibm.websphere.security.webseal.checkViaHeader	You can configure TAI so that the Via header can be ignored when validating trust for a request. Set this property to false if none of the hosts in the Via header need to be trusted. When set to false, you do not need to set the trusted host names and host ports properties. The only mandatory property to check when the Via header is false is com.ibm.WebSphere.security.Webseal.loginId. The default value of the checkViaHeader property is false.	No
com.ibm.websphere.security.webseal.loginId	The WebSEAL trusted user created with the Creating a trusted user account function in Tivoli Access Manager The format of the user name is the short name representation.	Yes
com.ibm.websphere.security.webseal.id	A comma-separated list of headers that exists in the request. If all of the configured headers do not exist in the request, trust cannot be established. The default value for the ID property is iv-creds. Any other values set in WebSphere Application Server are added to the list along with iv-creds, separated by commas.	No
com.ibm.websphere.security.webseal.hostnames	The property specifies the host names (case sensitive) that are trusted and expected in the request header. Requests arriving from unlisted hosts might not be trusted. If the checkViaHeader property is not set or is set to false then the trusted host names property has no influence. If the checkViaHeader property is set to true, and the trusted host names property is not set, TAI initialization fails.	No

Option	Description	Mandatory?
com.ibm.websphere.secur ity.webseal.ports	This property is a comma-separated list of trusted host ports. Requests that arrive from unlisted ports might not be trusted. If the checkViaHeader property is not set, or is set to false, this property has no influence. If the checkViaHeader property is set to true, and the trusted host ports property is not set in WebSphere Application Server, the TAI initialization fails.	No
com.ibm.websphere.secur ity.webseal.viaDepth	A positive integer that specifies the number of source hosts in the via header to check for trust. By default, every host in the via header is checked, and if any host is not trusted, trust cannot be established. The viaDepth property is used when only some of the hosts in the via header have to be trusted. The setting indicates the number of hosts that are required to be trusted.	No
com.ibm.websphere.secur ity.webseal.ssoPwdExiry	After trust is established for a request, the single sign-on user password is cached, eliminating the need to have the TAI re-authenticate the single sign-on user with Tivoli Access Manager for every request. You can modify the cache timeout period by setting the single sign-on password expiry property to the required time in seconds. If the password expiry property is set to 0, the cached password never expires. The default value for the password expiry property is 600.	No
com.ibm.websphere.secur ity.webseal.ignoreProxy	This property can be used to tell the TAI to ignore proxies as trusted hosts. If set to true, the comments field of the hosts entry in the via header is checked to determine if a host is a proxy. Remember that not all proxies insert comments in the via header indicating that they are proxies. The default value of the ignoreProxy property is false. If the checkViaHeader property is set to false, then the ignoreProxy property has no influence in establishing trust.	No

Option	Description	Mandatory?
com.ibm.websphere.security.webseal.configURL	For the TAI to establish trust for a request, it requires that the SvrSslCfg run for the Java Virtual Machine on the Application Server and result in the creation of a properties file. If this properties file is not at the default URL, which is `file://java.home/PdPerm.properties`, the correct URL of the properties file must be set in the configuration URL property. If this property is not set, and the SvrSslCfg-generated properties file is not in the default location, the TAI initialization fails. The default value for the config URL property is `file://${WAS_INSTALL_ROOT}/java/jre/PdPerm.properties`.	No

Note: The property names are case sensitive. The TAI interceptor does not start if you use misspelled property names. Be careful when typing this configuration. We strongly recommend that you re-check your configuration before saving.

6. In our test case, we set some properties, as shown in Figure 4-15 on page 89.


Messages

Changes have been made to your local configuration. You can:
- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Secure administration, applications, and infrastructure > Trust association > Interceptors > com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus > Custom properties

Specifies arbitrary name and value pairs of data. The name is a property key and the value is a string value that can be used to set internal system configuration properties.

Select	Name	Value	Description
☐	com.ibm.websphere.security.webseal.checkViaHeader	false	
☐	com.ibm.websphere.security.webseal.configURL	C:/Program Files/IBM/WebSphere/AppServer/java/jre/PdPerm.properties	
☐	com.ibm.websphere.security.webseal.hostnames	rbtam	
☐	com.ibm.websphere.security.webseal.id	iv-creds	
☐	com.ibm.websphere.security.webseal.loginId	websealuser	
☐	com.ibm.websphere.security.webseal.ports	80	
☐	com.ibm.websphere.security.webseal.ssoPwdExpiry	0	

Total 7

Figure 4-15 Example of custom properties for TAI++ interceptor

7. After setting all properties and their values, press **OK** and then **Save**.
8. Restart WebSphere Application Server and see if the TAI++ interceptor is successfully initialized on the startup sequence.

You can check the messages in the systemout.log, which is the standard out log of WebSphere Application Server. An example is provided in Example 4-2.

Example 4-2 SystemOut.log produced by WebSphere Application Server

```
[9/25/08 17:16:14:000 PDT] 0000000a TrustAssociat A  SECJ0121I:
Trust Association Init class
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus loaded
successfully
[9/25/08 17:16:17:171 PDT] 0000000a TAMTrustAssoc I
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlusinitialize
(Properties) The Trust Association Interceptor component of embedded
Tivoli Access Manager has been initialized.
[9/25/08 17:16:17:171 PDT] 0000000a TrustAssociat A  SECJ0122I:
Trust Association Init Interceptor signature: $Id: @(#)64 1.7
src/pdwas/com/ibm/ws5/security/Web/TAMTrustAssociationInterceptorPlu
s.java, amemb.jacc.was, amemb600, 070522a 07/05/21 19:49:56 @(#) $
```

4.2.5 Verify the TAI++ interceptor function

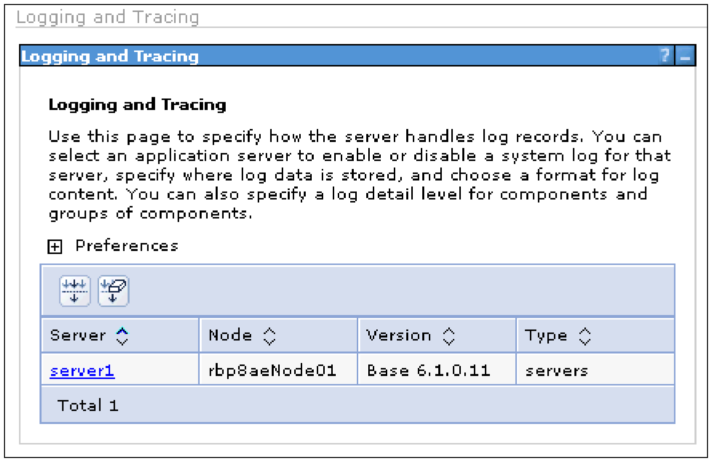
Now is a good time to verify if the TAI++ interceptor works properly with Access Manager.

We can use the *snoop servlet* to verify if the TAI++ interceptor works correctly. A snoop servlet is installed on WebSphere Application Server by default. In our example, the URL for the snoop servlet is `http://rbtam/Workplace/snoop`. In some cases, TAI++ seems to work properly, but TAI++ is not the sole factor in play. For example, if a dummy password for the `Webseald-default.conf` file is the same as an actual user's password in the WebSphere registry, and TAI++ does not work correctly because of a misconfiguration, WebSphere authentication can only be done by an HTTP BA header. You can verify if your authentication is done successfully by the TAI++ interceptor or BA header by running a trace and looking at the messages in the trace log.

How to obtain TAI++ trace

In order to obtain a TAI++ trace, follow these steps:

1. Open the WebSphere Integrated Solutions Console (ISC). Select the **Troubleshooting** tab, and then select **Logs and Trace** (see Figure 4-16 on page 91).


Logging and Tracing

Logging and Tracing

Logging and Tracing

Use this page to specify how the server handles log records. You can select an application server to enable or disable a system log for that server, specify where log data is stored, and choose a format for log content. You can also specify a log detail level for components and groups of components.

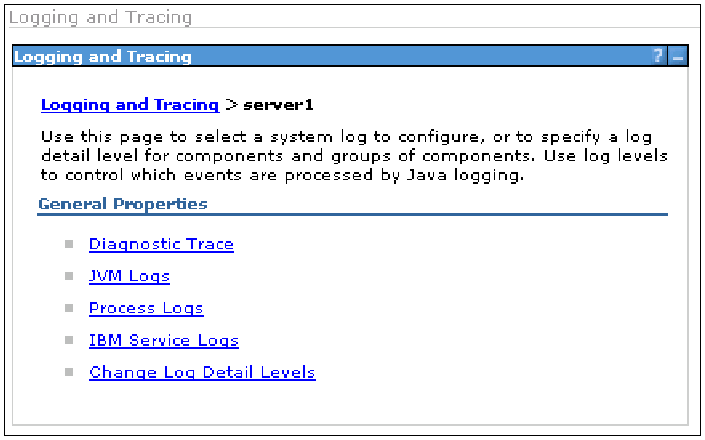
⊞ Preferences

Server	Node	Version	Type
server1	rbp8aeNode01	Base 6.1.0.11	servers

Total 1

Figure 4-16 Logging and Tracing window on the WebSphere ISC

2. Click the **server1** link. You should see the window shown in Figure 4-17.


Logging and Tracing

Logging and Tracing

Logging and Tracing > **server1**

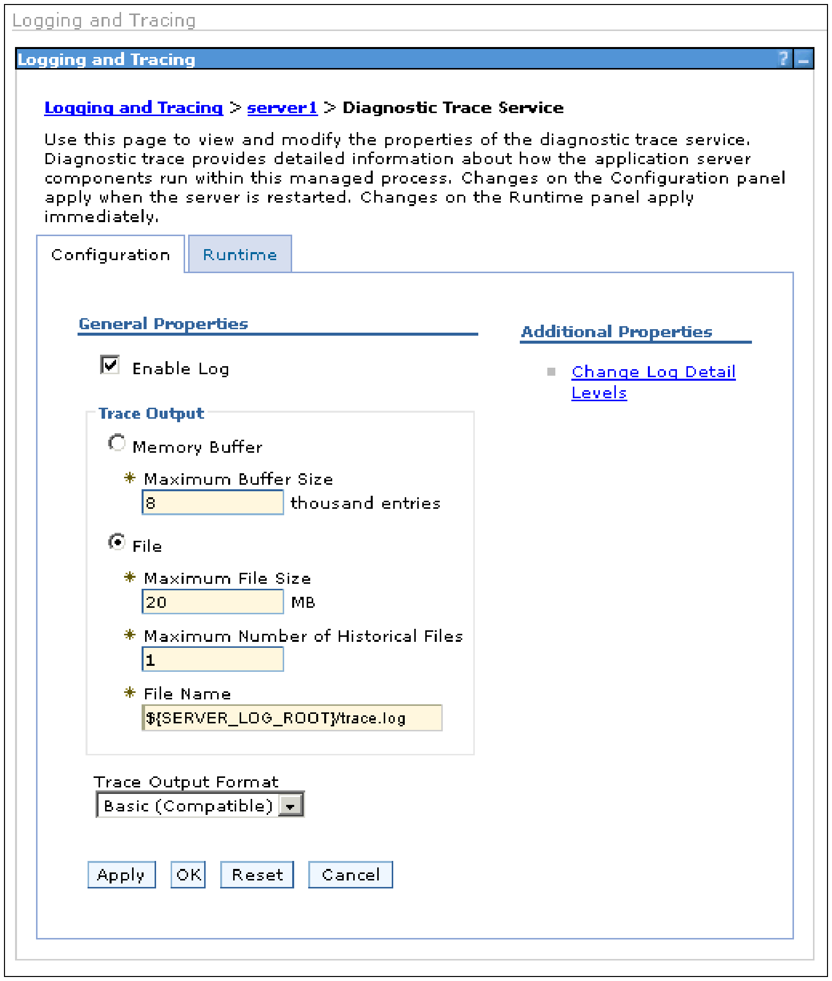
Use this page to select a system log to configure, or to specify a log detail level for components and groups of components. Use log levels to control which events are processed by Java logging.

General Properties

- Diagnostic Trace
- JVM Logs
- Process Logs
- IBM Service Logs
- Change Log Detail Levels

Figure 4-17 Detail menu of Logging and Tracing

3. Click **Diagnostic Trace**. You should see the window shown in Figure 4-18.



Logging and Tracing

Logging and Tracing

Logging and Tracing > server1 > Diagnostic Trace Service

Use this page to view and modify the properties of the diagnostic trace service. Diagnostic trace provides detailed information about how the application server components run within this managed process. Changes on the Configuration panel apply when the server is restarted. Changes on the Runtime panel apply immediately.

Configuration | Runtime

General Properties

[✓] Enable Log

Trace Output

( ) Memory Buffer

* Maximum Buffer Size
8 thousand entries

(●) File

* Maximum File Size
20 MB

* Maximum Number of Historical Files
1

* File Name
${SERVER_LOG_ROOT}/trace.log

Trace Output Format
Basic (Compatible)

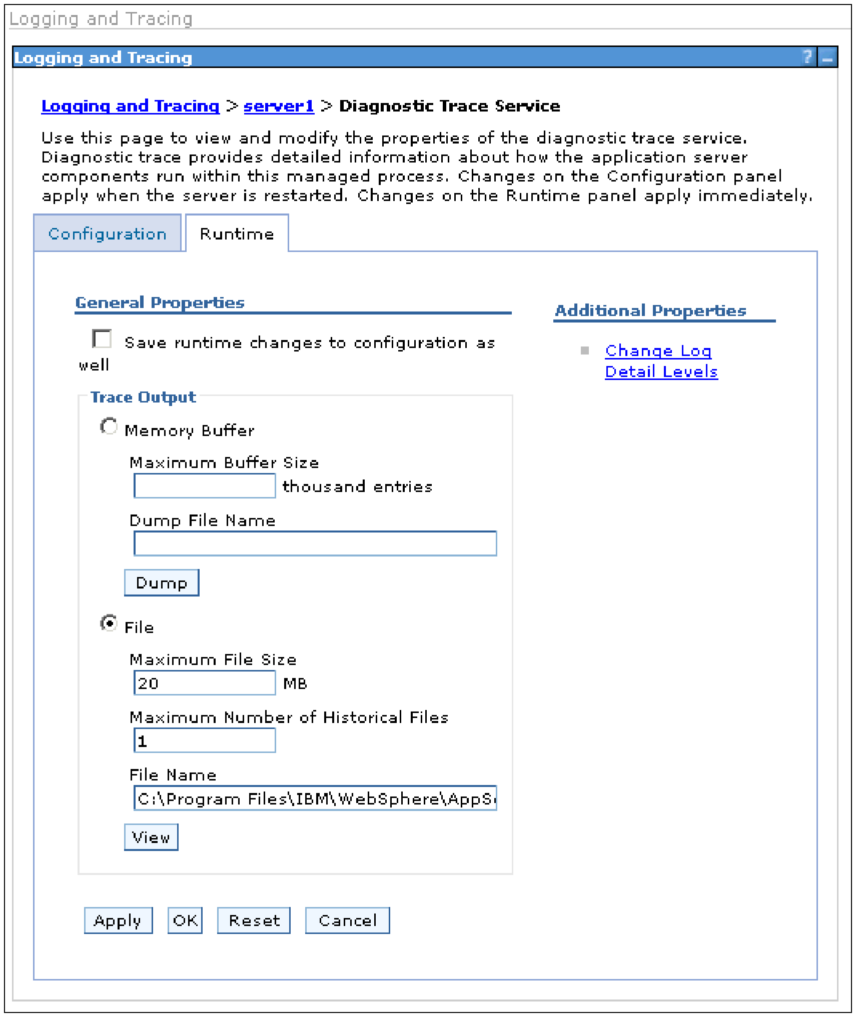
Apply | OK | Reset | Cancel

Additional Properties

- Change Log Detail Levels

Figure 4-18 Diagnostic trace service configuration

4. In our case, we only need a temporary trace and not a permanent one, so press the **Runtime** tab. You should see the window shown in Figure 4-19.


Logging and Tracing

Logging and Tracing

Logging and Tracing > server1 > Diagnostic Trace Service

Use this page to view and modify the properties of the diagnostic trace service. Diagnostic trace provides detailed information about how the application server components run within this managed process. Changes on the Configuration panel apply when the server is restarted. Changes on the Runtime panel apply immediately.

Configuration | Runtime

General Properties

☐ Save runtime changes to configuration as well

Trace Output

○ Memory Buffer

Maximum Buffer Size
[ ] thousand entries

Dump File Name
[ ]

Dump

[✓ File]

Maximum File Size
20 MB

Maximum Number of Historical Files
1

File Name
C:\Program Files\IBM\WebSphere\AppS

View

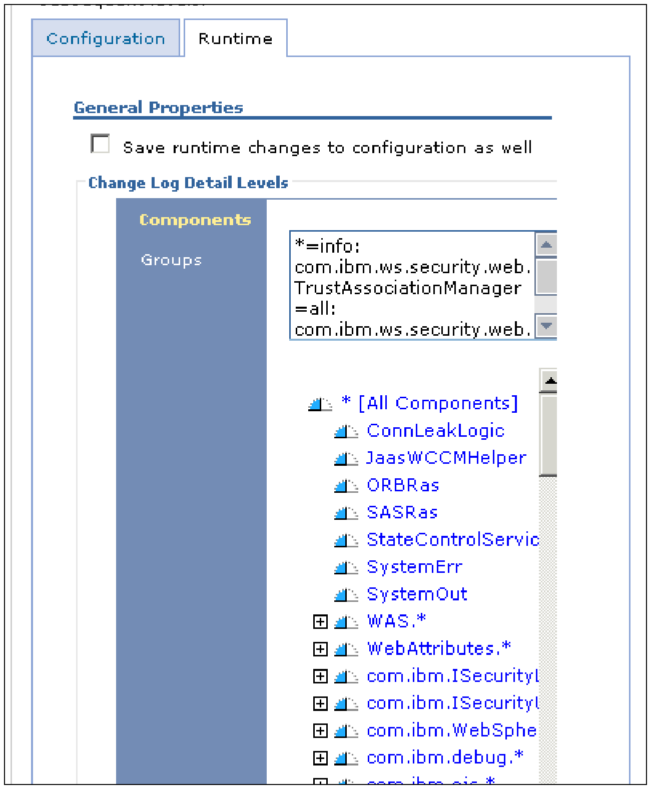
Apply | OK | Reset | Cancel

Additional Properties

- Change Log Detail Levels

Figure 4-19 Runtime setting for Diagnostic Trace Service

5. Click **Change Log Detail Levels** on the right side of the window. You should see the window shown in Figure 4-20. Make sure the Runtime tab is active in the window.



```
*=info:
com.ibm.ws.security.web.
TrustAssociationManager
=all:
com.ibm.ws.security.web.
```

Figure 4-20 Log detail level setting

6. You should enable **com.ibm.ws.security.web.TrustAssociationManager=all** and **com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus=all**. You can use the navigation in the window to pick the target of your trace, then right-click the target and choose all logging and trace on it.

   After picking the appropriate trace target in this window, click **Apply**.

   Now you can access the snoop servlet through WebSEAL. Open the your browser and access the site `http://<your tam host name>/Workplace/snoop`. You can see the trace log file produced by your access to the snoop servlet. The trace log file is located in the `<WAS HOME>/logs/server1/trace.log` directory.

In Example 4-3 we show parts of a trace log gathered in our test environment. In this case, we used the P8admin user to access Access Manager. You can see the following message:

```
TAMTrustAssoc >
com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus
buildSubject(PDPrincipal print, String cred) ENTRY P8Admin
```

This means that TAI++ successfully received the identity information and created the PDPrincipal subject for the P8Admin user.

Example 4-3 TAI++ trace example

```
[9/30/08 11:25:48:500 PDT] 00000023 TAMTrustAssoc >
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
validateEstablishedTrust(HttpServletRequest) ENTRY
[9/30/08 11:25:48:515 PDT] 00000023 TAMTrustAssoc 1
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlusvalidateEs
tablishedTrust(HttpServletRequest) SSO password is cached.
Attempting to authenticate Websealuser
[9/30/08 11:25:48:515 PDT] 00000023 TAMTrustAssoc 1
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlusvalidateEs
tablishedTrust(HttpServletRequest) Authentication succeeded.
[9/30/08 11:25:48:515 PDT] 00000023 TAMTrustAssoc <
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
validateEstablishedTrust(HttpServletRequest) RETURN
[9/30/08 11:25:48:515 PDT] 00000023 TAMTrustAssoc >
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
createPDPrincipal(String) ENTRY
[9/30/08 11:25:48:578 PDT] 00000023 TAMTrustAssoc <
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
createPDPrincipal(String) RETURN
[9/30/08 11:25:48:593 PDT] 00000023 TAMTrustAssoc >
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
buildSubject(PDPrincipal prin, String cred) ENTRY P8Admin
[9/30/08 11:25:48:593 PDT] 00000023 TAMTrustAssoc <
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
buildSubject(PDPrincipal, String) RETURN
[9/30/08 11:25:48:593 PDT] 00000023 TAMTrustAssoc <
com.ibm.ws.security.Web.TAMTrustAssociationInterceptorPlus
negotiateValidateandEstablishTrust(HttpServletRequest,
HttpServletResponse) RETURN
```

4.2.6 Deploy the P8 AE application

To enable TAI++ single sign-on for use within the P8 AE application, we need to deploy the P8 AE application on top of the TAI++ enabled WebSphere Application Server. The following steps need to be done:

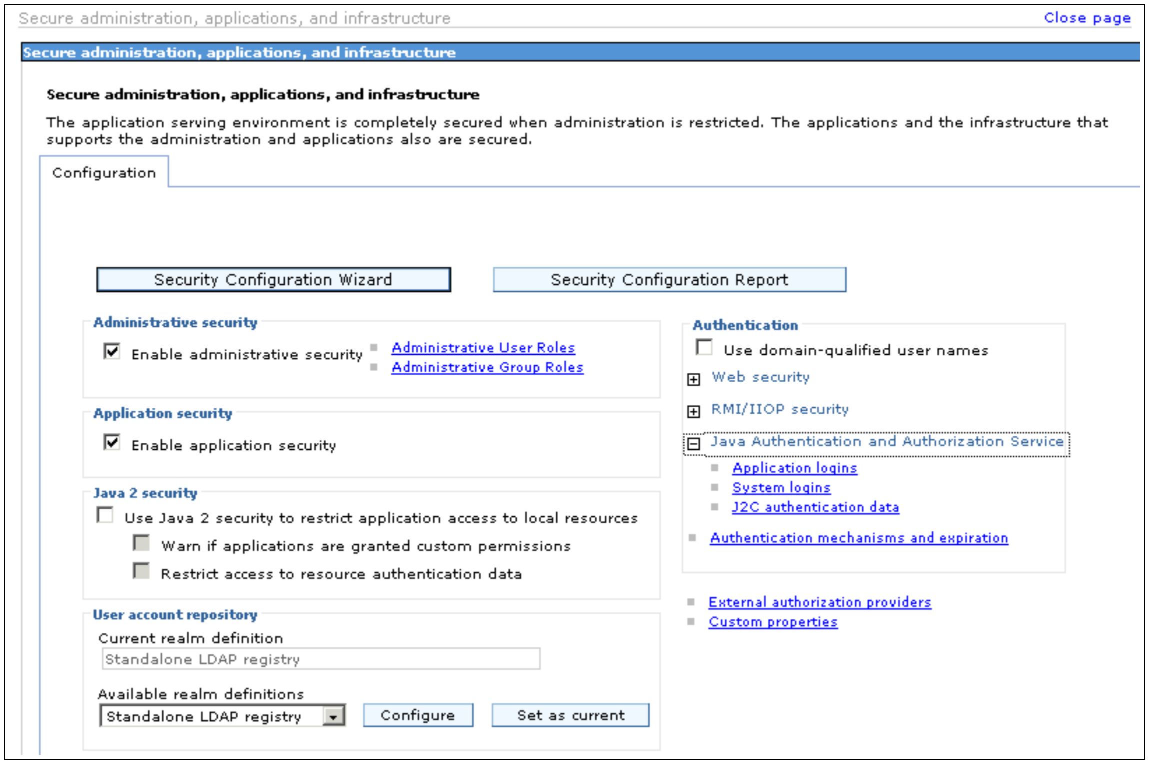
- “Configure Java Authentication and Authorization Service login” on page 96
- “Modify deployment descriptors for the P8 Application Engine” on page 101
- “Create WAR and EAR file packages” on page 107
- “Deploy the application using the WebSphere ISC” on page 108
- “Security mapping for P8 Workplace application” on page 113

Configure Java Authentication and Authorization Service login

Java Authentication and Authorization Service (JAAS) is a plug-in authentication mechanism for Java. The FileNet P8 Application Engine uses a specific authentication mechanism and module, so you need to configure the module on WebSphere Application Server Integrated Solutions Console.

Do these steps:

1. Open the WebSphere Integrated Solutions Console (ISC) by entering the correct URL. In our example, the URL for ISC is `https://rbp8ae:9060/ibm/console`. Select **Security** ∅ **Secure administration, applications, and infrastructure**, as shown in Figure 4-21.



Secure administration, applications, and infrastructure

Close page

Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

The application serving environment is completely secured when administration is restricted. The applications and the infrastructure that supports the administration and applications also are secured.

Configuration

Security Configuration Wizard | Security Configuration Report

Administrative security

- [✓] Enable administrative security
- Administrative User Roles
- Administrative Group Roles

Application security

- [✓] Enable application security

Java 2 security

- [ ] Use Java 2 security to restrict application access to local resources
  - [ ] Warn if applications are granted custom permissions
  - [ ] Restrict access to resource authentication data

User account repository

Current realm definition: Standalone LDAP registry

Available realm definitions: Standalone LDAP registry | Configure | Set as current

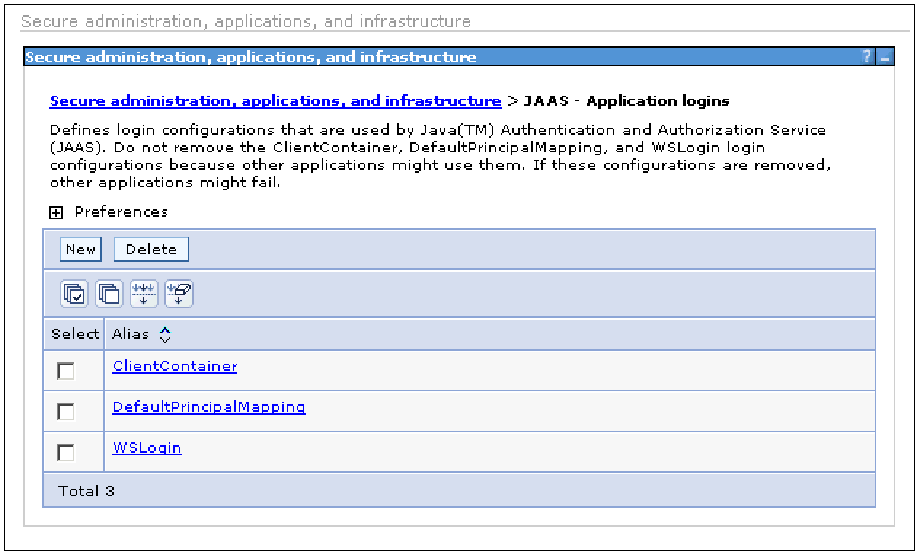
Authentication

- [ ] Use domain-qualified user names
- ⊞ Web security
- ⊞ RMI/IIOP security
- ⊟ Java Authentication and Authorization Service
  - Application logins
  - System logins
  - J2C authentication data
- Authentication mechanisms and expiration

- External authorization providers
- Custom properties

Figure 4-21 Secure administration, applications, and infrastructure window on the ISC

2. Next, go to the right pane and select **Authentication ∅ Java Authentication Authorization Service ∅ Application logins**, as shown in Figure 4-22.


Secure administration, applications, and infrastructure

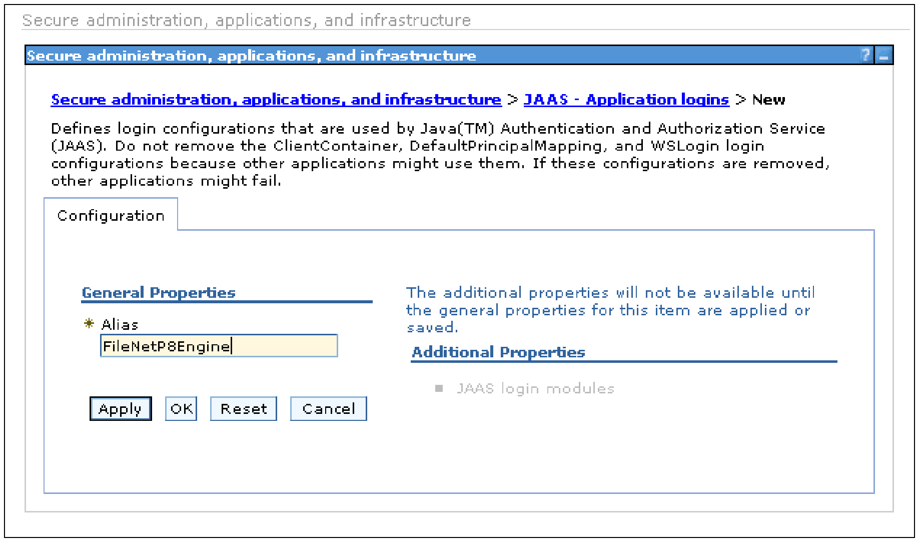
Secure administration, applications, and infrastructure > JAAS - Application logins

Defines login configurations that are used by Java(TM) Authentication and Authorization Service (JAAS). Do not remove the ClientContainer, DefaultPrincipalMapping, and WSLogin login configurations because other applications might use them. If these configurations are removed, other applications might fail.

Select	Alias
☐	ClientContainer
☐	DefaultPrincipalMapping
☐	WSLogin
Total 3	

Figure 4-22 List of default JAAS application logins

3. We need to add one application login context for FileNet P8, so click **New**. The window shown in Figure 4-23 appears.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > JAAS - Application logins > New

Defines login configurations that are used by Java(TM) Authentication and Authorization Service (JAAS). Do not remove the ClientContainer, DefaultPrincipalMapping, and WSLogin login configurations because other applications might use them. If these configurations are removed, other applications might fail.

General Properties

* Alias: FileNetP8Engine

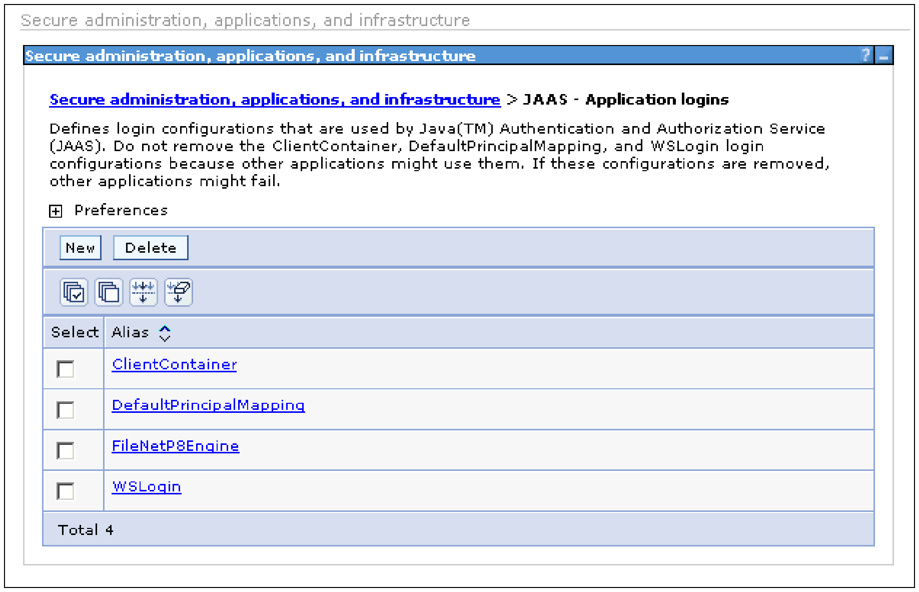
The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties

Figure 4-23 Window for new JAAS application login

4. Type in the alias name FileNetP8Engine and click **OK**. Click **Save**.

   After you have saved the configuration, you should see the list shown in Figure 4-24. There now are four login configurations.



Secure administration, applications, and infrastructure

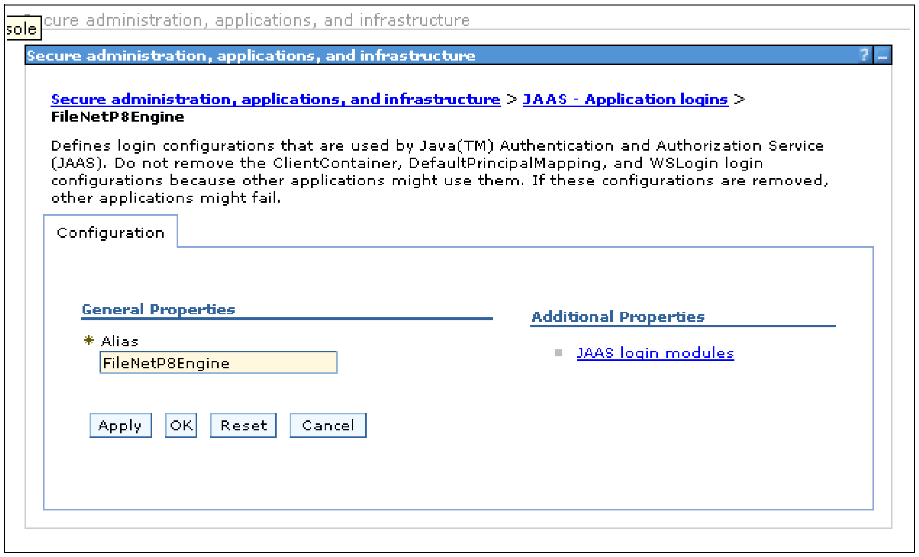
Secure administration, applications, and infrastructure > JAAS - Application logins

Defines login configurations that are used by Java(TM) Authentication and Authorization Service (JAAS). Do not remove the ClientContainer, DefaultPrincipalMapping, and WSLogin login configurations because other applications might use them. If these configurations are removed, other applications might fail.

Select	Alias
☐	ClientContainer
☐	DefaultPrincipalMapping
☐	FileNetP8Engine
☐	WSLogin
Total 4	

Figure 4-24 JAAS application logins after adding the FileNetP8Engine entry

5. Click the link for the **FileNetP8Engine**. You should see the window shown in Figure 4-25.



sole Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > JAAS - Application logins > FileNetP8Engine

Defines login configurations that are used by Java(TM) Authentication and Authorization Service (JAAS). Do not remove the ClientContainer, DefaultPrincipalMapping, and WSLogin login configurations because other applications might use them. If these configurations are removed, other applications might fail.

Configuration

General Properties

* Alias

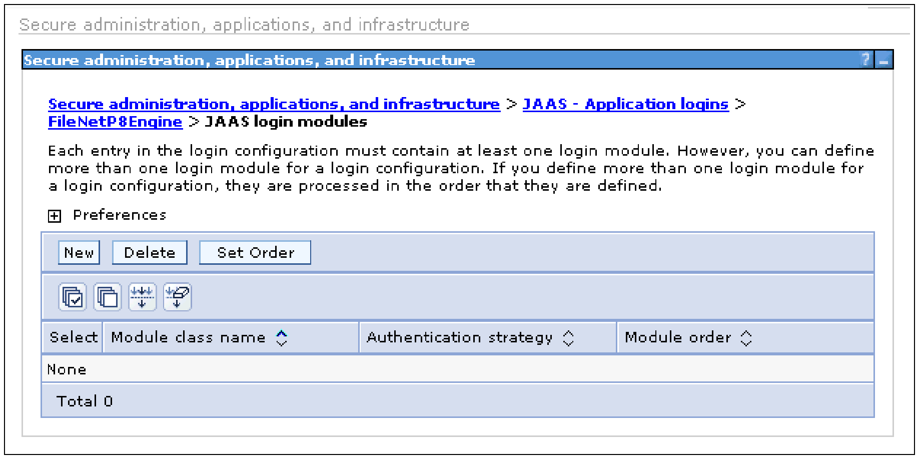
FileNetP8Engine

Additional Properties

JAAS login modules

Figure 4-25 JAAS application login general properties for FileNetP8Engine

6. Click the **JAAS login modules** link. You should see the window shown in Figure 4-26.



Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

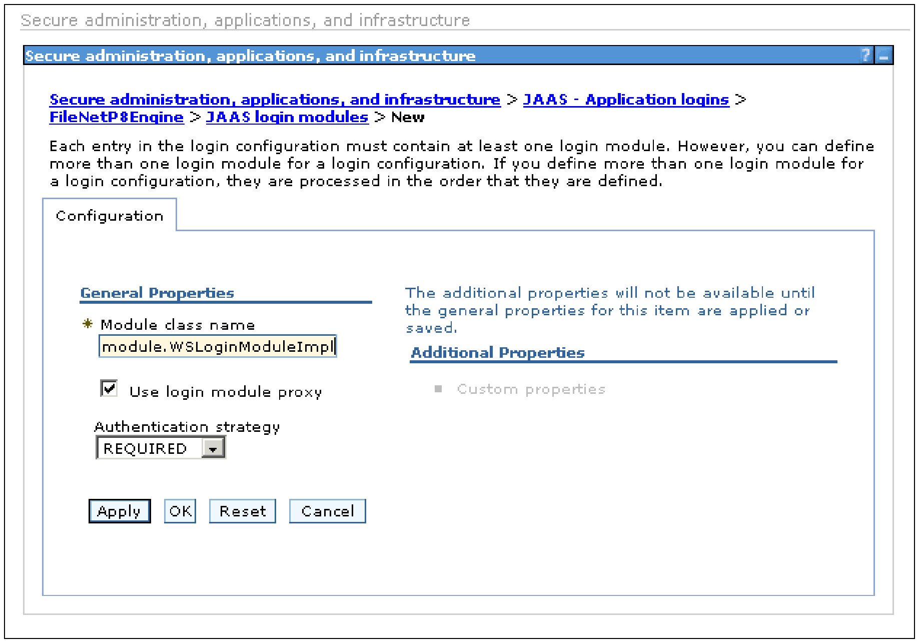
Secure administration, applications, and infrastructure > JAAS - Application logins > FileNetP8Engine > JAAS login modules

Each entry in the login configuration must contain at least one login module. However, you can define more than one login module for a login configuration. If you define more than one login module for a login configuration, they are processed in the order that they are defined.

Select	Module class name	Authentication strategy	Module order
None			
Total 0			

Figure 4-26 JAAS login module for FileNetP8Engine

7. Click the **New** button. The window shown in Figure 4-27 on page 101should appear.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > JAAS - Application logins > FileNetP8Engine > JAAS login modules > New

Each entry in the login configuration must contain at least one login module. However, you can define more than one login module for a login configuration. If you define more than one login module for a login configuration, they are processed in the order that they are defined.

Configuration

General Properties

* Module class name: module.WSLoginModuleImpl

[✓] Use login module proxy

Authentication strategy: REQUIRED

The additional properties will not be available until the general properties for this item are applied or saved.

Additional Properties

- Custom properties

Figure 4-27 Specify module class name for JAAS login module

8. In this window, you need to specify the com.ibm.ws.security.common.auth.module.WSLoginModuleImpl for Module class name. Confirm that **Use login module proxy** is checked. Click **OK** and then **Save**.

Note: The Module class name is case sensitive, so be careful when typing in the name.

Modify deployment descriptors for the P8 Application Engine

If you already deployed your P8 AE on WebSphere Application Server, you have to uninstall it. After you have successfully uninstalled it, you may continue the deployment steps.

First, you must modify some of the parts of the application deployment descriptors and re-deploy it, as shown in the following steps:

1. Open and modify the `web.xml` file. The file is located in `C:\Program Files\FileNet\AE\Workplace\WEB-INF` in Windows. The sections in **bold** need to be updated, as shown in Example 4-4.

Example 4-4 The parts that need to be updated in Web.xml file

```
<filter>
    <filter-name>ContainerBasedFilter</filter-name>
    <filter-class>com.filenet.ae.toolkit.server.servlet.filter.Conta
inerBasedFilter</filter-class>
    <init-param>
        <param-name>challengeProxyEnabled</param-name>
        <param-value>false</param-value>
    </init-param>
    <init-param>
        <param-name>challengeProxyURI</param-name>
        <param-value>containerSecured/Return.jsp</param-value>
    </init-param>
    <init-param>
        <param-name>perimeterChallengeMode</param-name>
        <param-value>true</param-value>
    </init-param>
    <init-param>
        <param-name>ssoProxyContextPath</param-name>
        <param-value>/Workplace</param-value>
    </init-param>
    <init-param>
        <param-name>ssoProxyHost</param-name>
        <param-value>rbtam</param-value>
    </init-param>
    <init-param>
        <param-name>ssoProxyPort</param-name>
        <param-value>80</param-value>
    </init-param>
    <init-param>
        <param-name>ssoProxySSLPort</param-name>
        <param-value>443</param-value>
    </init-param>
</filter>
```

2. Locate the `security-constraint` section and delete everything from this point down to the end of the file. Insert the lines shown in Example 4-5 on page 103 at the end of file.

Example 4-5 Lines inserted into Web.xml

```
<security-constraint>
  <web-resource-collection>
    <web-resource-name>action</web-resource-name>
    <description>Define the container non-secured
resource</description>
    <url-pattern>/P8BPMWSBroker/*</url-pattern>
  </web-resource-collection>
</security-constraint>

<security-constraint>
  <web-resource-collection>
    <web-resource-name>action</web-resource-name>
    <description>Define the container secured resource</description>
    <url-pattern>/containerSecured/*</url-pattern>
    <url-pattern>/</url-pattern>
    <url-pattern>/author/*</url-pattern>
    <url-pattern>/Browse.jsp</url-pattern>
    <url-pattern>/eprocess/*</url-pattern>
    <url-pattern>/Favorites.jsp</url-pattern>
    <url-pattern>/GetPortalSitePreferences.jsp</url-pattern>
    <url-pattern>/GetTokenSignIn.jsp</url-pattern>
    <url-pattern>/GetUserInformation.jsp</url-pattern>
    <url-pattern>/GetUserToken.jsp</url-pattern>
    <url-pattern>/HomePage.jsp</url-pattern>
    <url-pattern>/IntegrationWebBasedHelp.jsp</url-pattern>
    <url-pattern>/is/*</url-pattern>
    <url-pattern>/operations/*</url-pattern>
    <url-pattern>/portlets/Author/edit.jsp</url-pattern>
    <url-pattern>/portlets/Author/portlet.jsp</url-pattern>
    <url-pattern>/portlets/Browse/edit.jsp</url-pattern>
    <url-pattern>/portlets/Browse/portlet.jsp</url-pattern>
    <url-pattern>/portlets/ExternalUrl/edit.jsp</url-pattern>
    <url-pattern>/portlets/ExternalUrl/portlet.jsp</url-pattern>
    <url-pattern>/portlets/GroupPageDesign.jsp</url-pattern>
    <url-pattern>/portlets/GroupPageSettings.jsp</url-pattern>
    <url-pattern>/portlets/Inbox/edit.jsp</url-pattern>
    <url-pattern>/portlets/Inbox/portlet.jsp</url-pattern>
    <url-pattern>/portlets/MultiPagesDesign.jsp</url-pattern>
    <url-pattern>/portlets/OrganizePages.jsp</url-pattern>
    <url-pattern>/portlets/PortalPageDesign.jsp</url-pattern>
    <url-pattern>/portlets/PortalPageInfo.jsp</url-pattern>
    <url-pattern>/portlets/PortletAlias.jsp</url-pattern>
    <url-pattern>/portlets/PortletSettings.jsp</url-pattern>
```

```
<url-pattern>/portlets/PreviewAndSetup.jsp</url-pattern>
<url-pattern>/portlets/PublicQueue/edit.jsp</url-pattern>
<url-pattern>/portlets/PublicQueue/portlet.jsp</url-pattern>
<url-pattern>/portlets/QuickSearch/edit.jsp</url-pattern>
<url-pattern>/portlets/QuickSearch/portlet.jsp</url-pattern>
<url-pattern>/portlets/Workflows/edit.jsp</url-pattern>
<url-pattern>/portlets/Workflows/portlet.jsp</url-pattern>
<url-pattern>/properties/*</url-pattern>
<url-pattern>/redirect/*</url-pattern>
<url-pattern>/regions/*</url-pattern>
<url-pattern>/Search.jsp</url-pattern>
<url-pattern>/select/*</url-pattern>
<url-pattern>/SelectReturn.jsp</url-pattern>
<url-pattern>/Tasks.jsp</url-pattern>
<url-pattern>/UI-INF/*</url-pattern>
<url-pattern>/utils/*</url-pattern>
<url-pattern>/WcmAdmin.jsp</url-pattern>
<url-pattern>/WcmAuthor.jsp</url-pattern>
<url-pattern>/WcmBootstrap.jsp</url-pattern>
<url-pattern>/WcmCloseWindow.jsp</url-pattern>
<url-pattern>/WcmDefault.jsp</url-pattern>
<url-pattern>/WcmError.jsp</url-pattern>
<url-pattern>/WcmJavaViewer.jsp</url-pattern>
<url-pattern>/WcmObjectBookmark.jsp</url-pattern>
<url-pattern>/WcmPortletHelp.jsp</url-pattern>
<url-pattern>/WcmPortletSearch.jsp</url-pattern>
<url-pattern>/WcmQueueBookmark.jsp</url-pattern>
<url-pattern>/WcmSignIn.jsp</url-pattern>
<url-pattern>/WcmSitePreferences.jsp</url-pattern>
<url-pattern>/WcmUserPreferences.jsp</url-pattern>
<url-pattern>/WcmWorkflowsBookmark.jsp</url-pattern>
<url-pattern>/wizards/*</url-pattern>
<url-pattern>/Author/*</url-pattern>
<url-pattern>/axis/*.jws</url-pattern>
<url-pattern>/Browse/*</url-pattern>
<url-pattern>/ceTunnel</url-pattern>
<url-pattern>/CheckoutList/*</url-pattern>
<url-pattern>/downloadMultiTransferElement/*</url-pattern>
<url-pattern>/ExternalUrl/*</url-pattern>
<url-pattern>/findRecordTarget</url-pattern>
<url-pattern>/formCallback/*</url-pattern>
<url-pattern>/getAnnotSecurity/*</url-pattern>
<url-pattern>/getCEAnnotations/*</url-pattern>
<url-pattern>/getContent/*</url-pattern>
<url-pattern>/getForm/*</url-pattern>
```

```
    <url-pattern>/getISAnnotations/*</url-pattern>
    <url-pattern>/getISAnnotSecurity/*</url-pattern>
    <url-pattern>/getISContent/*</url-pattern>
    <url-pattern>/getMultiContent/*</url-pattern>
    <url-pattern>/getPreview</url-pattern>
    <url-pattern>/getProcessor/*</url-pattern>
    <url-pattern>/getRealms/*</url-pattern>
    <url-pattern>/getUsersGroups/*</url-pattern>
    <url-pattern>/Inbox/*</url-pattern>
    <url-pattern>/integrationCommandProxy</url-pattern>
    <url-pattern>/integrationResponse</url-pattern>
    <url-pattern>/integrationResponseProxy</url-pattern>
    <url-pattern>/integrationWebBasedCommand</url-pattern>
    <url-pattern>/keepAlive</url-pattern>
    <url-pattern>/launch/*</url-pattern>
    <url-pattern>/PublicQueue/*</url-pattern>
    <url-pattern>/putContent/*</url-pattern>
    <url-pattern>/QuickSearch/*</url-pattern>
    <url-pattern>/signingServlet/*</url-pattern>
    <url-pattern>/transport/*</url-pattern>
    <url-pattern>/upload/*</url-pattern>
    <url-pattern>/vwsimsoapservlet</url-pattern>
    <url-pattern>/vwsoaprouter</url-pattern>
    <url-pattern>/Workflows/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
  <role-name>All Authenticated</role-name>
  </auth-constraint>
  <user-data-constraint>
    <description>User data constraints</description>
    <transport-guarantee>NONE</transport-guarantee>
  </user-data-constraint>
</security-constraint>

<security-role>
  <description>All Authenticated</description>
  <role-name>All Authenticated</role-name>
  </security-role>

</web-app>
```

3. Next, you need to modify the `WcmAPiConfig.properties` file that is located in the `C:\Program Files\FileNet\AE\Workplace\WEB-INF` directory. Make sure the jaasConfigurationName stanza is defined as follows:

   ```
   jaasConfigurationName=!
   ```

4. Open the `application.xml` file located in the `C:\Program Files\FileNet\AE\deploy\META-INF` directory. Modify the file as shown in Example 4-6 (the sections in **bold** need to be updated or included).

   Example 4-6 Changes and additions to the application.xml file

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <!DOCTYPE application PUBLIC "-//Sun Microsystems, Inc.//DTD J2EE
   Application 1.3//EN"
   "http://java.sun.com/dtd/application_1_3.dtd">
   <application id="workplace">
     <!-- Note: The display-name has to be kept short for WebSphere;
   otherwise the path to access class files in WSI will easily exceed
   the max path size (260). -->
     <display-name>Workplace</display-name>
     <description>FileNet Application Engine</description>
     <module id="wp">
       <web>
         <web-uri>app_engine.war</web-uri>
         <context-root>Workplace</context-root>
       </web>
     </module>
     <security-role id="SecurityRole_1164745308375">
       <description>All Authenticated</description>
       <role-name>All Authenticated</role-name>
     </security-role>
   </application>
   ```

5. Next, create a new file with the name `ibm-application-bnd.xml` in the `C:\Program Files\FileNet\AE\deploy\META-INF` directory. The content of the file is shown in Example 4-7.

   Example 4-7 The new ibm-application-bnd.xml file

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <com.ibm.ejs.models.base.bindings.applicationbnd:ApplicationBinding
   xmi:version="2.0"
   xmlns:xmi="http://www.omg.org/XMI"
   xmlns:com.ibm.ejs.models.base.bindings.applicationbnd="applicationbn
   d.xmi"
   xmi:id="Application_ID_Bnd">
   ```

```
  <authorizationTable xmi:id="AuthorizationTable_1">
    <authorizations xmi:id="RoleAssignment_1155940459609">
      <specialSubjects
xmi:type="com.ibm.ejs.models.base.bindings.applicationbnd:AllAuthent
icatedUsers" xmi:id="AllAuthenticatedUsers_1164745308375"
name="AllAuthenticatedUsers"/>
      <role
href="META-INF/application.xml#SecurityRole_1164745308375"/>
    </authorizations>
  </authorizationTable>
  <application href="META-INF/application.xml#workplace"/>
  <runAsMap xmi:id="RunAsMap_1"/>
</com.ibm.ejs.models.base.bindings.applicationbnd:ApplicationBinding
>
```

6. Create a new file with the name `ibm-application-ext.xml` in the `C:\Program Files\FileNet\AE\deploy\META-INF` directory. The content of the file is shown in Example 4-8.

Example 4-8 The new ibm-application-ext.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<applicationext:ApplicationExtension xmi:version="2.0"
xmlns:xmi="http://www.omg.org/XMI"
xmlns:applicationext="applicationext.xmi"
xmlns:application="application.xmi"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmi:id="Application_ID_Ext"
sharedSessionContext="false">
<moduleExtensions xmi:type="applicationext:WebModuleExtension"
xmi:id="WebModule_1_Ext" altRoot="ALT-INF/app_engine.war">
  <module xmi:type="application:WebModule"
  href="META-INF/application.xml#wp"/>
</moduleExtensions>
<application href="META-INF/application.xml#workplace"/>
</applicationext:ApplicationExtension>
```

Create WAR and EAR file packages

Now you need to re-package your deployment descriptor with the changes performed in the previous steps.

Open a command line, change the current directory to `C:\Program Files\FileNet\AE\deploy`, and execute the following two batch files:

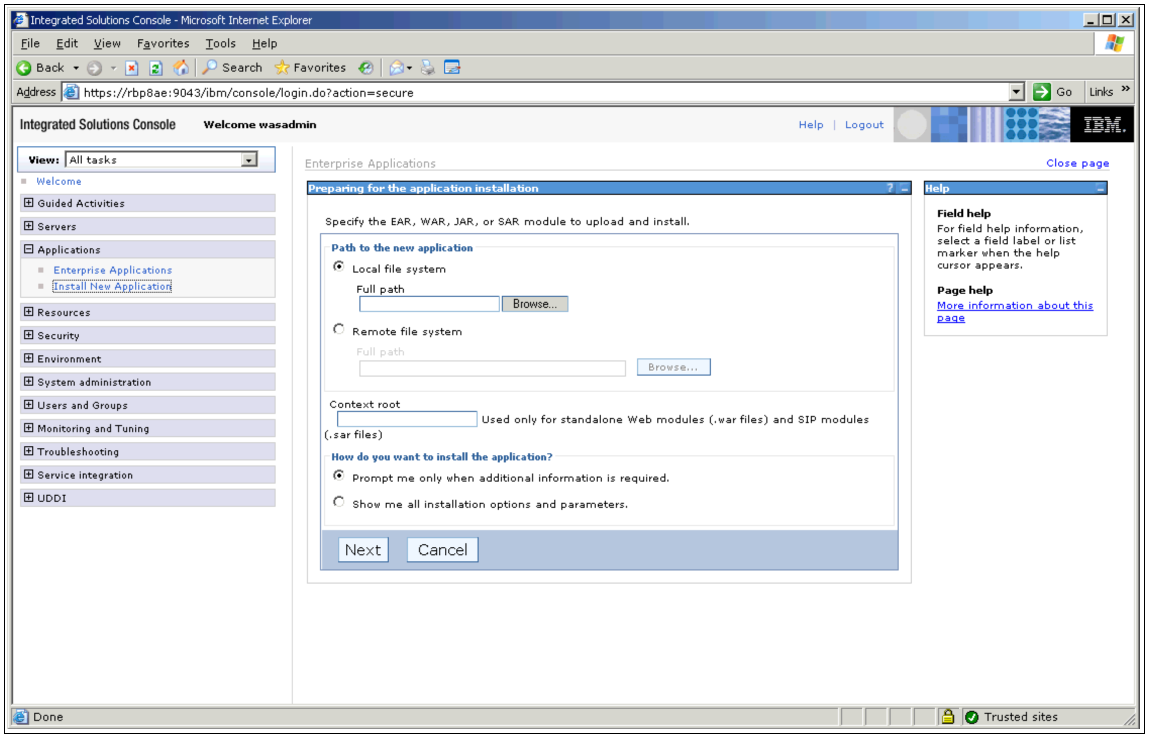
```
create_app_engine_war.bat
create_app_engine_ear.bat
```

Deploy the application using the WebSphere ISC

Now we are ready to deploy the FileNet P8 Application Engine application.

Do these steps:

1. Open the WebSphere Integrated Solutions Console in your Web browser. Click **Application** on the left side of the window, then click **Install New Application**. You should see the window shown in Figure 4-28.



Specify the EAR, WAR, JAR, or SAR module to upload and install.

Path to the new application

(●) Local file system

Full path [ ] Browse...

( ) Remote file system

Full path [ ] Browse...

Context root [ ] Used only for standalone Web modules (.war files) and SIP modules (.sar files)

How do you want to install the application?

(●) Prompt me only when additional information is required.

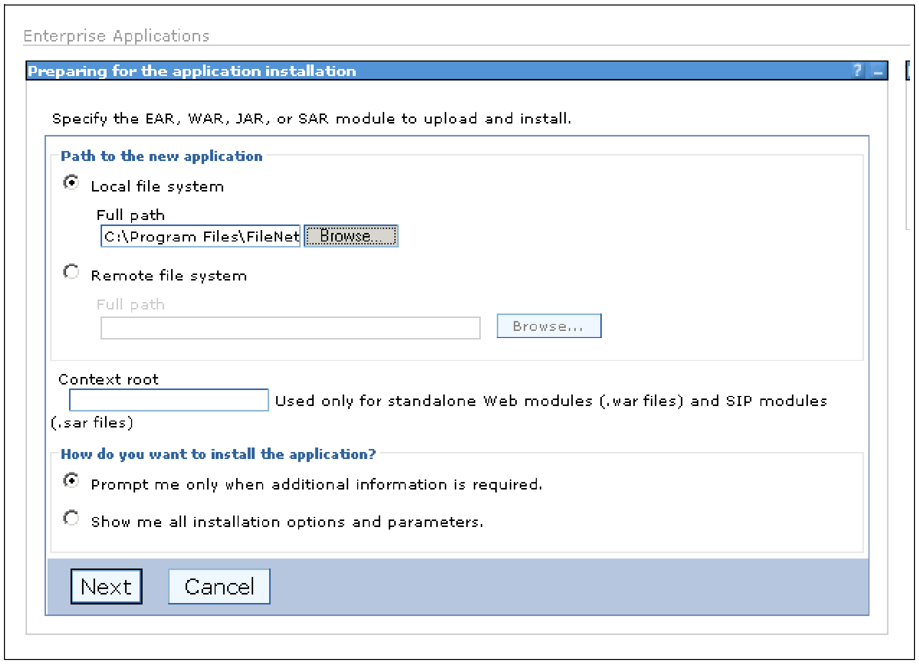
( ) Show me all installation options and parameters.

Field help
For field help information, select a field label or list marker when the help cursor appears.

Page help
More information about this page

Figure 4-28 Install new WebSphere application

2. Select **Local file system** as the install source and browse to `C:\Program Files\FileNet\AE\deploy\app_engine.ear`. Click **Next** to proceed to the window shown in Figure 4-29.



Enterprise Applications

Preparing for the application installation

Specify the EAR, WAR, JAR, or SAR module to upload and install.

Path to the new application

[✓ Local file system]

Full path: C:\Program Files\FileNet

Remote file system

Full path:

Context root: Used only for standalone Web modules (.war files) and SIP modules (.sar files)

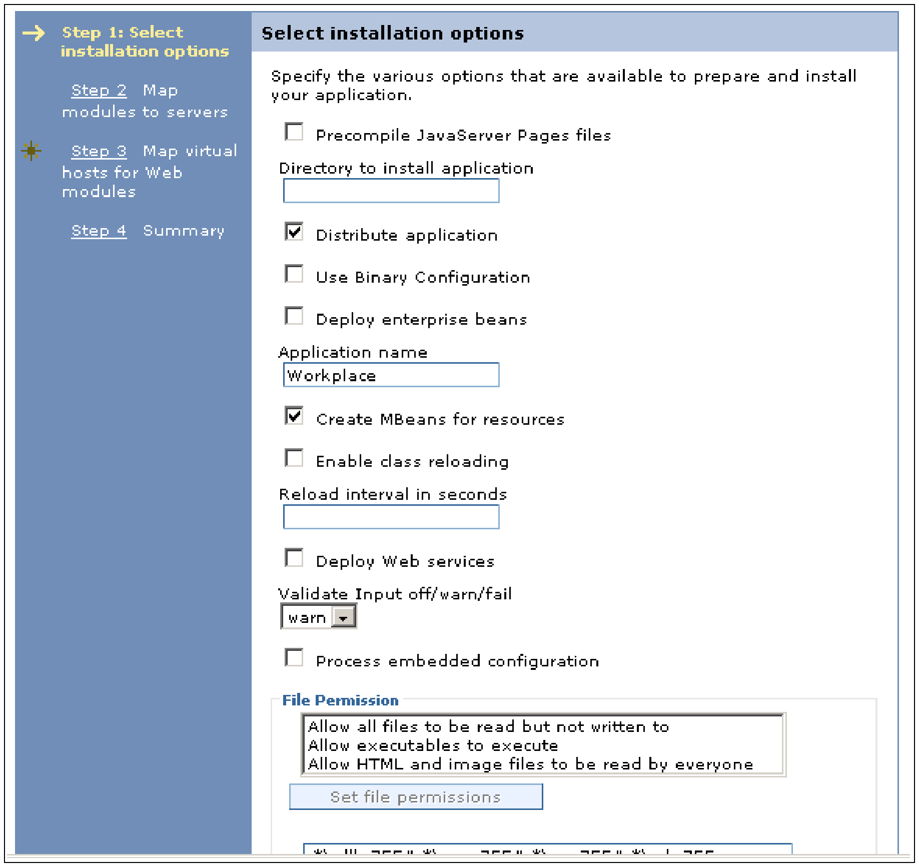
How do you want to install the application?

[✓ Prompt me only when additional information is required.]

Show me all installation options and parameters.

Figure 4-29 Specify the location of EAR file

3. No changes are needed in this window. Click **Next** to select more installation options, as shown in Figure 4-30.



Step 1: Select installation options

Step 2 Map modules to servers

Step 3 Map virtual hosts for Web modules

Step 4 Summary

Select installation options

Specify the various options that are available to prepare and install your application.

- [ ] Precompile JavaServer Pages files

Directory to install application:

- [x] Distribute application
- [ ] Use Binary Configuration
- [ ] Deploy enterprise beans

Application name: Workplace

- [x] Create MBeans for resources
- [ ] Enable class reloading

Reload interval in seconds:

- [ ] Deploy Web services

Validate Input off/warn/fail: warn

- [ ] Process embedded configuration

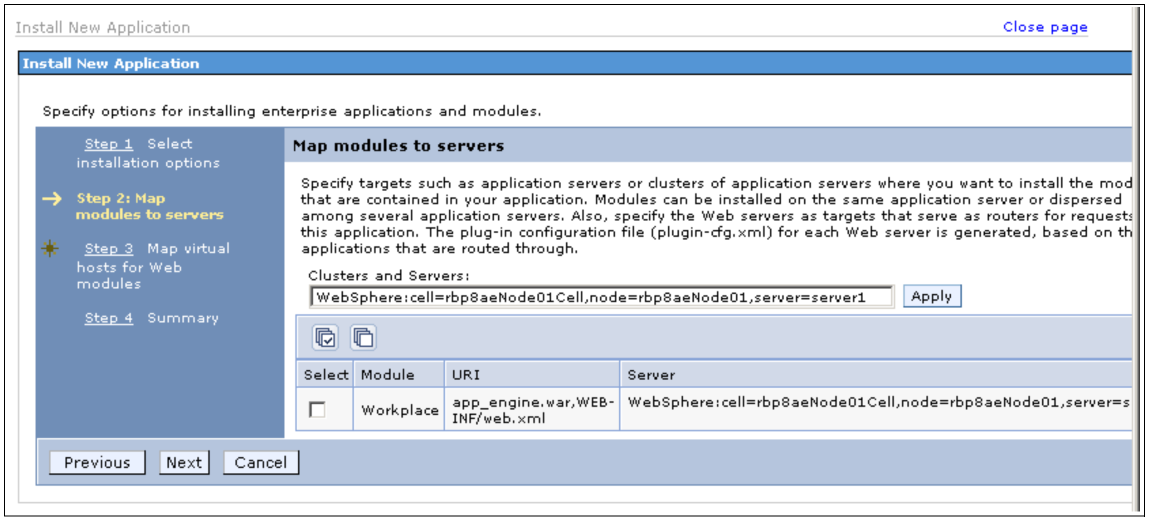
File Permission

Allow all files to be read but not written to
Allow executables to execute
Allow HTML and image files to be read by everyone

Set file permissions

Figure 4-30 Select installation options

4. No changes are needed in this window. Click **Next** to go to the window shown in Figure 4-31 on page 111.


Install New Application

Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2: Map modules to servers

Step 3 Map virtual hosts for Web modules

Step 4 Summary

Map modules to servers

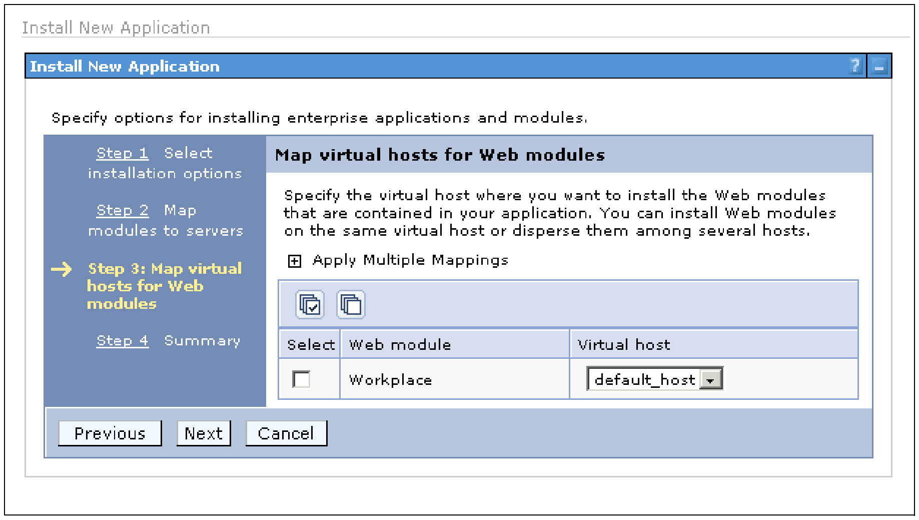
Specify targets such as application servers or clusters of application servers where you want to install the mod that are contained in your application. Modules can be installed on the same application server or dispersed among several application servers. Also, specify the Web servers as targets that serve as routers for requests this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated, based on th applications that are routed through.

Clusters and Servers:
WebSphere:cell=rbp8aeNode01Cell,node=rbp8aeNode01,server=server1

Select	Module	URI	Server
☐	Workplace	app_engine.war,WEB-INF/web.xml	WebSphere:cell=rbp8aeNode01Cell,node=rbp8aeNode01,server=s

Figure 4-31 Map modules to servers

5. No changes are needed in this window. Click **Next** to go to the window shown in Figure 4-32.


Install New Application

Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2 Map modules to servers

Step 3: Map virtual hosts for Web modules

Step 4 Summary

Map virtual hosts for Web modules

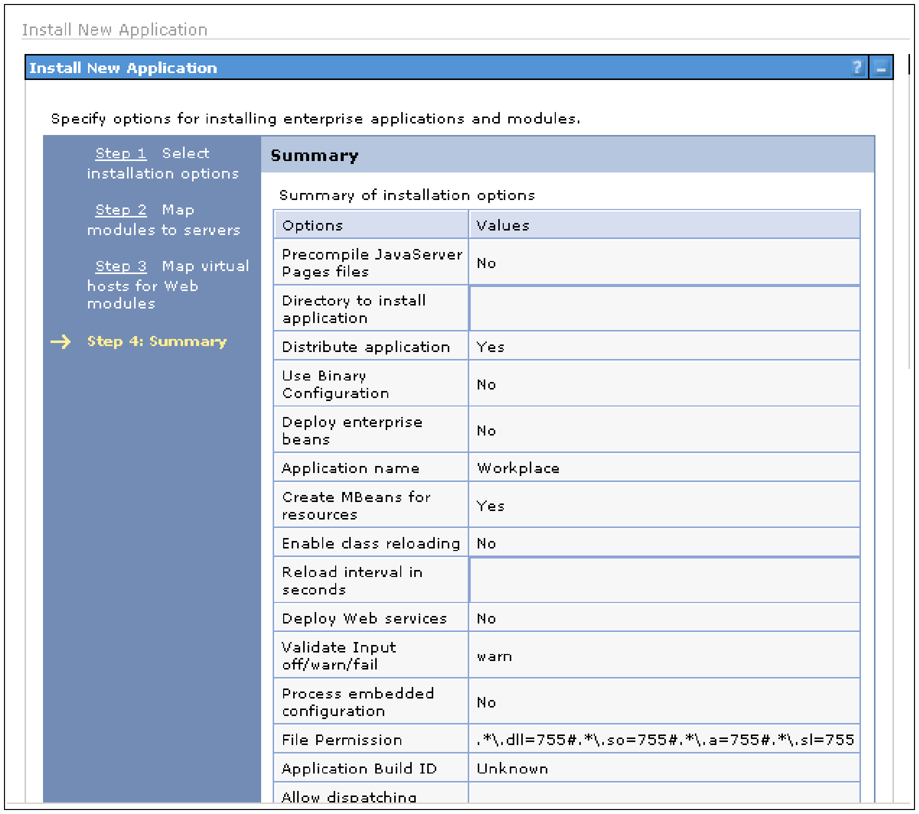
Specify the virtual host where you want to install the Web modules that are contained in your application. You can install Web modules on the same virtual host or disperse them among several hosts.

⊞ Apply Multiple Mappings

Select	Web module	Virtual host
☐	Workplace	default_host

Figure 4-32 Virtual hosts mapping for Web modules

6. No changes are needed in this window. Click **Next** to go to the installation summary window shown in Figure 4-33.


Install New Application

Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2 Map modules to servers

Step 3 Map virtual hosts for Web modules

Step 4: Summary

Summary

Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	
Distribute application	Yes
Use Binary Configuration	No
Deploy enterprise beans	No
Application name	Workplace
Create MBeans for resources	Yes
Enable class reloading	No
Reload interval in seconds	
Deploy Web services	No
Validate Input off/warn/fail	warn
Process embedded configuration	No
File Permission	.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
Application Build ID	Unknown
Allow dispatching	

Figure 4-33 Summary of installation

7. Click **Finish** and wait for the completion of the installation. If the installation is completed successfully, you should see the window shown in Figure 4-34 on page 113.

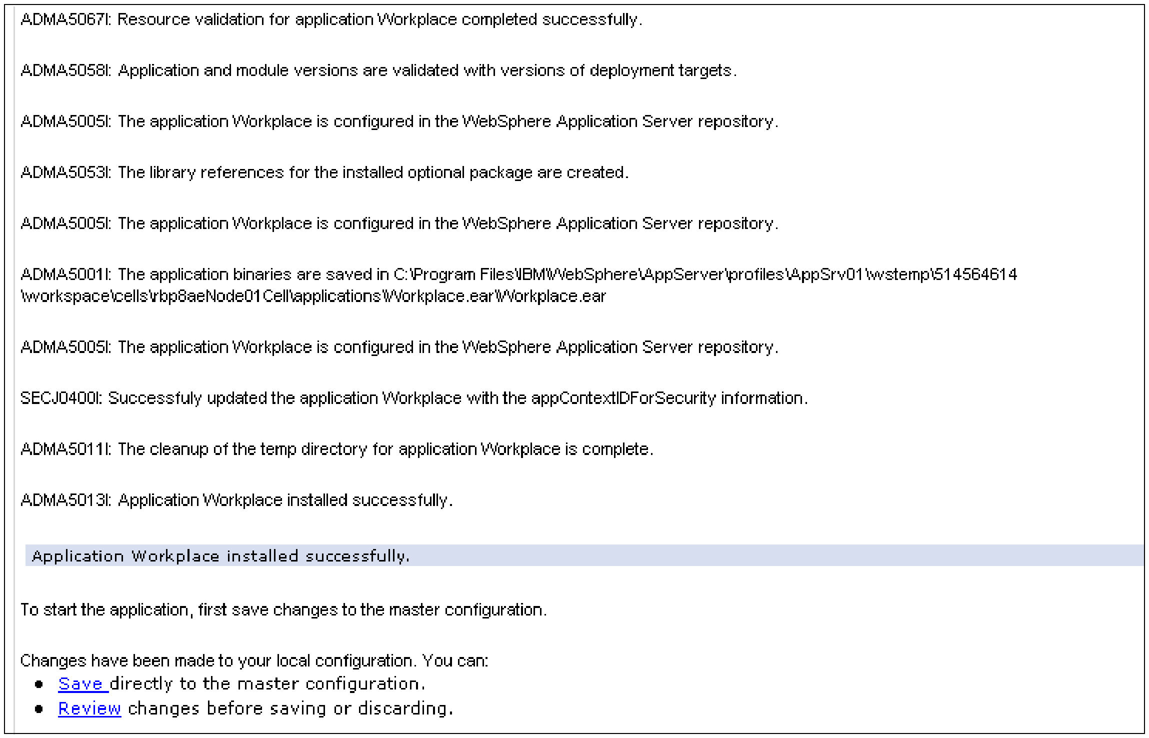

```
ADMA5067I: Resource validation for application Workplace completed successfully.

ADMA5058I: Application and module versions are validated with versions of deployment targets.

ADMA5005I: The application Workplace is configured in the WebSphere Application Server repository.

ADMA5053I: The library references for the installed optional package are created.

ADMA5005I: The application Workplace is configured in the WebSphere Application Server repository.

ADMA5001I: The application binaries are saved in C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\wstemp\514564614
\workspace\cells\rbp8aeNode01Cell\applications\Workplace.ear\Workplace.ear

ADMA5005I: The application Workplace is configured in the WebSphere Application Server repository.

SECJ0400I: Successfuly updated the application Workplace with the appContextIDForSecurity information.

ADMA5011I: The cleanup of the temp directory for application Workplace is complete.

ADMA5013I: Application Workplace installed successfully.

Application Workplace installed successfully.

To start the application, first save changes to the master configuration.

Changes have been made to your local configuration. You can:
 • Save directly to the master configuration.
 • Review changes before saving or discarding.
```

Figure 4-34 Example of successful deployment message

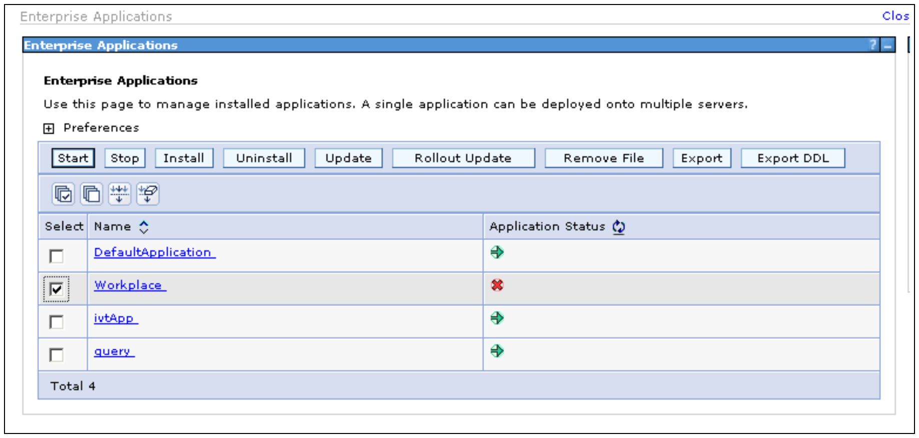
8. Click **Save** to save your configuration to the master configuration repository.

Security mapping for P8 Workplace application

Next, you should set the appropriate security role mapping for the Workplace application.

Do these steps:

1. Go to the **Application** menu on the left side of the window. Click **Enterprise Application**. You should get the windows shown in Figure 4-35.



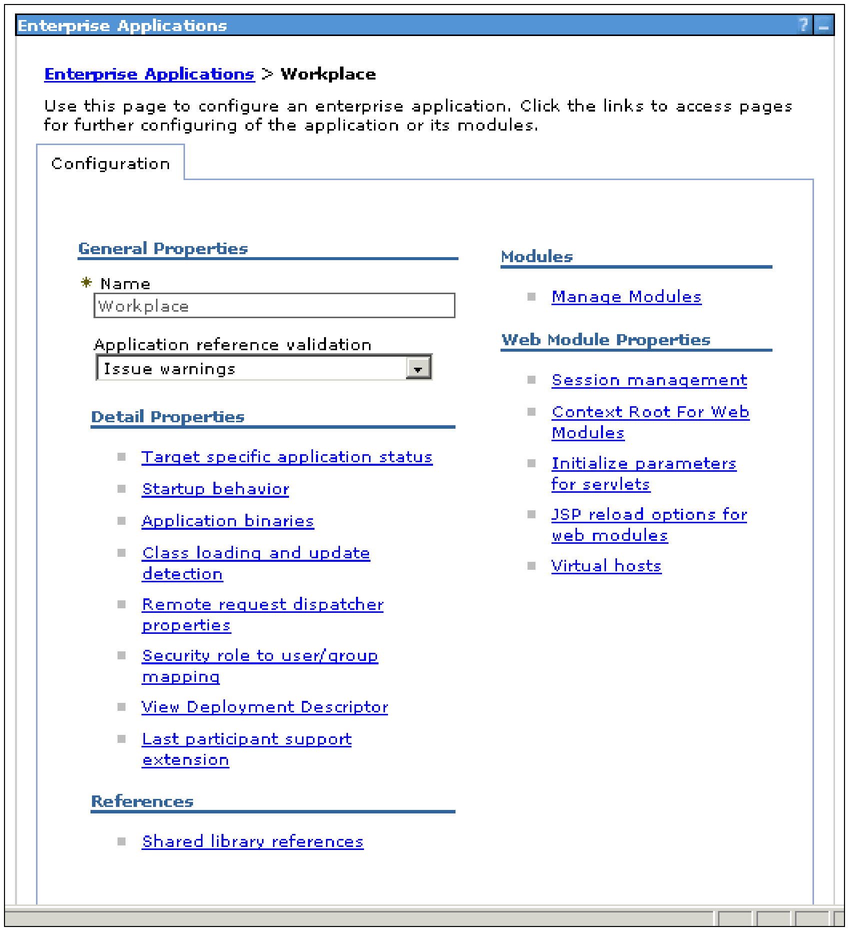
Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple servers.

Select	Name	Application Status
☐	DefaultApplication	➜
☑	Workplace	✖
☐	ivtApp	➜
☐	query	➜
Total 4		

Figure 4-35 The list of Enterprise Applications

2. Click **Workplace**. You should get the window shown in Figure 4-36.



Enterprise Applications

Enterprise Applications > Workplace

Use this page to configure an enterprise application. Click the links to access pages for further configuring of the application or its modules.

Configuration

General Properties

* Name
Workplace

Application reference validation
Issue warnings

Detail Properties

- Target specific application status
- Startup behavior
- Application binaries
- Class loading and update detection
- Remote request dispatcher properties
- Security role to user/group mapping
- View Deployment Descriptor
- Last participant support extension

References

- Shared library references

Modules

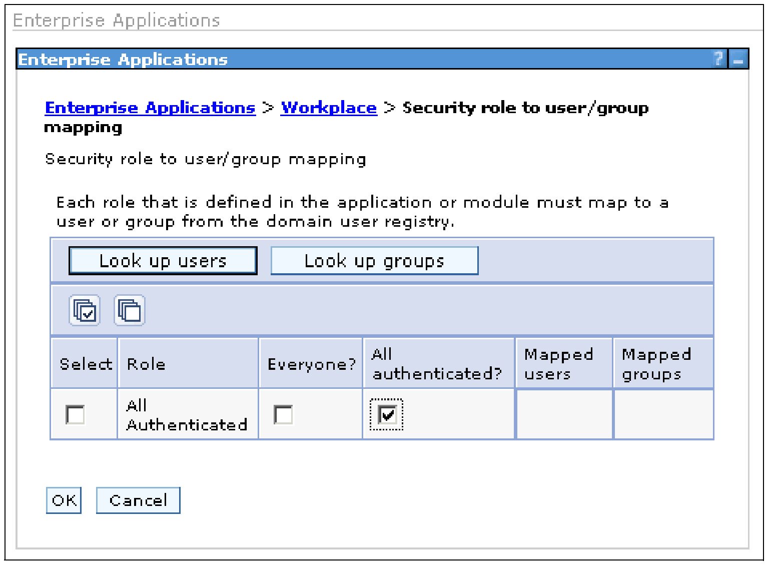
- Manage Modules

Web Module Properties

- Session management
- Context Root For Web Modules
- Initialize parameters for servlets
- JSP reload options for web modules
- Virtual hosts

Figure 4-36 Workspace application properties

3. Click **Security role to user/group mapping**. You should get the window shown in Figure 4-37.



Enterprise Applications

Enterprise Applications

Enterprise Applications > Workplace > Security role to user/group mapping

Security role to user/group mapping

Each role that is defined in the application or module must map to a user or group from the domain user registry.

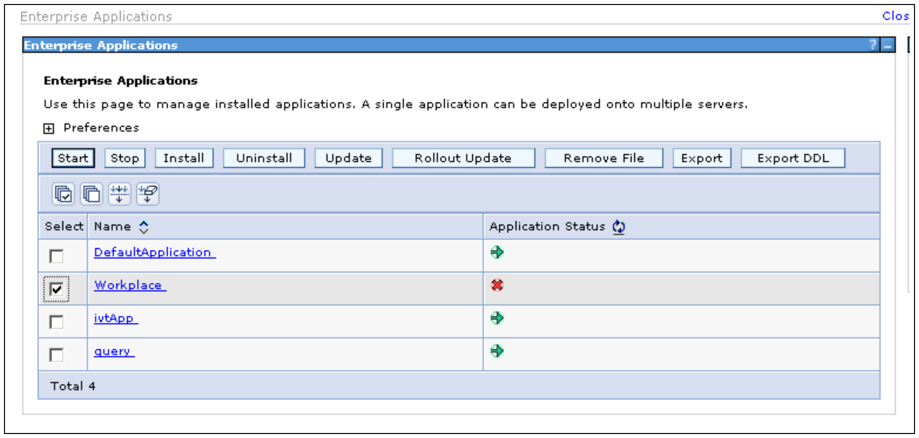
Select	Role	Everyone?	All authenticated?	Mapped users	Mapped groups
☐	All Authenticated	☐	☑		

Figure 4-37 Security role to user/group mapping for Workplace application

You should check **All authenticated?**, as shown in Figure 4-37. Then click **OK** and click **Save**.

Now you are ready to start your P8 AE server. In a real world deployment, this would be a good time to complete the AE post deployment steps.

4. Go to the **Application** menu on the left side of the window. Click **Enterprise Application**. You should see the window shown in Figure 4-38 on page 117.


Enterprise Applications

Enterprise Applications

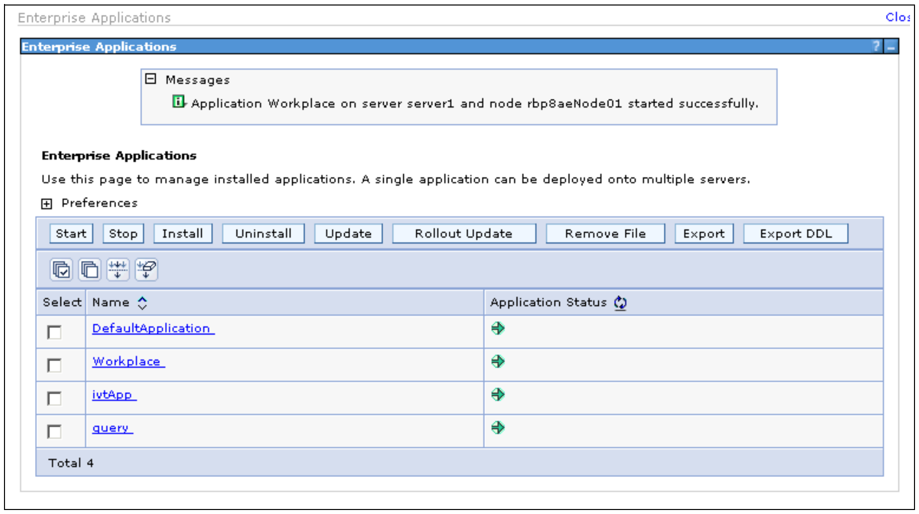
Use this page to manage installed applications. A single application can be deployed onto multiple servers.

⊞ Preferences

Select	Name	Application Status
☐	DefaultApplication	➜
☑	Workplace	✖
☐	ivtApp	➜
☐	query	➜
Total 4		

Figure 4-38 The list of Enterprise Applications

5. Check the box to the left of the **Workplace** application, as shown in Figure 4-39. Then click **Start**. After a while, you should see a successful startup message, as shown in Figure 4-39.


Enterprise Applications

Enterprise Applications

⊟ Messages

Application Workplace on server server1 and node rbp8aeNode01 started successfully.

Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple servers.

⊞ Preferences

Select	Name	Application Status
☐	DefaultApplication	➜
☐	Workplace	➜
☐	ivtApp	➜
☐	query	➜
Total 4		

Figure 4-39 Successfully start the workplace application

4.2.7 Additional configuration steps for SPNEGO authentication

In our final configuration section for Access Manager, we want to explore the SPNEGO functionality. Let us take a look at the following details:

- “What SPNEGO desktop single sign-on is” on page 118
- “Server side settings” on page 118
- “Client side settings” on page 120

What SPNEGO desktop single sign-on is

SPNEGO is used to provide single sign-on functionality from a Microsoft Windows based desktop, which is a member of a Microsoft Active Directory domain, to Web servers that support the SPNEGO protocol.

Tivoli Access Manager for e-business supports SPNEGO, so if your client PC is a member of an Active Directory domain, WebSEAL can use an SPNEGO token for its user authentication and never ask the user for their identity and password information.

Typically, FileNet P8 is deployed on an existing Active Directory infrastructure. This means that users always log on to their Active Directory domain from their client PC. In this environment, we can omit an additional user logon process for WebSEAL because we can utilize SPNEGO.

Note: There are some limitations when using SPNEGO with WebSEAL.

1. The user cannot log out from WebSEAL because the client is always re-authenticated automatically, even if the user terminates their session.
2. The user cannot change their password for Access Manager. The **`pkmspasswd`** command does not work with SPNEGO authentication enabled.
3. Access Manager Protected Object Policy (POP) or session-timer based re-authentication does not work.
4. Microsoft NTLM authentication is not supported. This means Active Directory must run on machines with Windows 2000 or newer, Windows NT is not supported. Also, a client on Windows NT is not supported.

Server side settings

On the server side, we have to configure two things:

- “Create Kerberos principal for WebSEAL instance” on page 119
- “Enable SPNEGO authentication on WebSEAL” on page 119

Create Kerberos principal for WebSEAL instance

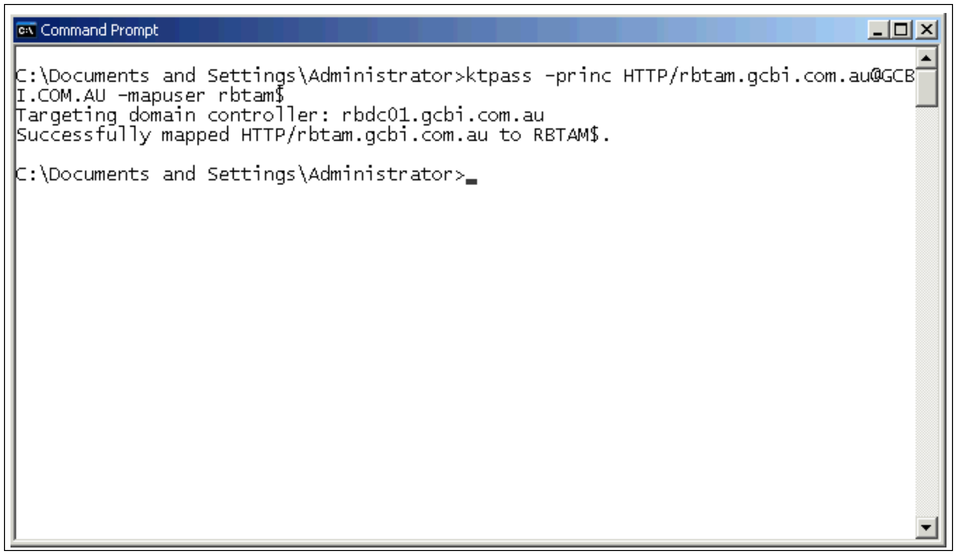
We have to execute the **ktpass** command on the domain controller machine in order to create a Kerberos principal for our WebSEAL server.

Two options for the **ktpass** command are mandatory: -princ and -mapuser. For the option -princ, you should specify the fixed keyword HTTP/ plus your WebSEAL server fully qualified DN host name, including an @ mark plus your active directory domain name. For the option -mapuser, you should specify the host name without a domain name using a trailing $ sign. Here is our example:

```
ktpass -princ HTTP/rbtam.gcbi.com.au@GCBI.COM.AU -mapuser rbtam$
```

Note: The **ktpass** command is a part of the Windows support tools. To use the **ktpass** command, you should install the Windows support tools from your Windows installation media.

Figure 4-40 shows the execution of our example command with its response in a command prompt window.



```
C:\Documents and Settings\Administrator>ktpass -princ HTTP/rbtam.gcbi.com.au@GCB
I.COM.AU -mapuser rbtam$
Targeting domain controller: rbdc01.gcbi.com.au
Successfully mapped HTTP/rbtam.gcbi.com.au to RBTAM$.

C:\Documents and Settings\Administrator>_
```

Figure 4-40 An example of the ktpass command and its response

Enable SPNEGO authentication on WebSEAL

In order to enable WebSEAL for SPNEGO authentication, follow these steps:

1. Open the `Webseal-default.conf` file located in `C:\Program Files\Tivoli\PDWeb\etc`.

2. Find the `spnego` stanza and change the value for the `spnego-auth` parameter to `both`.

   There are three choices for this parameter: http, https, and both. If you need to restrict access to SSL only, you should specify `https` for this parameter. Since our test case is not a production environment and it is more convenient for us to use a non-SSL protocol to get a debug trace over the wire, we use `both` for this parameter as follows:

   ```
   spnego-auth = both
   ```

3. Next, find the `authentication-mechanisms` stanza and provide the following value for the kerberos5 entry, which specifies the location of the DLL for the SPNEGO authentication:

   ```
   kerberos5 = C:\PROGRA~1\Tivoli\Policy~1\bin\stliauthn.dll
   ```

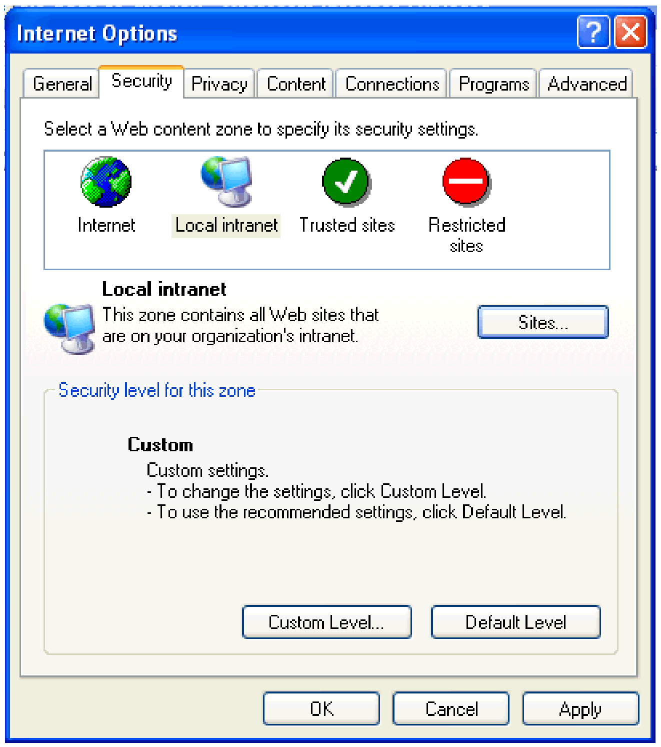
4. To activate your changes, you have to restart WebSEAL.

Client side settings

To correctly enable the SPNEGO authentication, you need to appropriately configure the Microsoft Internet Explorer browser.

Do these steps:

1. Add the host name of the WebSEAL server to the Local intranet zone or Trusted sites zone. Start Internet Explorer, then select **Tools** ∅ **Internet Options...** ∅ **Security**. You should see the window shown in Figure 4-41.



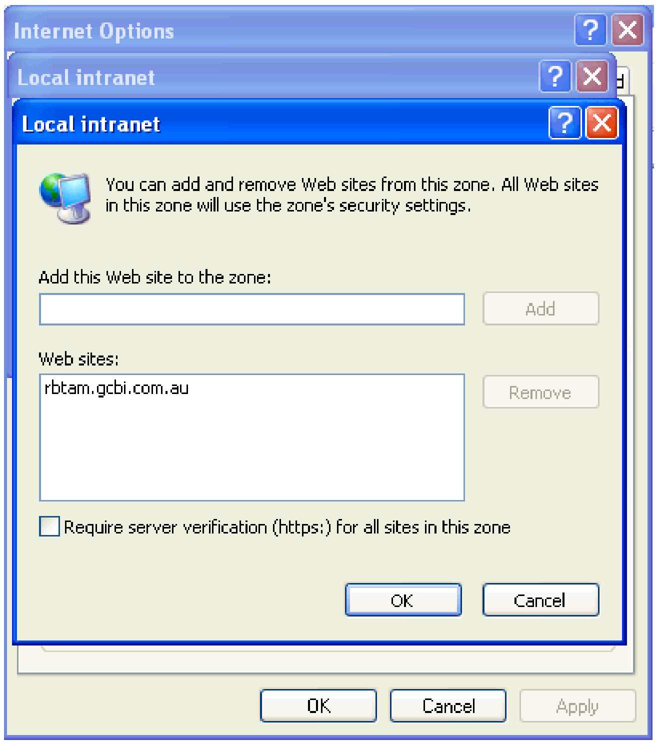
Select a Web content zone to specify its security settings.

Local intranet
This zone contains all Web sites that are on your organization's intranet.

Custom
Custom settings.
- To change the settings, click Custom Level.
- To use the recommended settings, click Default Level.

Figure 4-41 IE Security zone setting window

2. Select the **Local intranet** icon and click **Sites...**. You should see the window shown in Figure 4-42.



You can add and remove Web sites from this zone. All Web sites in this zone will use the zone's security settings.

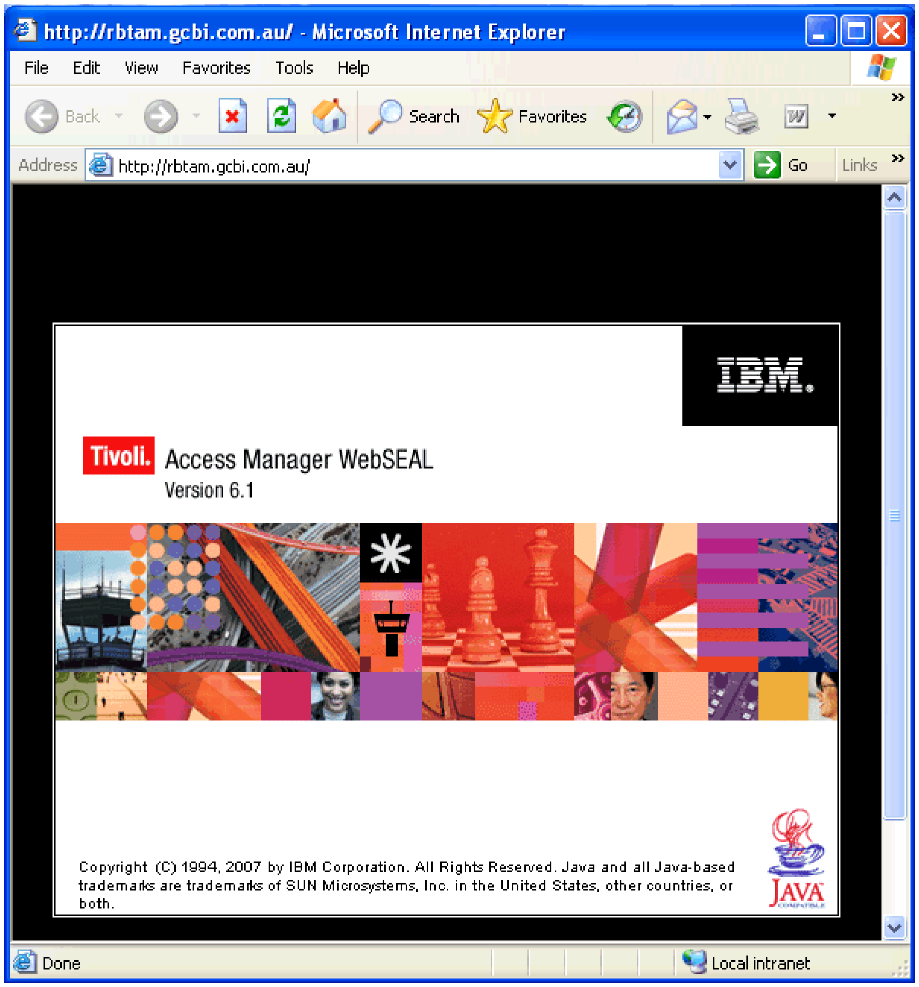
Add this Web site to the zone:

Web sites:
rbtam.gcbi.com.au

Require server verification (https:) for all sites in this zone

Figure 4-42 Add Access Manager host to local intranet site zone

3. Type your WebSEAL host name in the Add this Web site to the zone field and click **Add**. Then click **OK** (in our case, the WebSEAL host name is rbtam.gcbi.com.au).
4. You may want to confirm that your configuration is valid. In order to do this task, you can access the Web site you want to check and look at the icon at the corner of the lower right. The icon shown in Figure 4-43 on page 123 shows *Local intranet*.

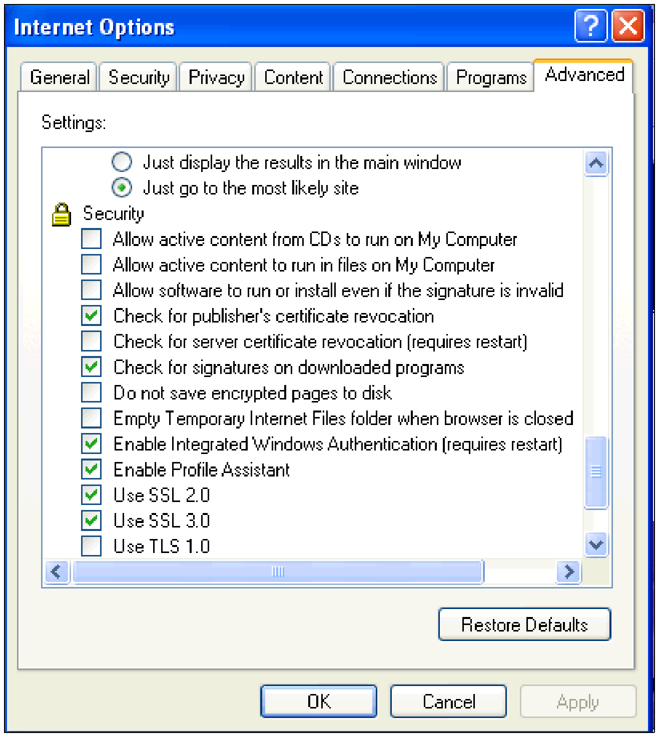

http://rbtam.gcbi.com.au/ - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address http://rbtam.gcbi.com.au/
Go
Links
IBM.
Tivoli.

Access Manager WebSEAL
Version 6.1

Copyright (C) 1994, 2007 by IBM Corporation. All Rights Reserved. Java and all Java-based trademarks are trademarks of SUN Microsystems, Inc. in the United States, other countries, or both.

Figure 4-43 Confirm that Access Manager host is in an intranet zone

5. There is one more option that is needed for a SPNEGO authentication through Internet Explorer. Select **Tools** ∅ **Internet Options** ∅ **Advanced**. In the Settings box, find Enabled Integrated Windows Authentication in the Security section and make sure this parameter is checked, as shown in Figure 4-44. Click **OK** and restart Internet Explorer.



Settings:

- ○ Just display the results in the main window
- ● Just go to the most likely site
- Security
  - ☐ Allow active content from CDs to run on My Computer
  - ☐ Allow active content to run in files on My Computer
  - ☐ Allow software to run or install even if the signature is invalid
  - ✓ Check for publisher's certificate revocation
  - ☐ Check for server certificate revocation (requires restart)
  - ✓ Check for signatures on downloaded programs
  - ☐ Do not save encrypted pages to disk
  - ☐ Empty Temporary Internet Files folder when browser is closed
  - ✓ Enable Integrated Windows Authentication (requires restart)
  - ✓ Enable Profile Assistant
  - ✓ Use SSL 2.0
  - ✓ Use SSL 3.0
  - ☐ Use TLS 1.0

Figure 4-44 Verify that Integrated Windows Authentication is enabled

The configuration steps in this section have to be performed on all machines that use the Access Manager SPNEGO authentication in your environment.

A quick verification test

Now you can check if your configuration is configured correctly. Access the FileNet P8 junction on WebSEAL using Internet Explorer. In our example, the URL is `http://rbtam.gcbi.com.au/Workplace`. If your configuration is correct, you will get the response shown in Figure 4-45 on page 125.

Attention: Once FileNet Workplace or Workplace XT has been configured for SSO with Tivoli Access Manager, the application server URLs should not be accessed directly anymore and all requests have to be routed through the Tivoli Access Manager WebSEAL server. There are several different ways to prevent this; one common way is to configure the application server to allow only requests coming in from the Tivoli Access Manager WebSEAL server.

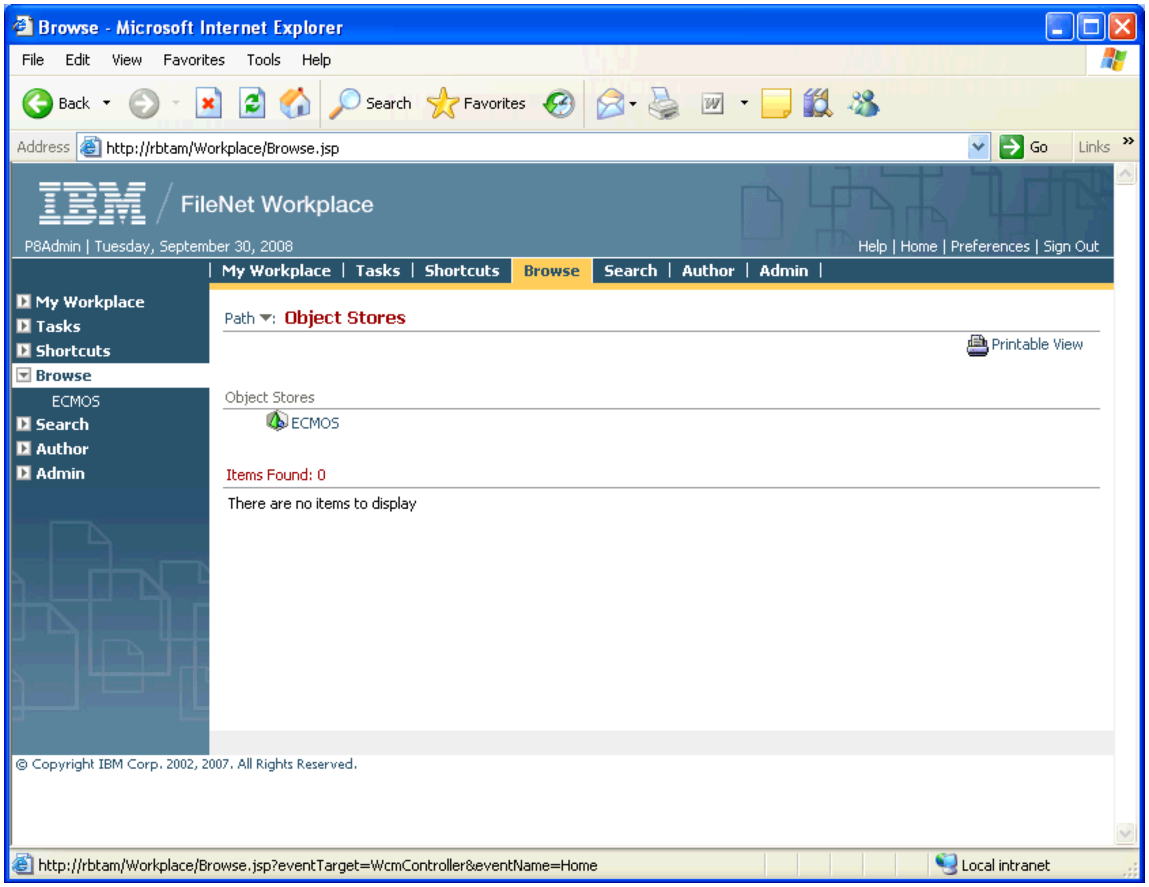

Browse - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://rbtam/Workplace/Browse.jsp Go Links
IBM FileNet Workplace
P8Admin | Tuesday, September 30, 2008
Help | Home | Preferences | Sign Out
My Workplace | Tasks | Shortcuts | Browse | Search | Author | Admin
My Workplace
Tasks
Shortcuts
Browse
ECMOS
Search
Author
Admin
Path Object Stores
Printable View
Object Stores
ECMOS
Items Found: 0
There are no items to display
© Copyright IBM Corp. 2002, 2007. All Rights Reserved.
http://rbtam/Workplace/Browse.jsp?eventTarget=WcmController&eventName=Home
Local intranet

Figure 4-45 FileNet P8 Workplace through Access Manager WebSEAL with SPNEGO desktop SSO

Note: Depending on the specification of your individual FileNet P8 Application Engine, you may get a different window, that is, the bootstrap window for the initial setup of Workplace.

4.3 Integration test

An integration test is very important in the deployment of a reverse proxy type single sign-on product because problems might occur with URL filtering.

The following use cases must be thoroughly tested:

- Create the folder.
- Access the folder.
- Upload the document.
- Download the document.
- Open the admin window in the Workplace application.

4.4 Conclusion

In this chapter, we enabled Tivoli Access Manager for e-business, WebSphere Application Server, and the FileNet P8 Application Engine to deliver a Web-based single sign-on experience for the user. We also showed the SPNEGO configuration in conjunction with Access Manager for e-business so that users only need to authenticate once on their personal Windows workstation while their credentials are then used to seamlessly log them on to the back-end P8 application space.

5

Single sign-on using SPNEGO

In this chapter, we discuss the SPNEGO based single sign-on from a Windows desktop to the FileNet P8 Application Engine running on WebSphere Application Server. With the SPNEGO capabilities provided with WebSphere Application Server, users can seamlessly log on to the FileNet P8 Application Engine without having to provide their credentials other than for the Windows domain logon. This chapter is outlined as follows:

- “GCBI user experience improvement” on page 128
- “Step-by-step configuration” on page 128

An introduction to the SPNEGO architecture has been provided in 2.2, “Simple Protected GSSAPI Negotiation Mechanism architecture overview” on page 35.

Support: In Version 4.0 of IBM FileNet P8, the SPNEGO single sign-on capability is not officially supported through IBM software support. However, SNPEGO based authentication is officially supported as of the IBM FileNet P8 4.5.0 release, for WebSphere and WebLogic. Refer to the appropriate P8 4.5.0 hardware and software product documentation for support details.

5.1 GCBI user experience improvement

In this short section, we want to reiterate GCBI's requirement for an integrated Windows desktop single sign-on into the FileNet P8 Application Engine.

GCBI uses an Active Directory domain to authenticate users on desktop PCs. First, the users log on to the Windows XP operating system using Active Directory authentication, then they have to log on again to the FileNet P8 Application Engine when they use the Web browser. In GCBI, the FileNet P8 registry is Active Directory, so the user uses the same user ID and password for both Windows logon and Application Engine logon.

In order to provide a more comfortable user experience, GCBI would like to implement a single sign-on mechanism that only requires a user logon to the Windows desktop, so that no further credentials have to be provided by the users when they access the FileNet P8 Application Engine.

SPNEGO on WebSphere Application Server is one of the candidates to address this challenge. By using SPNEGO, WebSphere Application Server can retrieve authentication information from the Windows desktop within an encrypted Kerberos ticket and can then verify if the user may be granted access. This approach provides a great improvement in user experience.

5.2 Step-by-step configuration

In this section, we cover the necessary steps to set up single sign-on between IBM FileNet P8 AE and the Windows desktop. The individual sections are outlined below:

- "Prerequisites" on page 129
- "Creating a Kerberos service principal and keytab file" on page 130
- "Creating the Kerberos configuration file" on page 133
- "Configure the SPNEGO TAI interceptor" on page 136
- "Configure SPNEGO TAI interceptor on JVM" on page 141
- "Configure the Web browsers to support SPNEGO" on page 153
- "Quick test for SPNEGO single sign-on" on page 155
- "Troubleshooting SPNEGO" on page 156
- "Deploy the FileNet P8 Application Engine" on page 162
- "Integration test for P8 AE and SPNEGO" on page 178

5.2.1 Prerequisites

Before we describe the configuration steps for SPNEGO SSO in the FileNet P8 Application Engine environment, we need to ensure that a basic system configuration has already been performed:

- The Active Directory domain is configured. Client PCs are configured as members of the Active Directory domain.
- WebSphere Application Server V6.1 is installed and application security is enabled.
- Check that WebSphere Application Server uses the Active Directory on the domain controller as a registry.
- The FileNet P8 Application Engine is installed. During the installation, *Container Managed Authentication* has been selected.
- The FileNet P8 Process Engine and Content Engine are configured and deployed on a different machine than the one that hosts the FileNet P8 Application Engine. As a minimum requirement, you need two machines to deploy SPNEGO based single sign-on scenario for FileNet P8 AE.

We use the following software in the our lab setup:

- Windows Server 2003 Enterprise Edition Service Pack 1 for Access Manager for e-business servers, FileNet P8 AE/CE/PE servers, and domain controller servers
- Windows XP Service Pack 2 for the client machine
- WebSphere Application Server V6.1.0.11
- FileNet P8 Content Engine V.4.0.1-004
- FileNet P8 Process Engine V4.0.2-001
- FileNet P8 Application Engine V4.0.1-004
- DB2 Enterprise Edition V9.5

5.2.2 Creating a Kerberos service principal and keytab file

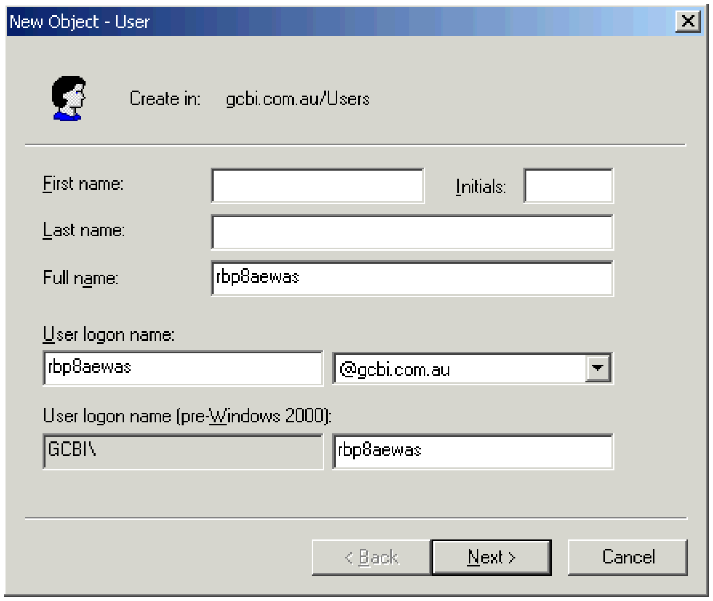
In an SPNEGO configuration, a *Kerberos service principal* is needed on the Web server. In our case, we need a Kerberos service principal for WebSphere Application Server. To create a Kerberos service principal, we first need to create a Windows user account, then create a mapping between the service principal and the Windows user account. This configuration task consists of the following steps:

- "Create a Windows user account" on page 130
- "Create a keytab file" on page 131

Create a Windows user account

To create a Windows user account, do these steps:

1. Open the Microsoft Management Console (MMC) on your Active Directory domain controller. Click **Start** ∅ **Administrative Tools** ∅ **Active Directory Users and Computers.** Next, right-click the appropriate folder (it is typically the User folder) you would like the user belong to and select **New** and then **User**. You should see the window shown in Figure 5-1.



Create in: gcbi.com.au/Users

First name:		Initials:	
Last name:			
Full name:	rbp8aewas		

User logon name: rbp8aewas @gcbi.com.au

User logon name (pre-Windows 2000): GCBI\ rbp8aewas

Figure 5-1 Create the rbp8aewas account on Active Directory domain gcbi.com.au

2. Set the Full name and the User logon name. These names should be different from your host name for WebSphere. In our case, we use rbp8aewas as the

user name. Click **Next**. You should see the window shown in Figure 5-2 on page 131.

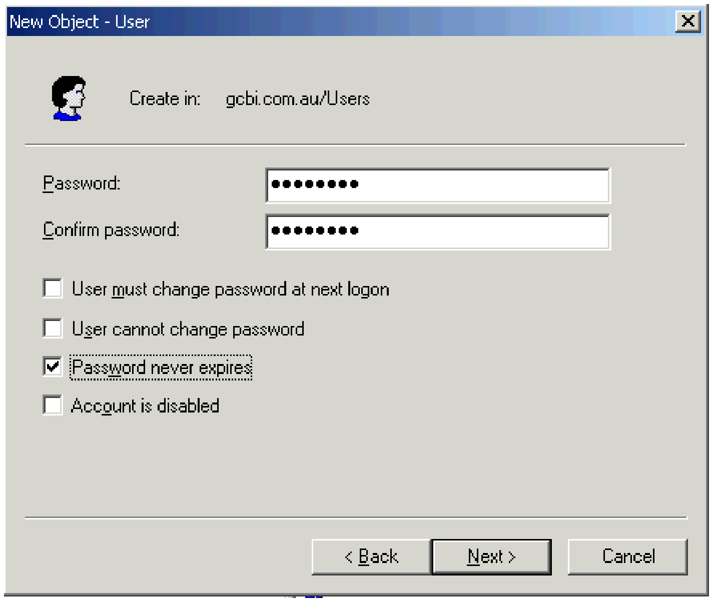

New Object - User
Create in: gcbi.com.au/Users
Password: ●●●●●●●●●
Confirm password: ●●●●●●●●●
☐ User must change password at next logon
☐ User cannot change password
☑ Password never expires
☐ Account is disabled
< Back
Next >
Cancel

Figure 5-2 Create the rbp8aewas account on Active Directory domain gcbi.com.au

3. Enter the password for the account, (we use Passw0rd). This password is needed in future configuration steps. After filling out the Password and the Confirm password fields, click **Next**, and then click **Finish** in the next window.

Create a keytab file

After the account has been created, we need to map the account to the Kerberos service principal name (SPN) and create a keytab file. We can use the **`ktpass`** command for the mapping and creation of the keytab file on the Windows domain controller machine.

Note: The **`ktpass`** command is provided as a part of the Windows support tools. To use the **`ktpass`** command, you need to install the Windows support tools from the Windows Server 2003 installation media before you can use the command.

Do these steps:

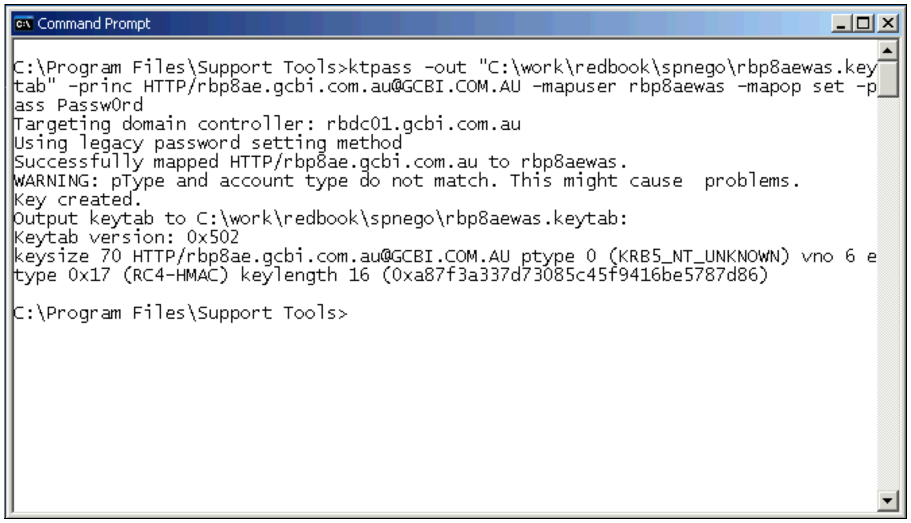
1. Open a command prompt window on the domain controller machine and enter the **ktpass** command as follows:

   ```
   ktpass -out rbp8aewas.keytab -princ
   HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU -mapuser rbp8aewas -pass
   Passw0rd
   ```

   Where:

-out	The file name of the keytab produced by the **ktpass** command.
-princ	The principal name for the Web server. It consists of three elements. The first one is always "HTTP/". The second element is the host name of the Web server as a fully qualified domain name. The third element is "@" followed by the Active Directory domain name. The host name part does not need to match the Windows account name.
-mapuser	This represents the user that is mapped to the principal. This user has to exist in Active Directory. In our example, we use Windows user account rbp8aewas that we created.
-pass	The password for the user shown above.

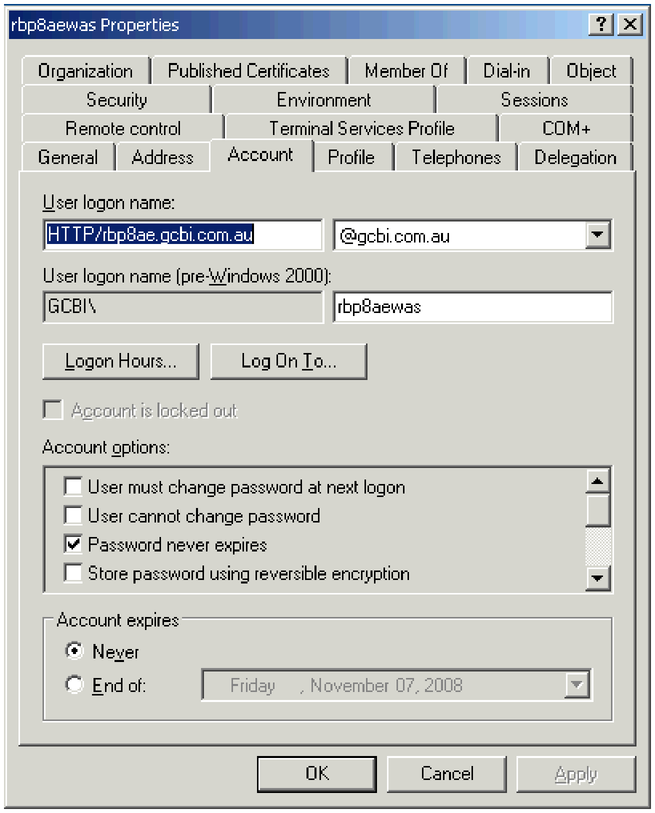
   The actual command and response is shown in Figure 5-3.



```
C:\Program Files\Support Tools>ktpass -out "C:\work\redbook\spnego\rbp8aewas.key
tab" -princ HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU -mapuser rbp8aewas -mapop set -p
ass Passw0rd
Targeting domain controller: rbdc01.gcbi.com.au
Using legacy password setting method
Successfully mapped HTTP/rbp8ae.gcbi.com.au to rbp8aewas.
WARNING: pType and account type do not match. This might cause  problems.
Key created.
Output keytab to C:\work\redbook\spnego\rbp8aewas.keytab:
Keytab version: 0x502
keysize 70 HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU ptype 0 (KRB5_NT_UNKNOWN) vno 6 e
type 0x17 (RC4-HMAC) keylength 16 (0xa87f3a337d73085c45f9416be5787d86)

C:\Program Files\Support Tools>
```

Figure 5-3 An example of the ktpass command

2. After executing the **ktpass** command, the User logon name field is changed to the service principal name HTTP/rbp8ae.gcbi.com.au (originally it was rbp8aewas), as shown in Figure 5-4.



User logon name: HTTP/rbp8ae.gcbi.com.au @gcbi.com.au

User logon name (pre-Windows 2000): GCBI\ rbp8aewas

☐ Account is locked out

Account options:
- ☐ User must change password at next logon
- ☐ User cannot change password
- ☑ Password never expires
- ☐ Store password using reversible encryption

Account expires
- ● Never
- ○ End of: Friday , November 07, 2008

Figure 5-4 Account properties after executing the ktpass command

3. Finally, copy the keytab file produced by the **ktpass** command (in our example, the rbp8aewas.keytab file) to the appropriate directory on the WebSphere Application Server machine.

5.2.3 Creating the Kerberos configuration file

If you want to deploy your WebSphere Application Server on a UNIX machine, you need to install the Kerberos client software for the UNIX operating system. If you have already installed a Kerberos client, you have all the necessary Kerberos skeleton configuration files on your machine. You can begin to create the Kerberos configuration file by using the sample skeleton configuration file.

If you have deployed WebSphere Application Server on Windows, you do not need to install any Kerberos client software, but there are no sample Kerberos configuration files available on Windows. However, WebSphere Application Server offers a function to create a Kerberos configuration file for the JVM. You can use this function to create your Kerberos configuration file.

Do these steps:

1. Example 5-1 shows the appropriate **`wsadmin`** commands needed to create the Kerberos configuration file. Before executing the **`wsadmin`** command, you need to open a Windows command prompt and navigate to the `C:\Program Files\IBM\WebSphere\AppServer\bin` directory. There you execute the **`setupCmdLine.bat`** command. This command sets the necessary environment variables to properly execute WebSphere Application Server commands, including the **`wsadmin`** command.

   > **Note:** If you have an installation with multiple profiles you will need to specify the profile name when you execute the command:
   >
   > `C:\IBM\WebSphere\AppServer\bin>setupCmdLine.bat -profileName AppSrv01`

   After executing the **`wsadmin`** command, you need to enter the information that is displayed in bold in Example 5-1.

   Example 5-1 wsadmin commands to create the Kerberos configuration file

```
wsadmin>$AdminTask createKrbConfigFile -interactive
Create Kerberos configuration file

This command creates a Kerberos configuration file (krb5.ini or
krb5.conf)

*Filesystem location of the Kerberos configuration file (krbpath):
C:\Windows\krb5.ini
*Kerberos realm name in kerberos configuration file (realm):
GCBI.COM.AU
*Host name of the Kerberos Key Distribution Center (kdcHost):
rbdc01.gcbi.com.au
Port number of the Kerberos Key Dstribution Center (kdcPort):
*Default name of the Domain Name Service (dns): gcbi.com.au
*Filesystem location of the keytab file (keytabPath):
C:\Progra~1\IBM\WebSphere\etc\rbp8aewas.keytab
Encryption type (encryption):

Create Kerberos configuration file
```

```
F (Finish)
C (Cancel)

Select [F, C]: [F]f
WASX7278I: Generated command line: $AdminTask createKrbConfigFile
{-krbPath C:\Windows\krb5.ini -realm GCBI.COM.AU -kdcHost
rbdc01.gcbi.com.au -keytabPath
C:\Progra~1\IBM\WebSphere\AppServer\etc\rbp8aewas.keytab}
C:\Windows\krb5.ini has been created.
wsadmin>exit
```

Example 5-2 shows an example of the `krb5.ini` file.

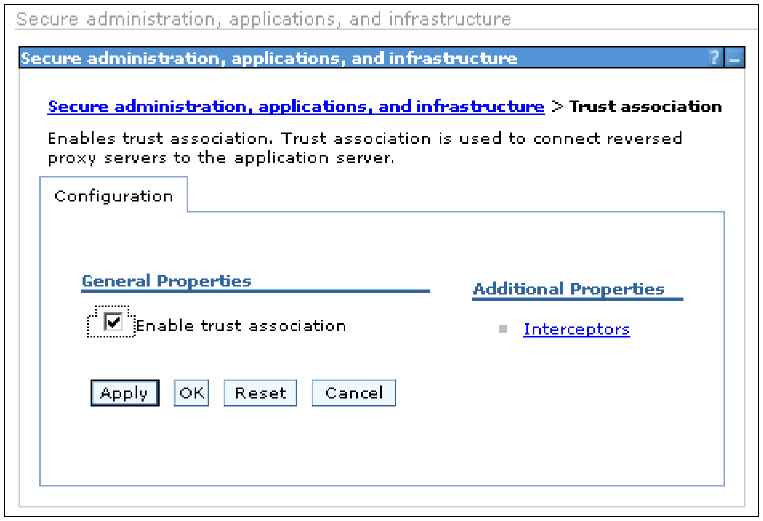
Example 5-2 Kerberos client configuration file

```
[libdefaults]
   default_realm = GCBI.COM.AU
   default_keytab_name =
FILE:C:\Progra~1\IBM\WebSphere\AppServer\etc\rbp8aewas.keytab
   default_tkt_enctypes = des-cbc-md5 rc4-hmac
   default_tgs_enctypes = des-cbc-md5 rc4-hmac
   kdc_default_options = 0x54800000
#  forwardable  = true
#  proxiable  = true
#  noaddresses = true
[realms]
   GCBI.COM.AU = {
      kdc = rbdc01.gcbi.com.au:88
      default_domain = gcbi.com.au
   }
[domain_realm]
   .gcbi.com.au = GCBI.COM.AU
```

5.2.4 Configure the SPNEGO TAI interceptor

In our example, we assume that the security and registry configuration has already been finished on the WebSphere Application Server. In order to configure the SPNEGO TAI interceptor for your WebSphere Application Server, follow these steps:

1. Open the IBM Integrated Solutions Console (ISC) using the URL `https://your WebSphere host:9060/ibm/console`. Select **Security** ∅ **Secure administration application, and infrastructure**. Click **Trust association** in the right pane. You should see the window shown in Figure 5-5.



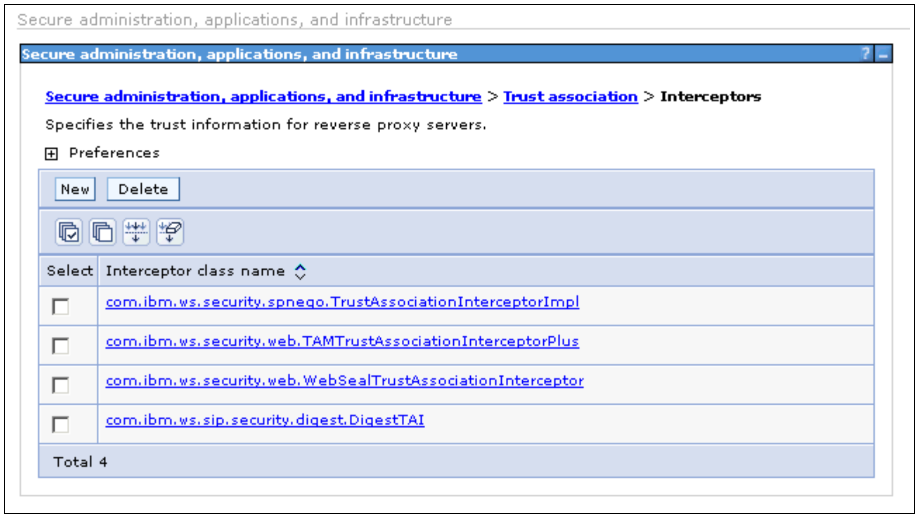
Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association

Enables trust association. Trust association is used to connect reversed proxy servers to the application server.

Figure 5-5 Trust association setting

2. Check **Enable trust association**, then click **OK** and click **Save**.
3. Go to the Trust association pane again and click **Interceptors**. You should see the window shown in Figure 5-6 on page 137.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association > Interceptors

Specifies the trust information for reverse proxy servers.

⊞ Preferences

New Delete

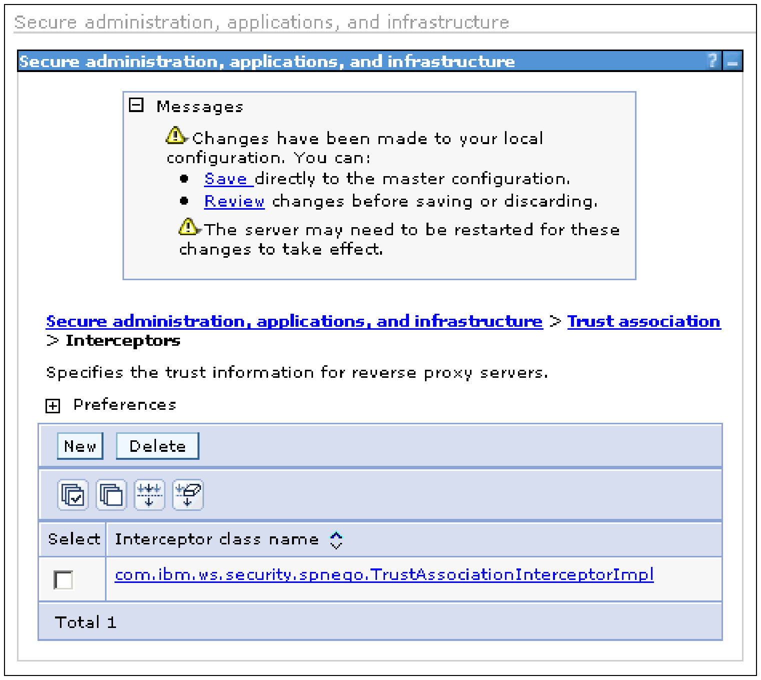
Select	Interceptor class name
☐	com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl
☐	com.ibm.ws.security.web.TAMTrustAssociationInterceptorPlus
☐	com.ibm.ws.security.web.WebSealTrustAssociationInterceptor
☐	com.ibm.ws.sip.security.digest.DigestTAI

Total 4

Figure 5-6 The list of default WebSphere Application Server interceptors

4. In WebSphere Application Server, there are four default interceptors. It can be a good idea to delete interceptors other than the SPNEGO interceptor because interceptors that are not properly configured can create wrong error messages in the WebSphere log files.

Figure 5-7 shows the window after deleting unnecessary interceptors.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

Messages

⚠ Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

⚠ The server may need to be restarted for these changes to take effect.

Secure administration, applications, and infrastructure > Trust association > **Interceptors**

Specifies the trust information for reverse proxy servers.

Preferences

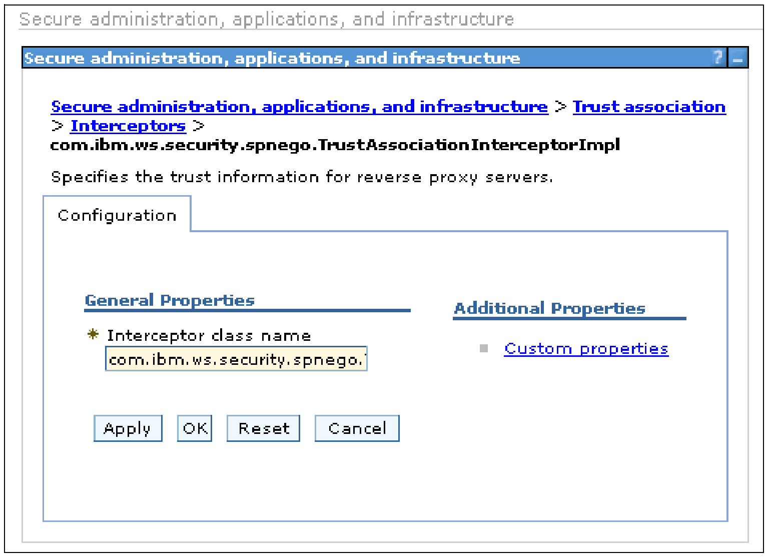
New Delete

Select	Interceptor class name
☐	com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl

Total 1

Figure 5-7 Interceptor list after deleting unnecessary ones

5. Click **Save**. Then click the **com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl** link to edit its properties. You should see the window shown in Figure 5-8.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association > Interceptors > com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl

Specifies the trust information for reverse proxy servers.

Configuration

General Properties

* Interceptor class name

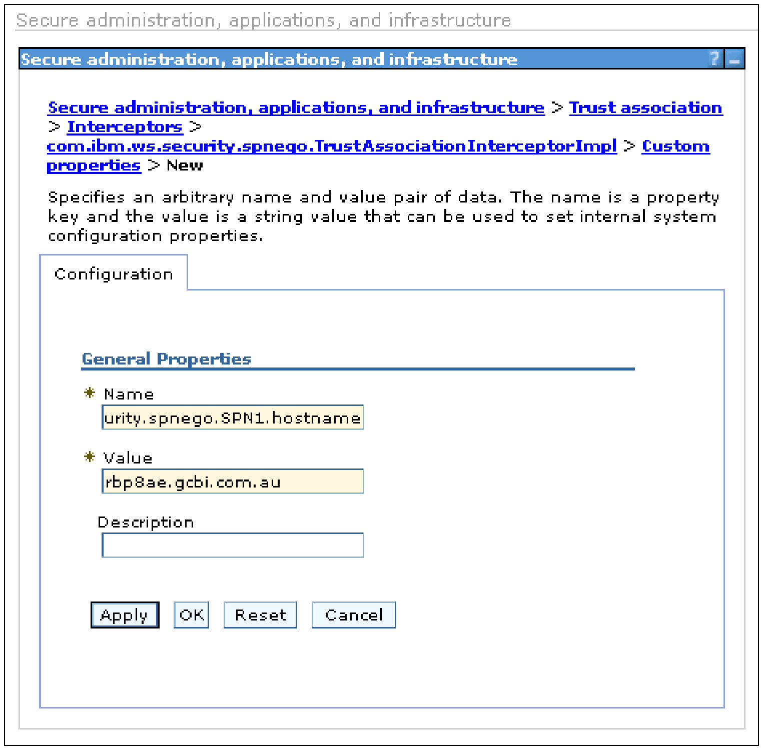
com.ibm.ws.security.spnego.

Additional Properties

- Custom properties

Figure 5-8 SPNEGO interceptor setting

6. You do not need to modify any properties in this window, so click **Custom properties**. Because there are no configured properties shown in the next window, click **New** to create a new property. You will see the window shown in Figure 5-9.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association > Interceptors > com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl > Custom properties > New

Specifies an arbitrary name and value pair of data. The name is a property key and the value is a string value that can be used to set internal system configuration properties.

* Name: urity.spnego.SPN1.hostname

* Value: rbp8ae.gcbi.com.au

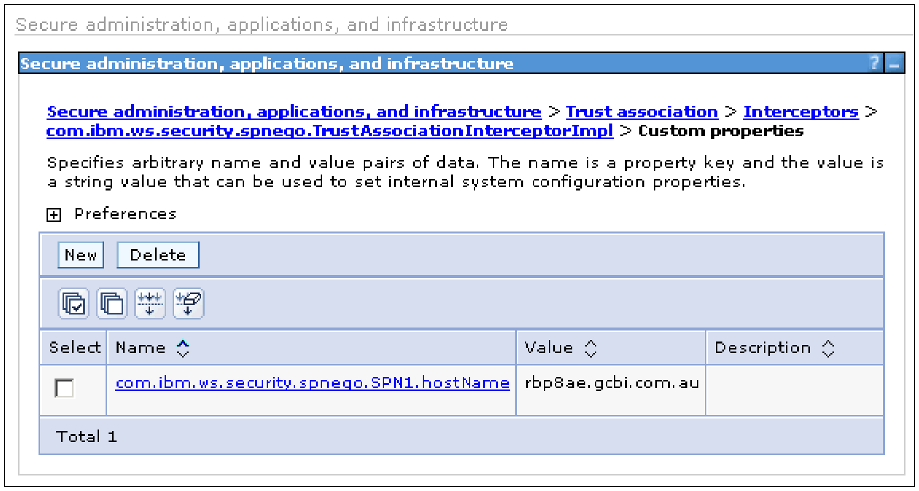
Description:

Figure 5-9 Custom properties for SPNEGO interceptor

7. Enter com.ibm.security.spnego.SPN1.hostname into the Name field. Type your FileNet P8 Application Engine's host name into the Value field; in our example, we use rbp8ae.gcbi.com.au. Click **OK** and click **Save**.

Note: The number we have specified in the SPN name value (in our example, it is 1) allows us to configure multiple SPNEGO interceptor configurations on a single WebSphere Application Server instance. In other words, if you have configured multiple virtual hosts on your WebSphere Application Server instance, you can configure a different SPNEGO interceptor setting for each virtual host.

After you have clicked **Save**, you should see the final window shown in Figure 5-10 on page 141.


Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure

Secure administration, applications, and infrastructure > Trust association > Interceptors > com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl > Custom properties

Specifies arbitrary name and value pairs of data. The name is a property key and the value is a string value that can be used to set internal system configuration properties.

⊞ Preferences

Select	Name	Value	Description
☐	com.ibm.ws.security.spnego.SPN1.hostName	rbp8ae.gcbi.com.au	
Total 1			

Figure 5-10 Custom properties are set in SPNEGO interceptor configuration window

5.2.5 Configure SPNEGO TAI interceptor on JVM

To enable the SPNEGO TAI interceptor on WebSphere Application Server, you must, in addition to doing the interceptor configuration for WebSphere Application Server (described in 5.2.4, “Configure the SPNEGO TAI interceptor” on page 136), to do a configuration on the JVM of your WebSphere Application Server. This process consists of the following steps:

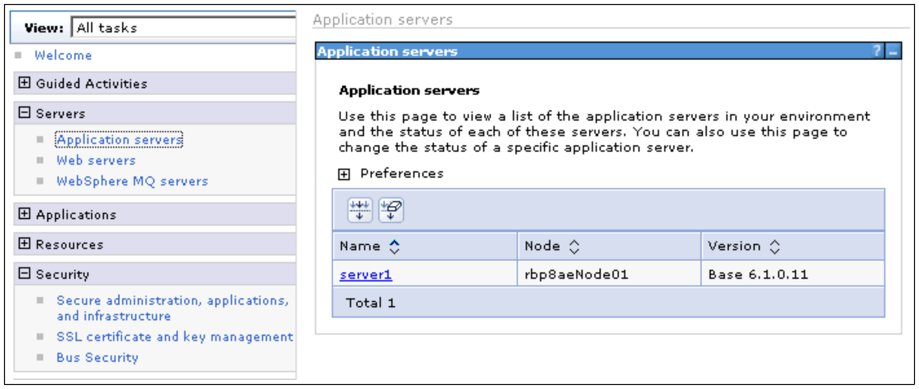
- “Prepare SPNEGO TAI interceptor in WebSphere Application Server” on page 141
- “Enable SPNEGO on JVM” on page 148

Prepare SPNEGO TAI interceptor in WebSphere Application Server

Do these steps:

1. Open the ISC in your Web browser. INn our scenarios, we use the URL `https://rbp8ae.gcbi.com.au:9060/ibm/console`.

2. Select **Servers** ∅ **Application Servers** in the left pane of the window. You should see the window shown in Figure 5-11.



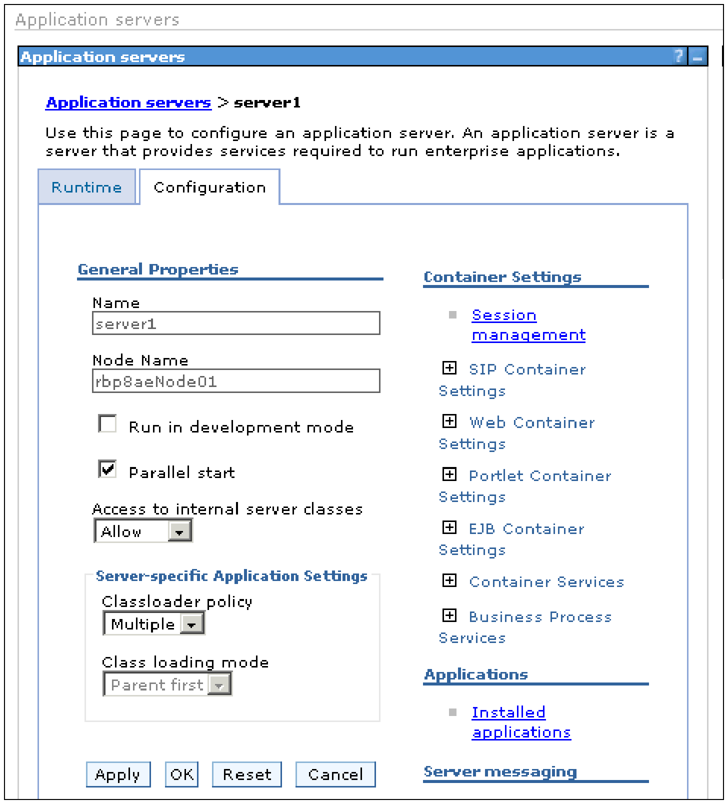
Application servers

Use this page to view a list of the application servers in your environment and the status of each of these servers. You can also use this page to change the status of a specific application server.

Name	Node	Version
server1	rbp8aeNode01	Base 6.1.0.11
Total 1		

Figure 5-11 Application server setting

3. Select the **server1** link. Make sure you navigate to the Configuration tab. You should see the window shown in Figure 5-12.



Application servers

Application servers

Application servers > **server1**

Use this page to configure an application server. An application server is a server that provides services required to run enterprise applications.

Runtime | Configuration

General Properties

Name: server1

Node Name: rbp8aeNode01

☐ Run in development mode

☑ Parallel start

Access to internal server classes: Allow

Server-specific Application Settings

Classloader policy: Multiple

Class loading mode: Parent first

Apply | OK | Reset | Cancel

Container Settings

- Session management
- ⊞ SIP Container Settings
- ⊞ Web Container Settings
- ⊞ Portlet Container Settings
- ⊞ EJB Container Settings
- ⊞ Container Services
- ⊞ Business Process Services

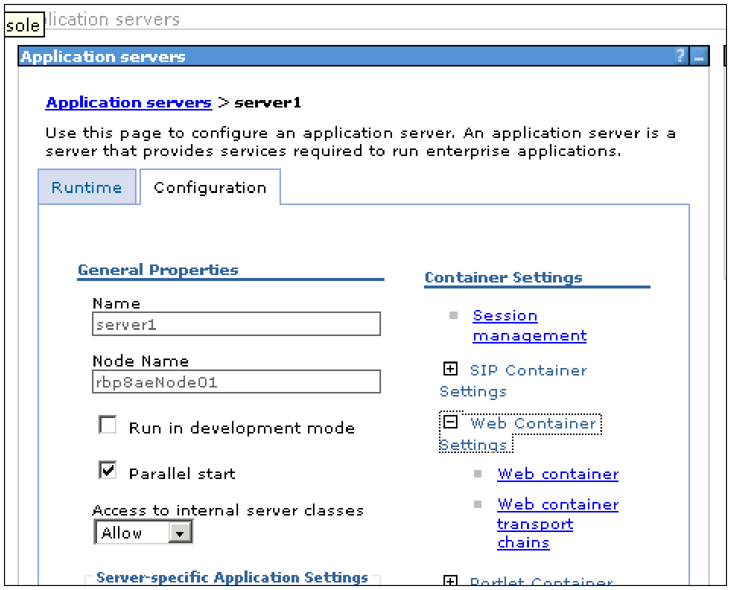
Applications

- Installed applications

Server messaging

Figure 5-12   Application Server setting

4. When you click the **+** mark for the **Web Container Settings**, some properties appear, as shown in Figure 5-13.



sole lication servers

Application servers

Application servers > server1

Use this page to configure an application server. An application server is a server that provides services required to run enterprise applications.

Runtime | Configuration

General Properties

Name
server1

Node Name
rbp8aeNode01

☐ Run in development mode

☑ Parallel start

Access to internal server classes
Allow

Server-specific Application Settings

Container Settings

- Session management

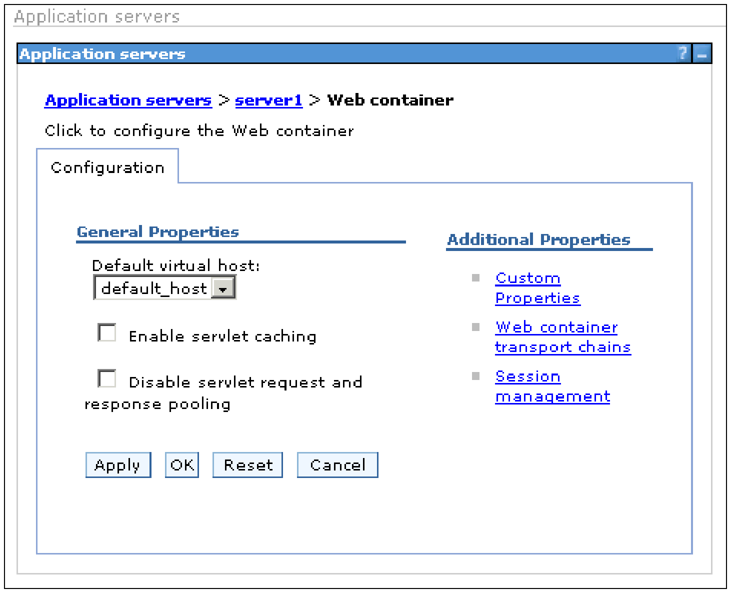
⊞ SIP Container Settings

⊟ Web Container Settings
  - Web container
  - Web container transport chains

⊞ Portlet Container

Figure 5-13 Web Container Settings menu

5. Click the **Web container** link (found under the Web Container Settings). You should see the window shown in Figure 5-14 on page 145.



Application servers > server1 > Web container

Click to configure the Web container

Configuration

General Properties

Default virtual host:
default_host

☐ Enable servlet caching

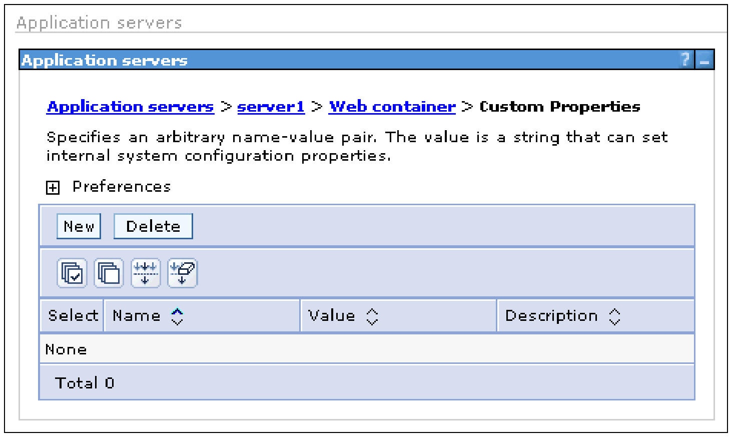
☐ Disable servlet request and response pooling

Additional Properties

- Custom Properties
- Web container transport chains
- Session management

Figure 5-14 Web container setting

6. Click **Custom Properties**. You should see the window shown in Figure 5-15.



Application servers > server1 > Web container > Custom Properties

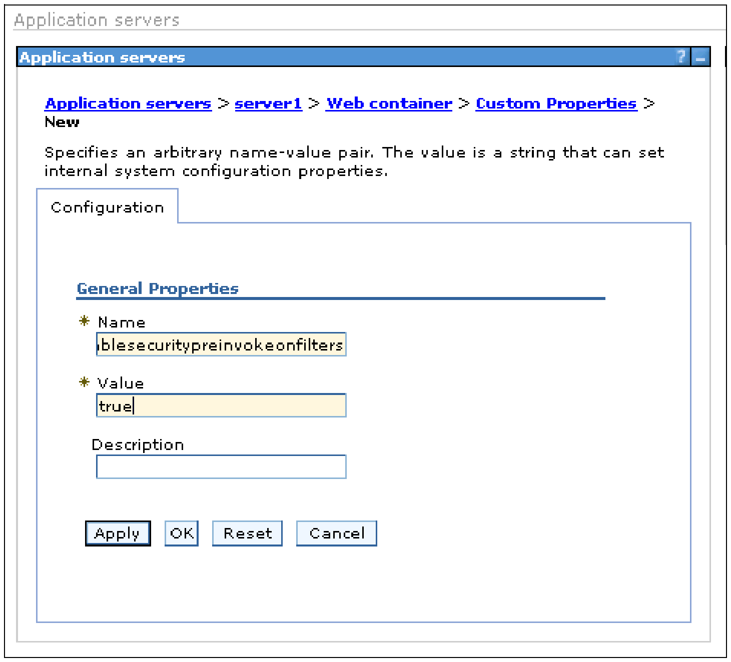
Specifies an arbitrary name-value pair. The value is a string that can set internal system configuration properties.

⊞ Preferences

Select	Name	Value	Description
None			
Total 0			

Figure 5-15 Custom properties for Web container

7. Click **New** to add new settings for the Custom Properties. You should see the window shown in Figure 5-16.


Application servers

Application servers

Application servers > server1 > Web container > Custom Properties > New

Specifies an arbitrary name-value pair. The value is a string that can set internal system configuration properties.

Configuration

General Properties

* Name
ıblesecuritypreinvokeonfilters

* Value
true

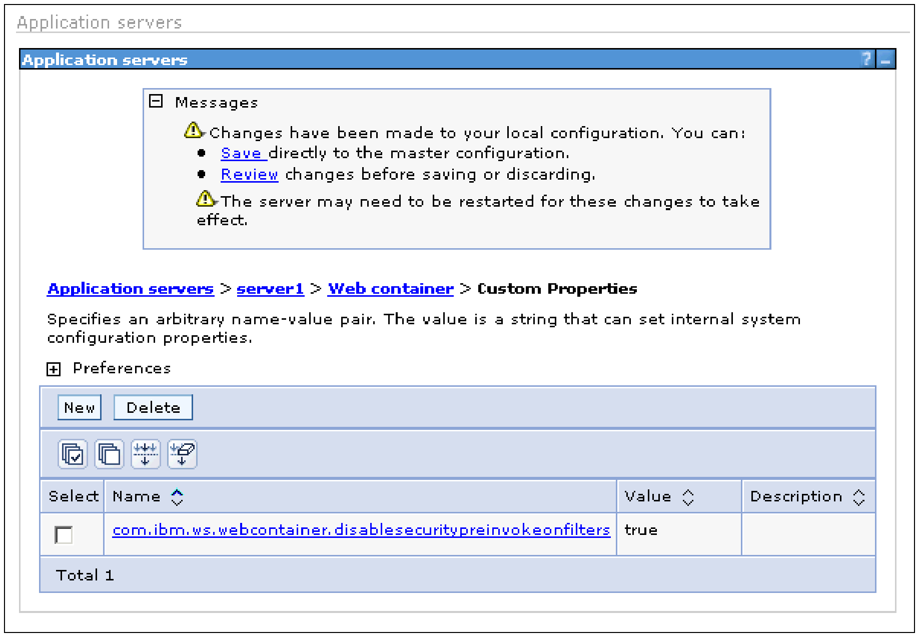
Description

Apply OK Reset Cancel

Figure 5-16 Custom Properties window for Web container

8. Enter com.ibm.ws.webcontainer.disablesecuritypreinvokeonfilters into the Name field and true into the Value field. Click **OK.** You should see the window shown in Figure 5-17 on page 147.

> **Note:** This particular setting is available with WebSphere Application Server Fix Pack V6.1.0.11 or later. This property needs to be set in order to allow Web applications that normally use a forms login page (such as the WebSphere Application Server administration console) to authenticate using SPNEGO correctly.


Application servers

Application servers

Messages

⚠ Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

⚠ The server may need to be restarted for these changes to take effect.

Application servers > server1 > Web container > Custom Properties

Specifies an arbitrary name-value pair. The value is a string that can set internal system configuration properties.

Preferences

New | Delete

Select	Name	Value	Description
☐	com.ibm.ws.webcontainer.disablesecuritypreinvokeonfilters	true	
Total 1			

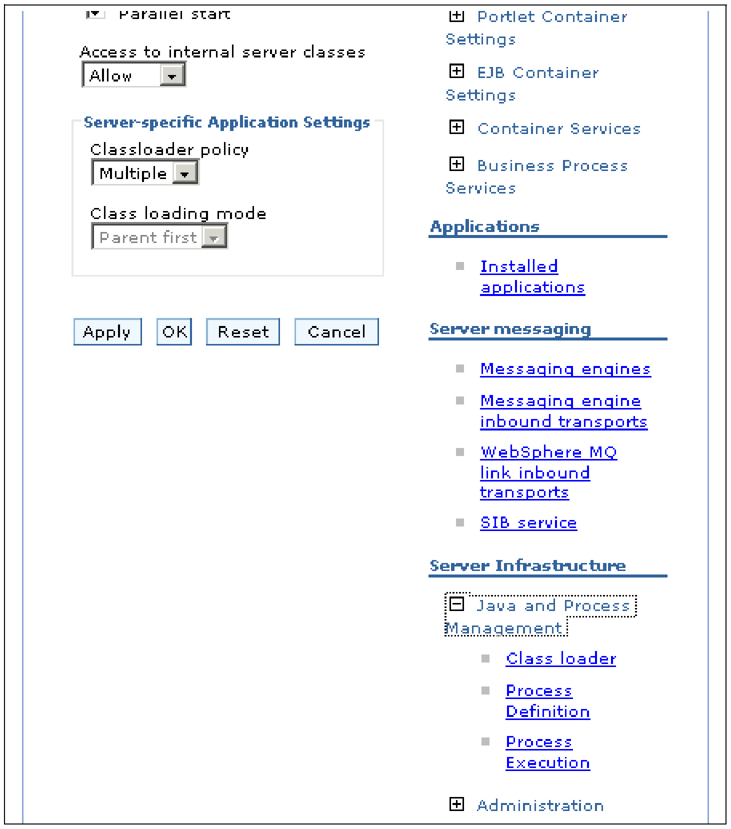
Figure 5-17 Mandatory properties set on the Web container

9. Click **Save** to finish the configuration.

Enable SPNEGO on JVM

Here we show how to enable SPNEGO on JVM. We assume that you still have the ISC opened from the previous step. Do these steps:

1. Select **Application Servers** ∅ **server1**. You should see the window shown in Figure 5-18.



Parallel start

Access to internal server classes: Allow

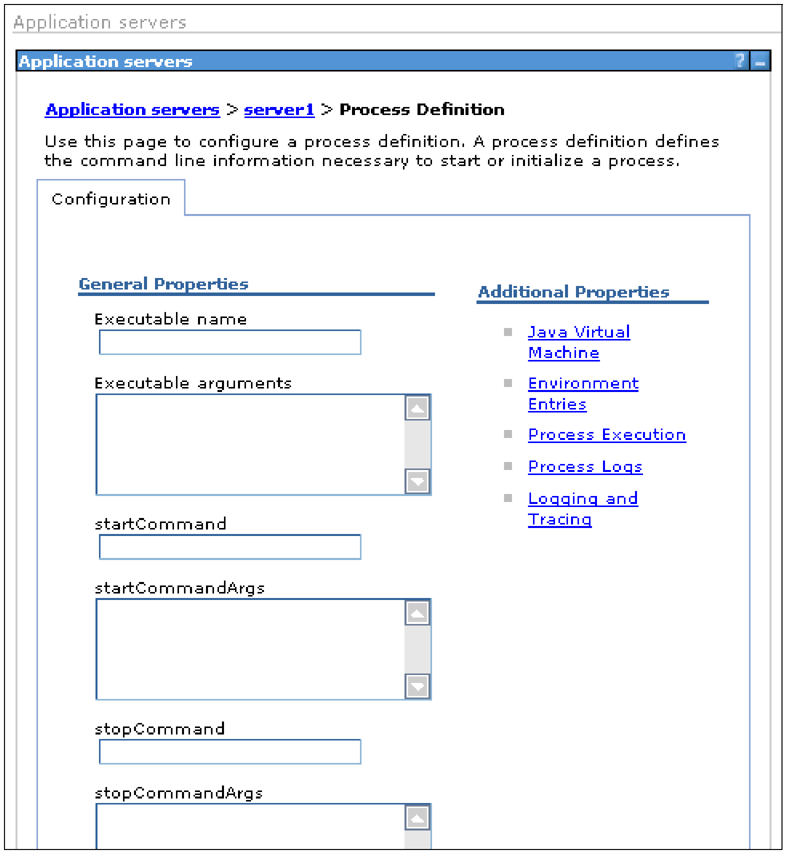
Server-specific Application Settings

Classloader policy: Multiple

Class loading mode: Parent first

Figure 5-18 Server setting

2. Click **Java and Process Management** to expand the list. Then click **Process Definition**. You should see the window shown in Figure 5-19.


Application servers

Application servers

Application servers > server1 > Process Definition

Use this page to configure a process definition. A process definition defines the command line information necessary to start or initialize a process.

Configuration

General Properties

Executable name

Executable arguments

startCommand

startCommandArgs

stopCommand

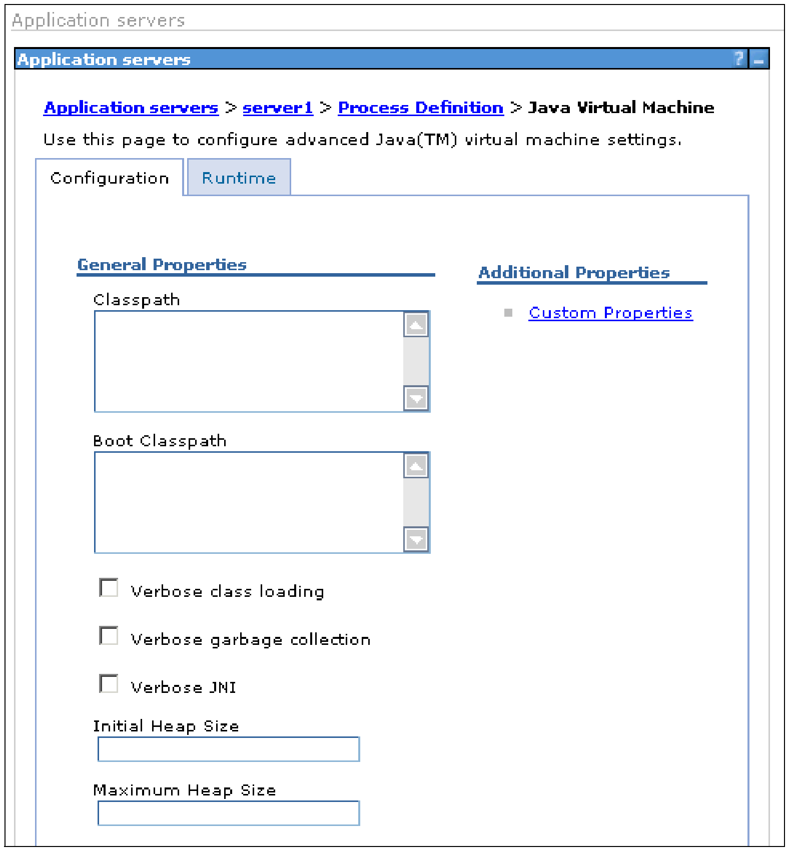
stopCommandArgs

Additional Properties

- Java Virtual Machine
- Environment Entries
- Process Execution
- Process Logs
- Logging and Tracing

Figure 5-19 Process definition

3. Click **Java Virtual Machine**. You should see the window shown in Figure 5-20.


Application servers

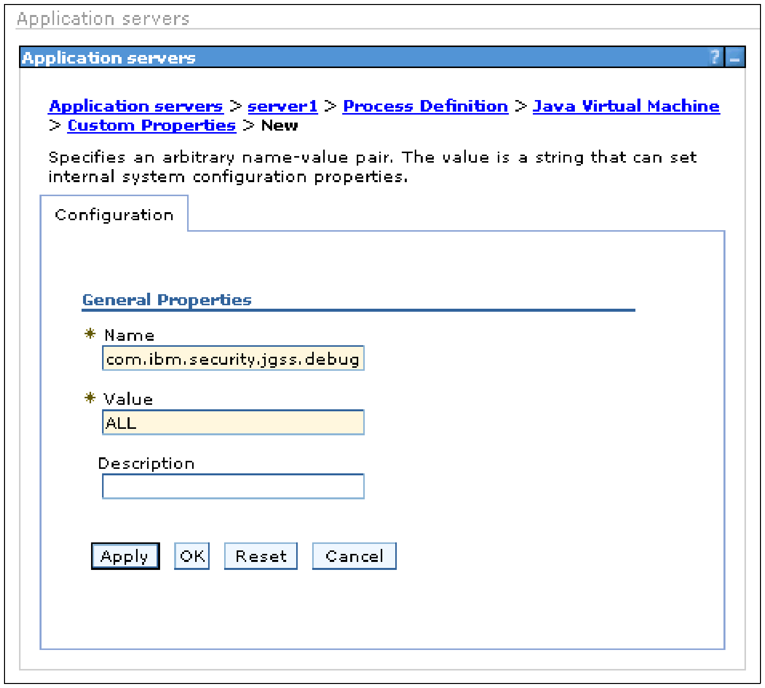
Application servers

Application servers > server1 > Process Definition > Java Virtual Machine

Use this page to configure advanced Java(TM) virtual machine settings.

Figure 5-20 JVM settings

4. Click **Custom Properties**. Again, there is no property entry defined yet. Click **New**, and you should see the window shown in Figure 5-21 on page 151.


Application servers

Application servers

Application servers > server1 > Process Definition > Java Virtual Machine > Custom Properties > New

Specifies an arbitrary name-value pair. The value is a string that can set internal system configuration properties.

Configuration

General Properties

* Name
com.ibm.security.jgss.debug

* Value
ALL

Description

Figure 5-21 New property for JVM

5. Enter com.ibm.security.jgss.debug into the Name field and ALL into the Value field. Click **OK** and click **Save**. In the same way, you should enter the four properties displayed in Table 5-1.

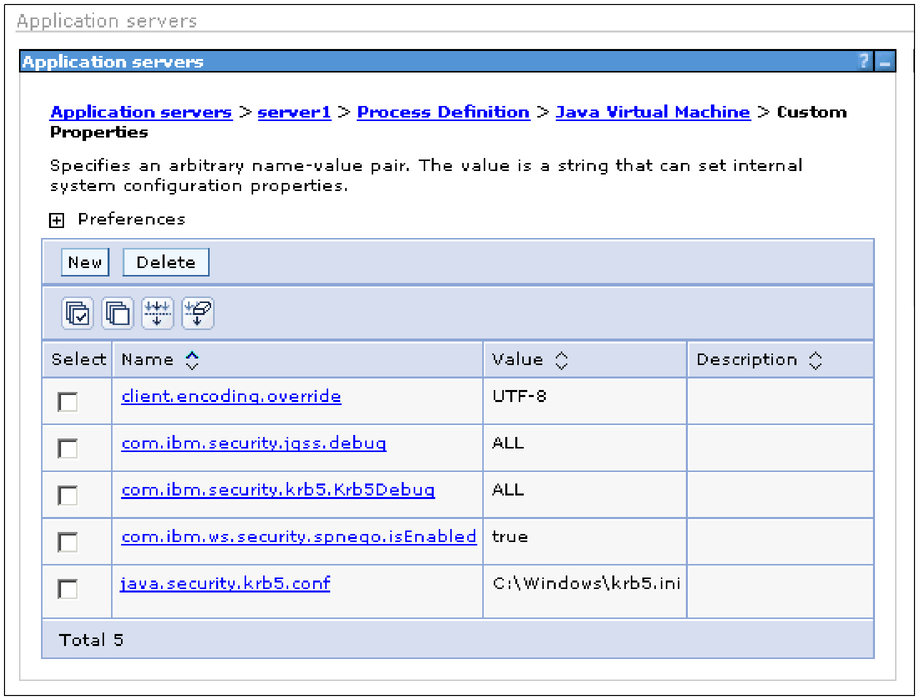
Table 5-1 JVM properties for SPNEGO

Name	Value	Mandatory?
com.ibm.security.krb.Krb5Debug	ALL	No
com.ibm.ws.security.spnego.isEnabled	true	Yes
java.security.krb5.conf	C:\windows\krb5.ini	No
client.encoding.override	UTF-8	No

Note:

- The property names are case sensitive.
- You should enable the debug settings because they are particularly useful for testing in the configuration phase, but they can impact performance because they create many messages in the log files. In a production system, we recommend you disable the debug settings. To disable the debug settings, you should delete the com.ibm.security.krb.Krb5Debug property itself, not just delete the property's value.

Figure 5-22 shows all configured custom properties in our example.



Application servers

Application servers

Application servers > server1 > Process Definition > Java Virtual Machine > Custom Properties

Specifies an arbitrary name-value pair. The value is a string that can set internal system configuration properties.

⊞ Preferences

Select	Name	Value	Description
☐	client.encoding.override	UTF-8	
☐	com.ibm.security.jgss.debug	ALL	
☐	com.ibm.security.krb5.Krb5Debug	ALL	
☐	com.ibm.ws.security.spnego.isEnabled	true	
☐	java.security.krb5.conf	C:\Windows\krb5.ini	

Total 5

Figure 5-22 Five properties set in the JVM setting

6. After all properties are configured and saved, restart WebSphere Application Server. Now the SPNEGO configuration on the server side is complete.

5.2.6 Configure the Web browsers to support SPNEGO

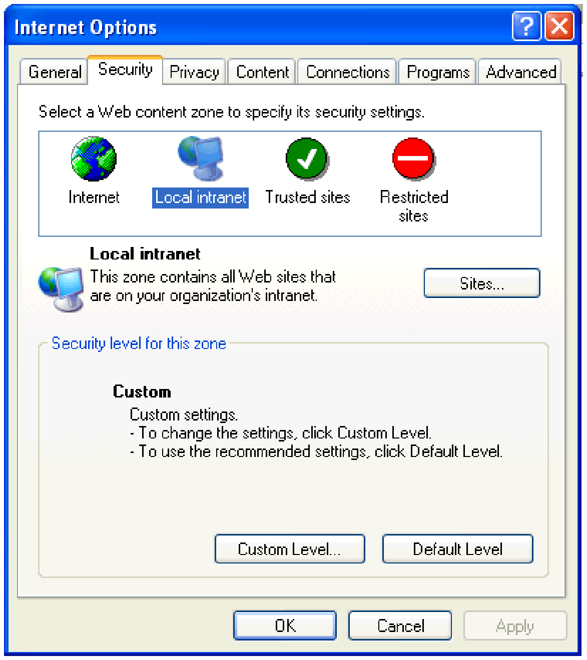
Next, we need to properly configure the Web browser in order to support SPNEGO on the client side. The configuration discussed in this section has to be performed on all client PCs, and consists of the following steps:

- "Add P8AE server host name into the Local intranet zone" on page 153
- "Enable integrated Windows authentication" on page 155

Add P8AE server host name into the Local intranet zone

Do these steps:

1. Start Internet Explorer on the client PC. Click **Tools** on the menu bar and select **Internet Options**. Then select the **Security** tab. You should see the window shown in Figure 5-23.



Select a Web content zone to specify its security settings.

Local intranet

This zone contains all Web sites that are on your organization's intranet.

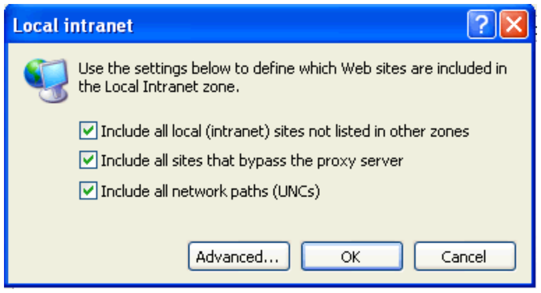
Security level for this zone

Custom

Custom settings.
- To change the settings, click Custom Level.
- To use the recommended settings, click Default Level.

Figure 5-23 Security tab in Internet Explorer internet options

2. Select the **Local intranet** zone, then click **Sites**. You should see the window shown in Figure 5-24.

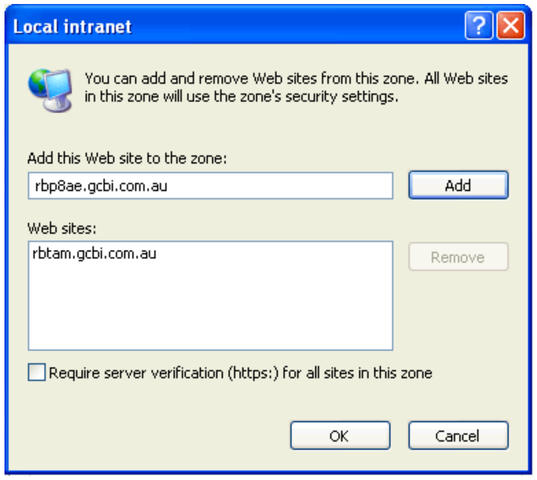


Use the settings below to define which Web sites are included in the Local Intranet zone.

☑ Include all local (intranet) sites not listed in other zones
☑ Include all sites that bypass the proxy server
☑ Include all network paths (UNCs)

Figure 5-24 Property for Local intranet

3. Click **Advanced**. You should see the window shown in Figure 5-25.



You can add and remove Web sites from this zone. All Web sites in this zone will use the zone's security settings.

Add this Web site to the zone: rbp8ae.gcbi.com.au

Web sites: rbtam.gcbi.com.au

☐ Require server verification (https:) for all sites in this zone

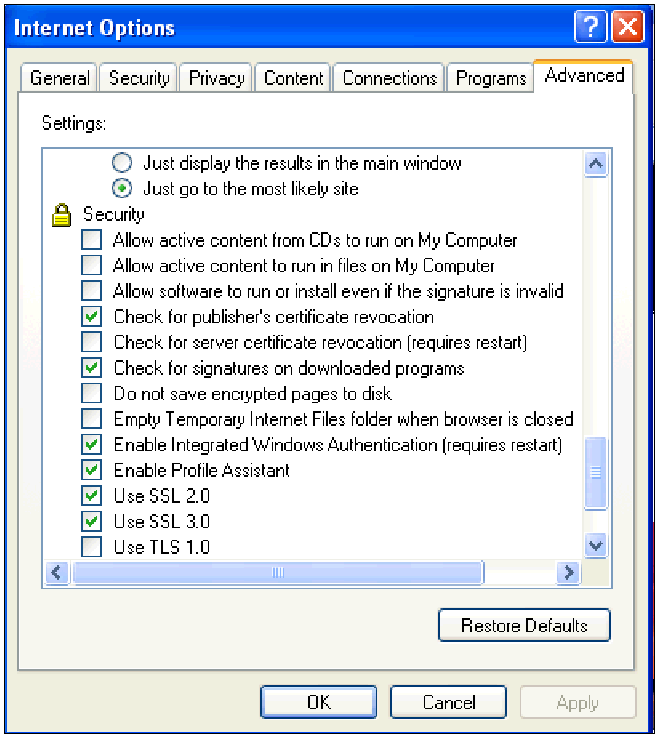
Figure 5-25 Add WebSphere host name to Local internet zone

4. Enter the FileNet P8 AE host name (in our example, it is rbp8ae.gcbi.com.au) into the Add this Web site to the zone field and click **Add**. Then click **OK**. Keep your Internet Options window open in Internet Explorer and continue to the next section.

Enable integrated Windows authentication

Do these steps:

1. Select the **Advanced** tab in the Internet Options window. You should see the window shown in Figure 5-26. Confirm that the **Enable Integrated Windows Authentication** option is checked. If it is not checked, check it. Click **OK** to finish this configuration step.



Internet Options

General | Security | Privacy | Content | Connections | Programs | Advanced

Settings:

- ○ Just display the results in the main window
- ● Just go to the most likely site
- Security
  - ☐ Allow active content from CDs to run on My Computer
  - ☐ Allow active content to run in files on My Computer
  - ☐ Allow software to run or install even if the signature is invalid
  - ☑ Check for publisher's certificate revocation
  - ☐ Check for server certificate revocation (requires restart)
  - ☑ Check for signatures on downloaded programs
  - ☐ Do not save encrypted pages to disk
  - ☐ Empty Temporary Internet Files folder when browser is closed
  - ☑ Enable Integrated Windows Authentication (requires restart)
  - ☑ Enable Profile Assistant
  - ☑ Use SSL 2.0
  - ☑ Use SSL 3.0
  - ☐ Use TLS 1.0

Restore Defaults

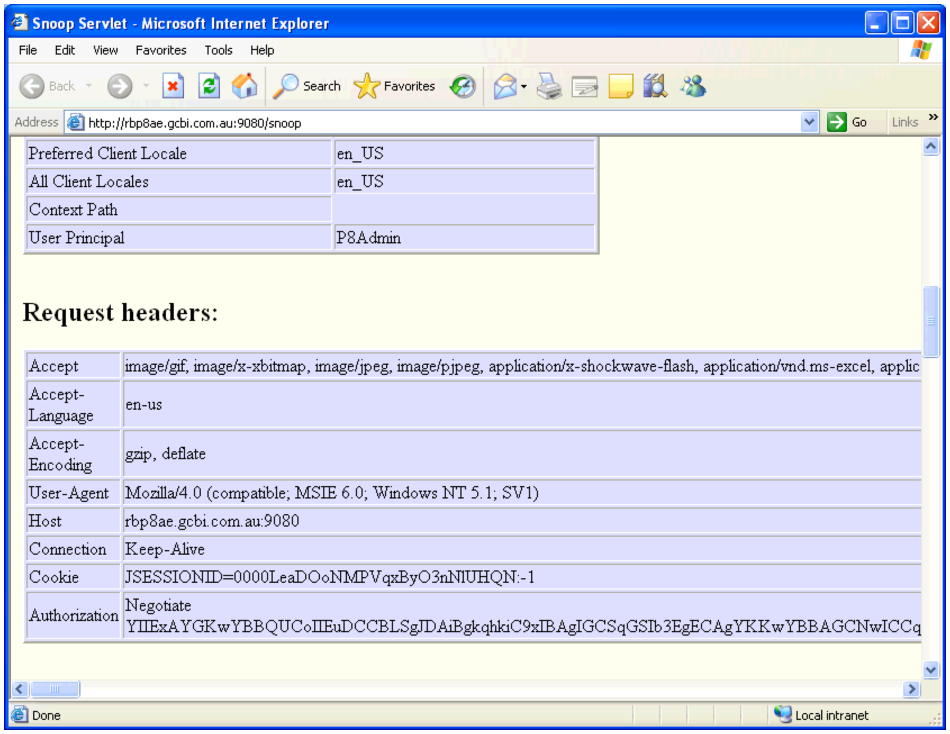
OK | Cancel | Apply

Figure 5-26 Enable Integrated Windows Authentication

5.2.7 Quick test for SPNEGO single sign-on

In this section, we introduce a quick way to confirm that your configuration has been performed correctly.

You can confirm that your SPNEGO configuration works correctly by accessing the snoop servlet on WebSphere Application Server. If you get a response from the snoop servlet, as shown in Figure 5-27, with the user principal and the negotiate string in the authorization header, you have been successfully authenticated by the SPNEGO TAI interceptor.


Snoop Servlet - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://rbp8ae.gcbi.com.au:9080/snoop
Go Links
Done
Local intranet

Preferred Client Locale	en_US
All Client Locales	en_US
Context Path	
User Principal	P8Admin

Request headers:

Accept	image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, applic
Accept-Language	en-us
Accept-Encoding	gzip, deflate
User-Agent	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)
Host	rbp8ae.gcbi.com.au:9080
Connection	Keep-Alive
Cookie	JSESSIONID=0000LeaDOoNMPVqxByO3nNIUHQN:-1
Authorization	Negotiate YIIExAYGKwYBBQUCoIIEuDCCBLSgJDAiBgkqhkiC9xIBAgIGCSqGSIb3EgECAgYKKwYBBAGCNwICCq

Figure 5-27 An example of snoop servlet response

5.2.8 Troubleshooting SPNEGO

If your configuration does not work correctly, you have to check your configuration and diagnose the cause of the problem by using the log files that are introduced in this section.

This section consists of:

- “Confirm successful initialization of interceptor” on page 157
- “Check points for successful SPNEGO working” on page 159

Confirm successful initialization of interceptor

If you defined the debug properties for the WebSphere Application Server JVM, as shown in Table 5-1 on page 151, you should get messages in the WebSphere Application Server `SystemOut.log` file, as shown in Example 5-3. The `Systemout.log` file is located in the `C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\logs\server1` directory.

To verify if your SPNEGO configuration has successfully initialized, look for the `CWSPN0006I: SPNEGO Trust Association Interceptor initialization is complete` message in the last part of the `SystemOut.log` example shown in Example 5-3.

If the initialization failed, you need to check your Kerberos configuration on both the WebSphere Application Server and Windows Active Directory side again.

Example 5-3 SPNEGO initialization messages in systemout.log

```
[10/8/08 18:07:14:286 PDT] 0000000a TrustAssociat A   SECJ0121I: Trust Association Init class
com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl loaded successfully
[10/8/08 18:07:14:334 PDT] 0000000a SystemOut     O [KRB_DBG_CFG] Config:main: Config name:
c:\winnt\krb5.ini
[10/8/08 18:07:14:399 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Number of system providers=8
[10/8/08 18:07:14:399 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechOidFromProperty: mech oid
string = 1.2.840.113554.1.2.2
[10/8/08 18:07:14:399 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechOidFromProperty: mech oid
string = 1.3.6.1.5.5.2
[10/8/08 18:07:14:416 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] 2 system providers found/added
[10/8/08 18:07:14:416 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechOidFromProperty: mech oid
string = 1.2.840.113554.1.2.2
[10/8/08 18:07:14:416 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechs: Mechanism(s) supported
by provider IBMJGSSProvider
[10/8/08 18:07:14:416 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] 1.2.840.113554.1.2.2
[10/8/08 18:07:14:416 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechOidFromProperty: mech oid
string = 1.3.6.1.5.5.2
[10/8/08 18:07:14:416 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechs: Mechanism(s) supported
by provider IBMSPNEGO
[10/8/08 18:07:14:432 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] 1.3.6.1.5.5.2
[10/8/08 18:07:14:432 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getMechs: 2 unique mechanism(s)
found
[10/8/08 18:07:14:432 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] [0]: 1.2.840.113554.1.2.2
[10/8/08 18:07:14:432 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] [1]: 1.3.6.1.5.5.2
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] MN not found; creating one
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Created new (empty) factory list
(size=1) for provider IBMJGSSProvider version 1.5
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Loading factory
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Factory class name for provider
IBMJGSSProvider version 1.5 is com.ibm.security.jgss.mech.krb5.Krb5MechFactory
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Loaded factory for provider
IBMJGSSProvider version 1.5
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Loaded factory ok
[10/8/08 18:07:14:448 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getFactory: index = 0 found
factory
[10/8/08 18:07:14:464 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Name cannonicalization complete,
resulting name string=HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU
[10/8/08 18:07:14:480 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Krb5 name type = 1
[10/8/08 18:07:14:497 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Name is not canonicalized for
mech 1.3.6.1.5.5.2, creating mech name
[10/8/08 18:07:14:497 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Krb5 name type = 1
[10/8/08 18:07:14:497 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Provider IBMJGSSProvider version
1.5 does not support mech 1.3.6.1.5.5.2
```

```
[10/8/08 18:07:14:497 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Created new (empty) factory list
(size=1) for provider IBMSPNEGO version 1.0
[10/8/08 18:07:14:497 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Loading factory
[10/8/08 18:07:14:497 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Factory class name for provider
IBMSPNEGO version 1.0 is com.ibm.security.jgss.mech.spnego.SPNEGOMechFactory
[10/8/08 18:07:14:513 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Loaded factory for provider
IBMSPNEGO version 1.0
[10/8/08 18:07:14:513 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Loaded factory ok
[10/8/08 18:07:14:513 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getFactory: index = 1 found
factory
[10/8/08 18:07:14:513 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Creating mech cred for
HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU, mech 1.3.6.1.5.5.2, usage accept only
[10/8/08 18:07:14:513 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] Provider IBMJGSSProvider version
1.5 does not support mech 1.3.6.1.5.5.2
[10/8/08 18:07:14:513 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getFactory: index = 1 found
factory
[10/8/08 18:07:14:529 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Adding mech cred
[10/8/08 18:07:14:529 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Creating mech cred for
HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU, mech 1.2.840.113554.1.2.2, usage accept only
[10/8/08 18:07:14:529 PDT] 0000000a SystemOut     O [JGSS_DBG_PROV] getFactory: index = 0 found
factory
[10/8/08 18:07:14:545 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Obtaining creds from keytab for
service HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KTAB] KeyTab:main: >>> KeyTab: load()
entry length: 70
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KTAB] KeyTableInputStream:main: >>>
KeyTabInputStream, readName(): GCBI.COM.AU
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KTAB] KeyTableInputStream:main: >>>
KeyTabInputStream, readName(): HTTP
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KTAB] KeyTableInputStream:main: >>>
KeyTabInputStream, readName(): rbp8ae.gcbi.com.au
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KDC] EncryptionKey:main: >>>
EncryptionKey: config default key type is rc4-hmac
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KDC] Credentials:main: >>> Credentials:
Created Credentials with 1 keys. Key types:
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KDC] Credentials:main: [1] rc4-hmac
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Got server creds for service
HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [KRB_DBG_KDC] Credentials:main:Client
Name:HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU
[10/8/08 18:07:14:578 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Got creds for
HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU
[10/8/08 18:07:14:594 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Kerberos ticket endtime is null,
using default max ticket lifetime: 86400 secs
[10/8/08 18:07:14:594 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Requested acceptLifetime:
2147483647 secs
[10/8/08 18:07:14:594 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Starttime: (1223514434 secs) Wed
Oct 08 18:07:14 PDT 2008
[10/8/08 18:07:14:594 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] AcceptLifetime: indefinite
[10/8/08 18:07:14:594 PDT] 0000000a SystemOut     O [JGSS_DBG_CRED] Adding mech cred
[10/8/08 18:07:14:594 PDT] 0000000a ServerCredent I
com.ibm.ws.security.spnego.ServerCredentialsFactory initializeServer CWSPN0016I: Ready to process
host: rbp8ae.gcbi.com.au.
[10/8/08 18:07:14:594 PDT] 0000000a TrustAssociat I
com.ibm.ws.security.spnego.TrustAssociationInterceptorImpl initialize CWSPN0006I: SPNEGO Trust
Association Interceptor initialization is complete. Configuration follows:
    TAI configuration (JVM) properties:
    com.ibm.ws.security.spnego.isEnabled=true
    Server configuration:
    Kerberos ServicePrincipalName=HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU
    com.ibm.ws.security.spnego.SPN.filter=null
    com.ibm.ws.security.spnego.SPN.filterClass=com.ibm.ws.security.spnego.HTTPHeaderFilter@35623562
    com.ibm.ws.security.spnego.SPN.NTLMTokenReceivedPage=null
    com.ibm.ws.security.spnego.SPN.spnegoNotSupportedPage=null
```

If the SPNEGO TAI interceptor initialization has failed, try to issue the **kinit** command on your WebSphere Application Server machine to make sure that your Kerberos configuration is valid. The **kinit** command is located in the `C:\Program Files\IBM\WebSphere\AppServer\bin` directory of your Windows operating system. Example 5-4 shows an example of the **kinit** command and its response. Characters in bold symbolize what you need to type in.

Example 5-4 An example of kinit command

```
C:\Documents and Settings\Administrator> cd C:\Program
Files\IBM\WebSphere\AppServer\java\jre\bin
C:\Program Files\IBM\WebSphere\AppServer\java\jre\bin>.\kinit -k
HTTP/rbp8ae.gcbi.com.au
Done!
New ticket is stored in cache file C:\Documents and
Settings\Administrator\krb5cc_Administrator
```

If there is an error message in the response of the **kinit** command, you have an improper configuration of either your Kerberos client or server.

Note: By default, the JVM assumes that the Kerberos client configuration file (`krb5.ini`) is located in `C:\Winnt`. In our configuration steps, we used the `C:\Windows` directory for the Kerberos client configuration file. Is this case, the **kinit** command produces an error that indicates `C:\Winnt\krb5.ini is not found`. To resolve this error, you can simply copy the `krb5.ini` file from `C:\Windows` to `C:\Winnt`.

Check points for successful SPNEGO working

Here are some check points to review when your configuration does not work correctly.

- Do you use the FQDN host name in an URL?

  The host name part in an URL should match your service principal name configuration and host name set in WebSphere Application Server. If the URL does not match your configuration, the SPNEGO authentication does not work, and an HTTP basic authentication prompt might appear.

- Does the **kinit** command returns a `success` response?

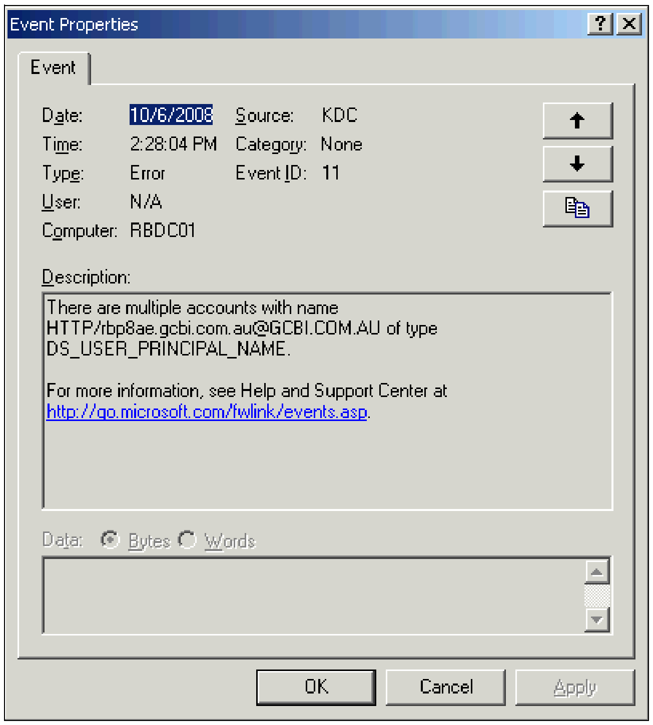
  If the **kinit** command returns an error, your SPNEGO TAI Interceptor does not work correctly, because the Interceptor could not be initialized without getting a Ticket Granting Ticket (TGT).

- Are there any duplicate SPN entries on your Active Directory?

  If you mistakenly executed the **ktpass** command using an inappropriate Windows user account, you must delete the mapping by using the **setspn** command. Here is an example of the **setspn** command:

  ```
  setspn -D HTTP/rbp8aewas.gcbi.com.au Wrongusername
  ```

  However, even if you delete the mapping information using the **setspn** command and execute the **ktpass** command again with the correct Windows user name, there will be some garbage and an unlinked entry left over in your Active Directory. This entry is responsible for the *duplicated SPN* errors. In this situation, the **kinit** command fails and the SPNEGO TAI Interceptor cannot be initialized. Figure 5-28 shows an example of this error, which can be seen in the security category of the event viewer on the Domain Controller machine.



Date: 10/6/2008 Source: KDC
Time: 2:28:04 PM Category: None
Type: Error Event ID: 11
User: N/A
Computer: RBDC01

Description:
There are multiple accounts with name HTTP/rbp8ae.gcbi.com.au@GCBI.COM.AU of type DS_USER_PRINCIPAL_NAME.

For more information, see Help and Support Center at http://go.microsoft.com/fwlink/events.asp.

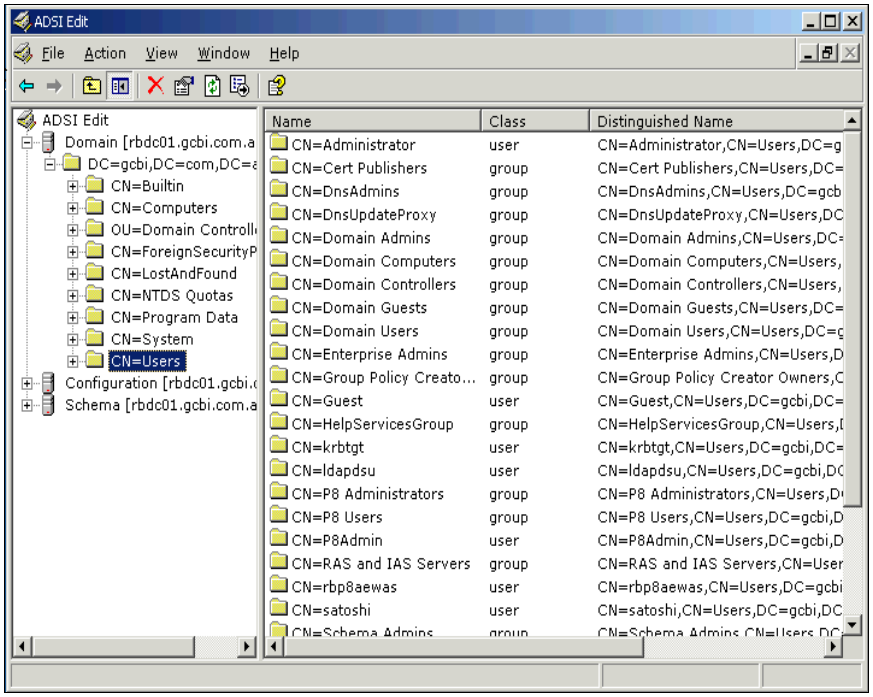
Data: [✓ Bytes] Words

Figure 5-28 Duplicated SPN error

If you encounter this problem, you must remove duplicated attributes from Active Directory manually. Here are the steps to remove inappropriate principal names from Active Directory:

a. Open the ADSI snap-in on the domain controller machine. Click **Start** ∅ **Run**, enter adsiedit.msc, and then click **OK**.

b. The ADSI snap-in starts up, as shown in Figure 5-29. Select the user that you mistakenly used to create the mapping.

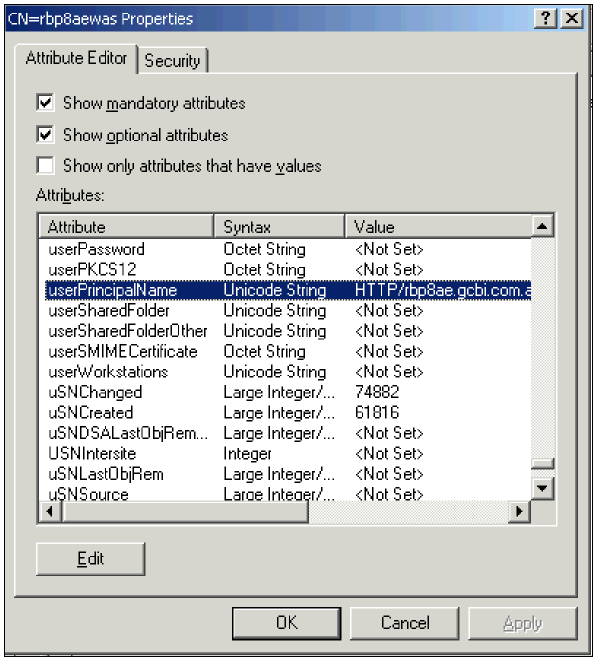


- ADSI Edit
  - Domain [rbdc01.gcbi.com.a
    - DC=gcbi,DC=com,DC=a
      - CN=Builtin
      - CN=Computers
      - OU=Domain Controlli
      - CN=ForeignSecurityP
      - CN=LostAndFound
      - CN=NTDS Quotas
      - CN=Program Data
      - CN=System
      - CN=Users
  - Configuration [rbdc01.gcbi.c
  - Schema [rbdc01.gcbi.com.a

Name	Class	Distinguished Name
CN=Administrator	user	CN=Administrator,CN=Users,DC=g
CN=Cert Publishers	group	CN=Cert Publishers,CN=Users,DC=
CN=DnsAdmins	group	CN=DnsAdmins,CN=Users,DC=gcb
CN=DnsUpdateProxy	group	CN=DnsUpdateProxy,CN=Users,DC
CN=Domain Admins	group	CN=Domain Admins,CN=Users,DC=
CN=Domain Computers	group	CN=Domain Computers,CN=Users,
CN=Domain Controllers	group	CN=Domain Controllers,CN=Users,
CN=Domain Guests	group	CN=Domain Guests,CN=Users,DC=
CN=Domain Users	group	CN=Domain Users,CN=Users,DC=g
CN=Enterprise Admins	group	CN=Enterprise Admins,CN=Users,D
CN=Group Policy Creato...	group	CN=Group Policy Creator Owners,C
CN=Guest	user	CN=Guest,CN=Users,DC=gcbi,DC=
CN=HelpServicesGroup	group	CN=HelpServicesGroup,CN=Users,I
CN=krbtgt	user	CN=krbtgt,CN=Users,DC=gcbi,DC=
CN=ldapdsu	user	CN=ldapdsu,CN=Users,DC=gcbi,DC
CN=P8 Administrators	group	CN=P8 Administrators,CN=Users,D
CN=P8 Users	group	CN=P8 Users,CN=Users,DC=gcbi,D
CN=P8Admin	user	CN=P8Admin,CN=Users,DC=gcbi,D
CN=RAS and IAS Servers	group	CN=RAS and IAS Servers,CN=User
CN=rbp8aewas	user	CN=rbp8aewas,CN=Users,DC=gcbi
CN=satoshi	user	CN=satoshi,CN=Users,DC=gcbi,DC
CN=Schema Admins	group	CN=Schema Admins,CN=Users,DC=

Figure 5-29 ADSI Edit

c. Right-click the user and select **Properties**. You should see the window shown in Figure 5-30.


CN=rbp8aewas Properties

Attribute Editor | Security

☑ Show mandatory attributes
☑ Show optional attributes
☐ Show only attributes that have values

Attributes:

Attribute	Syntax	Value
userPassword	Octet String	<Not Set>
userPKCS12	Octet String	<Not Set>
userPrincipalName	Unicode String	HTTP/rbp8ae.gcbi.com.a
userSharedFolder	Unicode String	<Not Set>
userSharedFolderOther	Unicode String	<Not Set>
userSMIMECertificate	Octet String	<Not Set>
userWorkstations	Unicode String	<Not Set>
uSNChanged	Large Integer/...	74882
uSNCreated	Large Integer/...	61816
uSNDSALastObjRem...	Large Integer/...	<Not Set>
USNIntersite	Integer	<Not Set>
uSNLastObjRem	Large Integer/...	<Not Set>
uSNSource	Large Integer/...	<Not Set>

Figure 5-30 Attribute list on Active Directory entry

d. Locate the userPrincipalName Attribute that shows the matching principal name value you have used. If it exists, you can delete it by clicking **Edit**. and, when the attribute editor comes up, click **Clear** and click **OK**.

5.2.9 Deploy the FileNet P8 Application Engine

In the next configuration step, we deploy the FileNet P8 Application Engine into the SPNEGO enabled WebSphere Application Server. This configuration step consists of:

- “Modify deployment descriptor files” on page 162
- “Create WAR file and EAR file” on page 168
- “Deploy the application to WebSphere Application Server” on page 169

Modify deployment descriptor files

Before deploying the application, you have to modify some deployment descriptor files for the FileNet P8 Application Engine server.

Note: The deployment descriptor files introduced in this section should be considered examples. Do not copy and paste these files to your environment without closely reviewing the files yourself. For example, there are absolute directory path definitions in the `web.xml` file. In our example, these paths use Windows style representation because we use a Windows-based server for FileNet P8 AE. If you use UNIX servers for FileNet P8 AE, the `web.xml` file shown here needs to be modified.

- `web.xml`

  This file is located in the `C:\Program Files\FileNet\AE\Workplace\WEB-INFO` directory on our Windows machine.

  a. Open the file with a text editor and locate the following section (Example 5-5) in the file and uncomment the section.

Example 5-5 web.xml

```
<!-- Uncomment this section if all resources that require
credentials must be secured in order to obtain a secured Thread.
If using WebSphere, this section must be uncommented.
<url-pattern>/</url-pattern>
<url-pattern>/author/*</url-pattern>
<url-pattern>/Browse.jsp</url-pattern>
<url-pattern>/eprocess/*</url-pattern>
<url-pattern>/Favorites.jsp</url-pattern>
<url-pattern>/GetPortalSitePreferences.jsp</url-pattern>
<url-pattern>/GetTokenSignIn.jsp</url-pattern>
<url-pattern>/GetUserInformation.jsp</url-pattern>
<url-pattern>/GetUserToken.jsp</url-pattern>
<url-pattern>/HomePage.jsp</url-pattern>
<url-pattern>/IntegrationWebBasedHelp.jsp</url-pattern>
<url-pattern>/is/*</url-pattern>
<url-pattern>/operations/*</url-pattern>
<url-pattern>/portlets/Author/edit.jsp</url-pattern>
<url-pattern>/portlets/Author/portlet.jsp</url-pattern>
<url-pattern>/portlets/Browse/edit.jsp</url-pattern>
<url-pattern>/portlets/Browse/portlet.jsp</url-pattern>
<url-pattern>/portlets/ExternalUrl/edit.jsp</url-pattern>
<url-pattern>/portlets/ExternalUrl/portlet.jsp</url-pattern>
<url-pattern>/portlets/GroupPageDesign.jsp</url-pattern>
<url-pattern>/portlets/GroupPageSettings.jsp</url-pattern>
<url-pattern>/portlets/Inbox/edit.jsp</url-pattern>
<url-pattern>/portlets/Inbox/portlet.jsp</url-pattern>
<url-pattern>/portlets/MultiPagesDesign.jsp</url-pattern>
```

```
<url-pattern>/portlets/OrganizePages.jsp</url-pattern>
<url-pattern>/portlets/PortalPageDesign.jsp</url-pattern>
<url-pattern>/portlets/PortalPageInfo.jsp</url-pattern>
<url-pattern>/portlets/PortletAlias.jsp</url-pattern>
<url-pattern>/portlets/PortletSettings.jsp</url-pattern>
<url-pattern>/portlets/PreviewAndSetup.jsp</url-pattern>
<url-pattern>/portlets/PublicQueue/edit.jsp</url-pattern>
<url-pattern>/portlets/PublicQueue/portlet.jsp</url-pattern>
<url-pattern>/portlets/QuickSearch/edit.jsp</url-pattern>
<url-pattern>/portlets/QuickSearch/portlet.jsp</url-pattern>
<url-pattern>/portlets/Workflows/edit.jsp</url-pattern>
<url-pattern>/portlets/Workflows/portlet.jsp</url-pattern>
<url-pattern>/properties/*</url-pattern>
<url-pattern>/redirect/*</url-pattern>
<url-pattern>/regions/*</url-pattern>
<url-pattern>/Search.jsp</url-pattern>
<url-pattern>/select/*</url-pattern>
<url-pattern>/SelectReturn.jsp</url-pattern>
<url-pattern>/Tasks.jsp</url-pattern>
<url-pattern>/UI-INF/*</url-pattern>
<url-pattern>/utils/*</url-pattern>
<url-pattern>/WcmAdmin.jsp</url-pattern>
<url-pattern>/WcmAuthor.jsp</url-pattern>
<url-pattern>/WcmBootstrap.jsp</url-pattern>
<url-pattern>/WcmCloseWindow.jsp</url-pattern>
<url-pattern>/WcmDefault.jsp</url-pattern>
<url-pattern>/WcmError.jsp</url-pattern>
<url-pattern>/WcmJavaViewer.jsp</url-pattern>
<url-pattern>/WcmObjectBookmark.jsp</url-pattern>
<url-pattern>/WcmPortletHelp.jsp</url-pattern>
<url-pattern>/WcmPortletSearch.jsp</url-pattern>
<url-pattern>/WcmQueueBookmark.jsp</url-pattern>
<url-pattern>/WcmSignIn.jsp</url-pattern>
<url-pattern>/WcmSitePreferences.jsp</url-pattern>
<url-pattern>/WcmUserPreferences.jsp</url-pattern>
<url-pattern>/WcmWorkflowsBookmark.jsp</url-pattern>
<url-pattern>/wizards/*</url-pattern>
<url-pattern>/Author/*</url-pattern>
<url-pattern>/axis/*.jws</url-pattern>
<url-pattern>/Browse/*</url-pattern>
<url-pattern>/ceTunnel</url-pattern>
<url-pattern>/CheckoutList/*</url-pattern>
<url-pattern>/downloadMultiTransferElement/*</url-pattern>
<url-pattern>/ExternalUrl/*</url-pattern>
<url-pattern>/findRecordTarget</url-pattern>
```

```
<url-pattern>/formCallback/*</url-pattern>
<url-pattern>/getAnnotSecurity/*</url-pattern>
<url-pattern>/getCEAnnotations/*</url-pattern>
<url-pattern>/getContent/*</url-pattern>
<url-pattern>/getForm/*</url-pattern>
<url-pattern>/getISAnnotations/*</url-pattern>
<url-pattern>/getISAnnotSecurity/*</url-pattern>
<url-pattern>/getISContent/*</url-pattern>
<url-pattern>/getMultiContent/*</url-pattern>
<url-pattern>/getPreview</url-pattern>
<url-pattern>/getProcessor/*</url-pattern>
<url-pattern>/getRealms/*</url-pattern>
<url-pattern>/getUsersGroups/*</url-pattern>
<url-pattern>/Inbox/*</url-pattern>
<url-pattern>/integrationCommandProxy</url-pattern>
<url-pattern>/integrationResponse</url-pattern>
<url-pattern>/integrationResponseProxy</url-pattern>
<url-pattern>/integrationWebBasedCommand</url-pattern>
<url-pattern>/keepAlive</url-pattern>
<url-pattern>/launch/*</url-pattern>
<url-pattern>/PublicQueue/*</url-pattern>
<url-pattern>/putContent/*</url-pattern>
<url-pattern>/QuickSearch/*</url-pattern>
<url-pattern>/signingServlet/*</url-pattern>
<url-pattern>/transport/*</url-pattern>
<url-pattern>/upload/*</url-pattern>
<url-pattern>/vwsimsoapservlet</url-pattern>
<url-pattern>/vwsoaprouter</url-pattern>
<url-pattern>/Workflows/*</url-pattern>
-->
```

b. Find the section shown in Example 5-6 and replace it with the section shown in Example 5-7.

Example 5-6 Original section in web.xml

```
<role-name>*</role-name>
```

Example 5-7 Modified section in web.xml

```
<role-name>All Authenticated</role-name>
```

c. Find the section shown in Example 5-8 and comment it out.

Example 5-8 login-config section in web.xml

```
<login-config>
   <auth-method>FORM</auth-method>
   <realm-name>AE Workplace</realm-name>
   <form-login-config>
   <form-login-page>/ContainerLogin.jsp</form-login-page>
   <form-error-page>/ContainerError.jsp</form-error-page>
   </form-login-config>
</login-config>
```

d. Insert the section shown in Example 5-9 under the commented out section from the previous step.

Example 5-9 Security role section in web.xml

```
<security-role>
   <description>All Authenticated</description>
   <role-name>All Authenticated</role-name>
</security-role>
```

- `WcmApiConfig.properties`

  This file is located in the `C:\Program Files\FileNet\AE\Workplace\WEB-INF` directory on our Windows Server. Make sure that the value for `jaasConfigurationName` is `FileNetP8Engine`, as shown in Example 5-10.

  Example 5-10 WcmApiConfig.properties changes

  ```
  jaasConfigurationName=FileNetP8Engine
  ```

- `application.xml`

  This file is located in the `C:\Program Files\FileNet\AE\deploy\META-INF` directory. As shown in Example 5-11, characters in bold should be added or modified from the original file.

  Example 5-11 application.xml changes

  ```
  <?xml version="1.0" encoding="UTF-8"?>

  <!DOCTYPE application PUBLIC '-//Sun Microsystems, Inc.//DTD J2EE
  Application 1.3//EN' 'http://java.sun.com/dtd/application_1_3.dtd'>

  <application id="workplace">
  ```

```
    <!-- Note: The display-name has to be kept short for WebSphere;
otherwise the path to access class files in WSI will easily exceed
the
               max path size (260). -->
    <display-name>Workplace</display-name>
    <description>FileNet Application Engine</description>

    <module id="wp">
        <web>
            <web-uri>app_engine.war</web-uri>
            <context-root>Workplace</context-root>
        </web>
    </module>
</application>
```

- `ibm-application-bnd.xml`

  You have to create this file from scratch. The file should be located in the `C:\Program Files\FileNet\AE\deploy\META-INF` directory. The contents of this file should be as shown in Example 5-12.

Example 5-12 ibm-application-bnd.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<com.ibm.ejs.models.base.bindings.applicationbnd:ApplicationBinding
xmi:version="2.0"xmlns:xmi="http://www.omg.org/XMI"
xmlns:com.ibm.ejs.models.base.bindings.applicationbnd="applicationbn
d.xmi" xmi:id="Application_ID_Bnd">
<authorizationTable xmi:id="AuthorizationTable_1">
<authorizations xmi:id="RoleAssignment_1155940459609">
<specialSubjects
xmi:type="com.ibm.ejs.models.base.bindings.applicationbnd:AllAuthent
icatedUsers" xmi:id="AllAuthenticatedUsers_1164745308375"
name="AllAuthenticatedUsers"/>
<role href="META-INF/application.xml#SecurityRole_1164745308375"/>
</authorizations>
</authorizationTable>
<application href="META-INF/application.xml#workplace"/>
<runAsMap xmi:id="RunAsMap_1"/>
</com.ibm.ejs.models.base.bindings.applicationbnd:ApplicationBinding
>
```

- `ibm-application-ext.xml`

  You should create this file from scratch. The file should be located in the `C:\Program Files\FileNet\AE\deploy\META-INF` directory. The contents of this file should be as shown in Example 5-13.

  Example 5-13 ibm-application-ext.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<com.ibm.ejs.models.base.bindings.applicationbnd:ApplicationBinding
xmi:version="2.0"xmlns:xmi="http://www.omg.org/XMI"
xmlns:com.ibm.ejs.models.base.bindings.applicationbnd="applicationbn
d.xmi" xmi:id="Application_ID_Bnd">
<authorizationTable xmi:id="AuthorizationTable_1">
<authorizations xmi:id="RoleAssignment_1155940459609">
<specialSubjects
xmi:type="com.ibm.ejs.models.base.bindings.applicationbnd:AllAuthent
icatedUsers" xmi:id="AllAuthenticatedUsers_1164745308375"
name="AllAuthenticatedUsers"/>
<role href="META-INF/application.xml#SecurityRole_1164745308375"/>
</authorizations>
</authorizationTable>
<application href="META-INF/application.xml#workplace"/>
<runAsMap xmi:id="RunAsMap_1"/>
</com.ibm.ejs.models.base.bindings.applicationbnd:ApplicationBinding
>
```

Create WAR file and EAR file

Next, we need to create the WAR and EAR files using the deployment descriptor files that we customized in the previous steps. This section should be performed on the FileNet P8 Application Engine machine.

Do these steps:

1. Open a Windows command line.
2. Change the directory to `C:\Program Files\FileNet\AE\deploy`.
3. Execute the batch file `create_app_engine_war.bat`.
4. Execute the batch file `create_app_engine_ear.bat`.

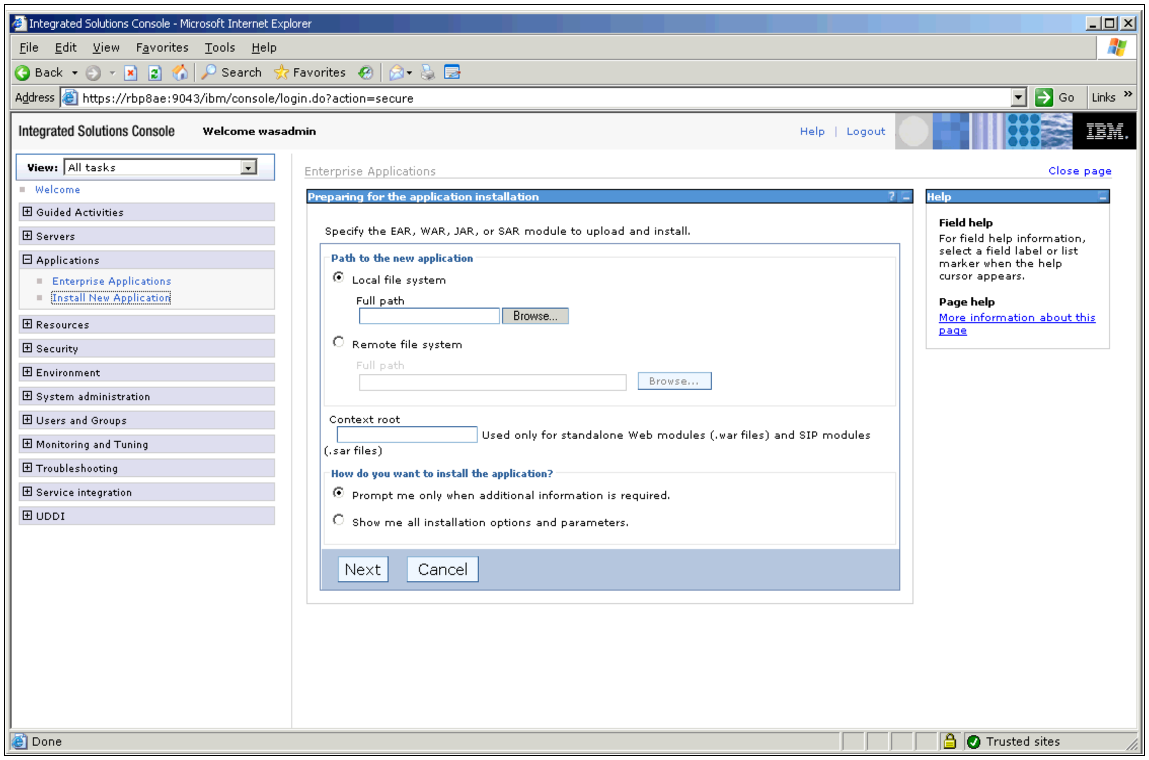
Now you are ready to deploy your application package on WebSphere Application Server.

Deploy the application to WebSphere Application Server

In this step, you deploy the FileNet P8 Application Engine application by using the WebSphere console.

Do these steps:

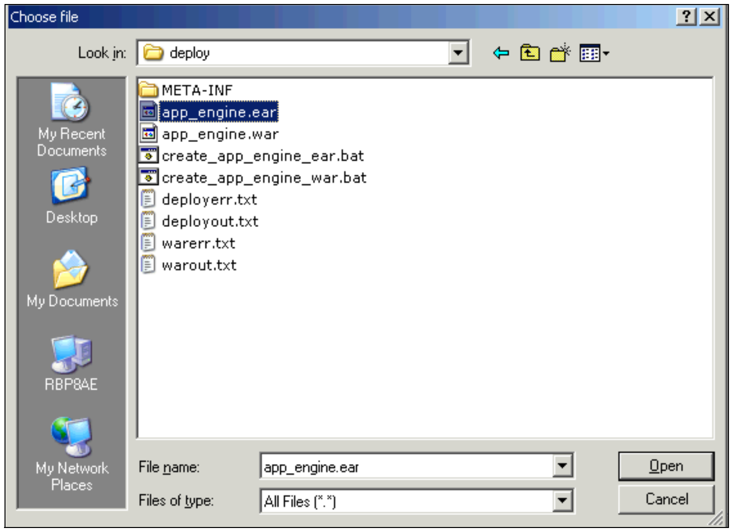
1. Open the IBM ISC in your browser. Click the **Application** tab, then click **Install New Application**. You should see the window shown in Figure 5-31.



Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://rbp8ae:9043/ibm/console/login.do?action=secure
Go Links
Integrated Solutions Console
Welcome wasadmin
Help | Logout
View: All tasks
Welcome
Guided Activities
Servers
Applications
Enterprise Applications
Install New Application
Resources
Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Enterprise Applications
Close page
Preparing for the application installation
Help
Field help
For field help information, select a field label or list marker when the help cursor appears.
Page help
More information about this page
Specify the EAR, WAR, JAR, or SAR module to upload and install.
Path to the new application
Local file system
Full path
Browse...
Remote file system
Full path
Browse...
Context root
Used only for standalone Web modules (.war files) and SIP modules (.sar files)
How do you want to install the application?
Prompt me only when additional information is required.
Show me all installation options and parameters.
Next
Cancel
Done
Trusted sites

Figure 5-31 Install new application window

2. Click **Browse** in the Local file system section. You should see the window shown in Figure 5-32.

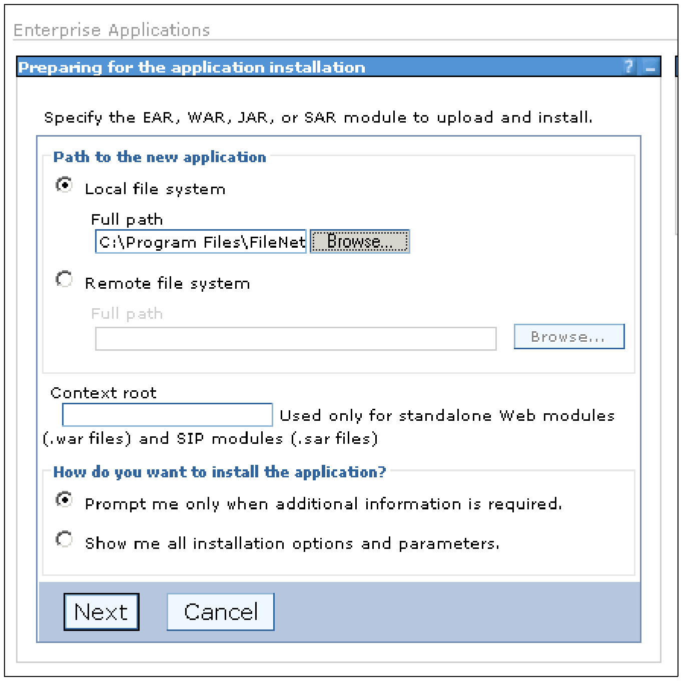


META-INF
app_engine.ear
app_engine.war
create_app_engine_ear.bat
create_app_engine_war.bat
deployerr.txt
deployout.txt
warerr.txt
warout.txt

File name: app_engine.ear
Files of type: All Files (*.*)

Figure 5-32 File selection box for the application

3. Select **C:\Program Files\FileNet\AE\deploy\app_engine.ear** and click **Open**. You should see the window shown in Figure 5-33.


Enterprise Applications

Preparing for the application installation

Specify the EAR, WAR, JAR, or SAR module to upload and install.

Path to the new application

(●) Local file system

Full path: C:\Program Files\FileNet [Browse...]

( ) Remote file system

Full path: [Browse...]

Context root: Used only for standalone Web modules (.war files) and SIP modules (.sar files)

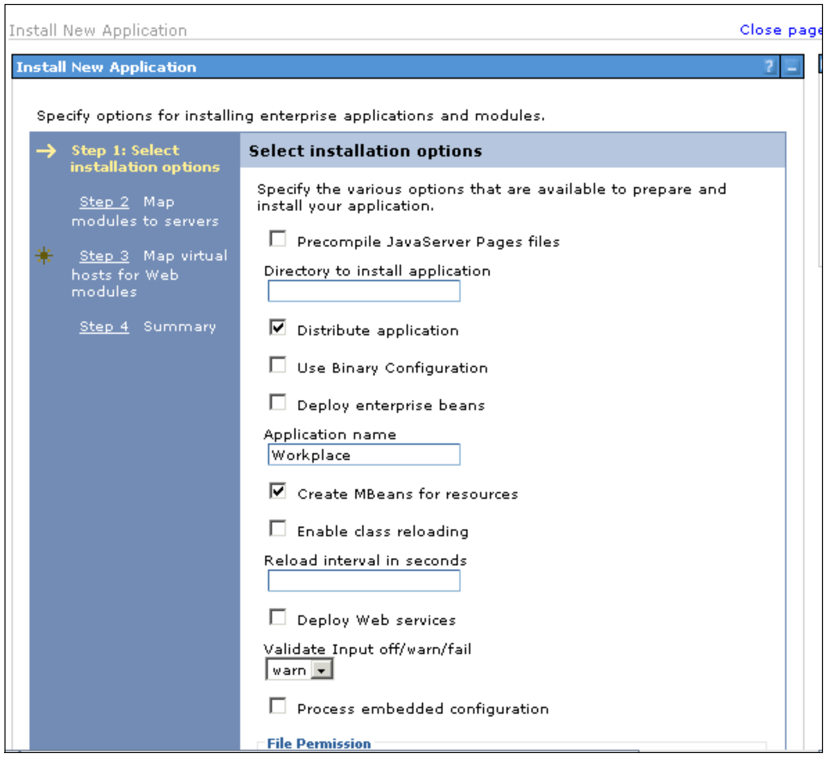
How do you want to install the application?

(●) Prompt me only when additional information is required.

( ) Show me all installation options and parameters.

Figure 5-33 Install new application window

4. Click **Next**. You should see the window shown in Figure 5-34.


Install New Application

Close page

Install New Application

Specify options for installing enterprise applications and modules.

→ Step 1: Select installation options
Step 2 Map modules to servers
Step 3 Map virtual hosts for Web modules
Step 4 Summary

Select installation options

Specify the various options that are available to prepare and install your application.

☐ Precompile JavaServer Pages files

Directory to install application

☑ Distribute application

☐ Use Binary Configuration

☐ Deploy enterprise beans

Application name: Workplace

☑ Create MBeans for resources

☐ Enable class reloading

Reload interval in seconds

☐ Deploy Web services

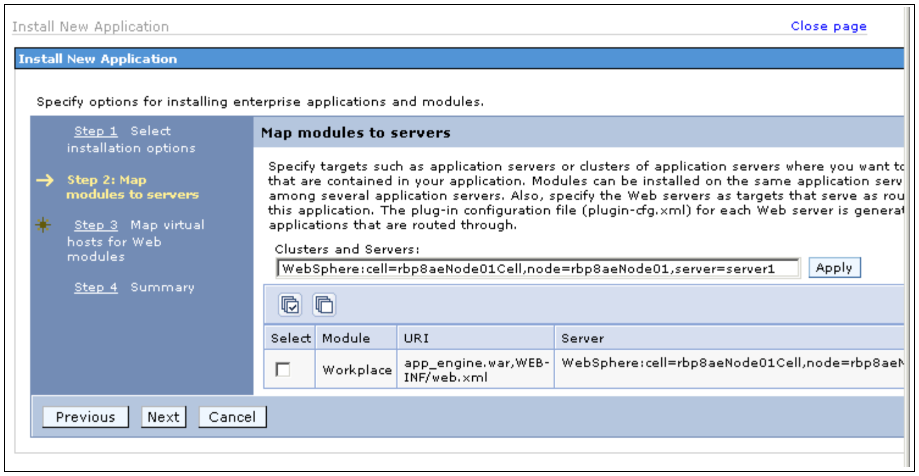
Validate Input off/warn/fail: warn

☐ Process embedded configuration

File Permission

Figure 5-34 Install New Application window

5. No changes are needed in this window. Click **Next**. You should see the window shown in Figure 5-35 on page 173.



Install New Application | Close page

Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

→ **Step 2: Map modules to servers**

Step 3 Map virtual hosts for Web modules

Step 4 Summary

Map modules to servers

Specify targets such as application servers or clusters of application servers where you want tc that are contained in your application. Modules can be installed on the same application serv among several application servers. Also, specify the Web servers as targets that serve as rou this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generat applications that are routed through.

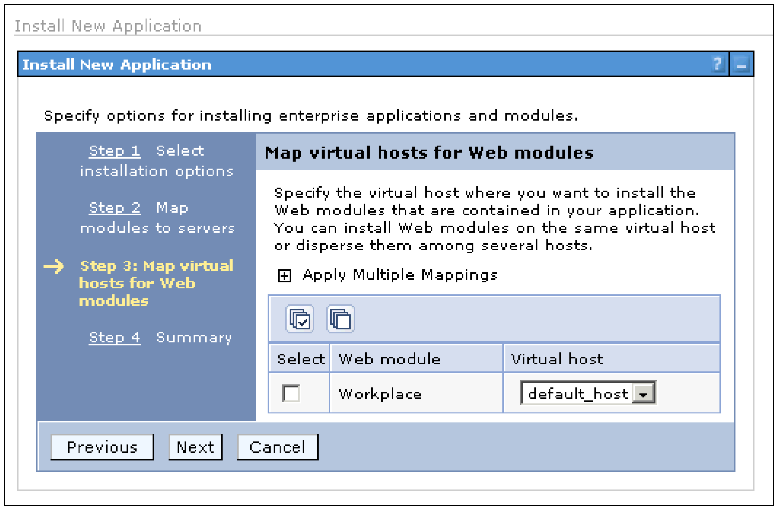
Clusters and Servers:
WebSphere:cell=rbp8aeNode01Cell,node=rbp8aeNode01,server=server1 | Apply

Select	Module	URI	Server
☐	Workplace	app_engine.war,WEB-INF/web.xml	WebSphere:cell=rbp8aeNode01Cell,node=rbp8aeN

Previous | Next | Cancel

Figure 5-35 Install New Application window

6. No changes are needed in this window. Click **Next**. You should see the window shown in Figure 5-36.



Install New Application

Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2 Map modules to servers

→ **Step 3: Map virtual hosts for Web modules**

Step 4 Summary

Map virtual hosts for Web modules

Specify the virtual host where you want to install the Web modules that are contained in your application. You can install Web modules on the same virtual host or disperse them among several hosts.

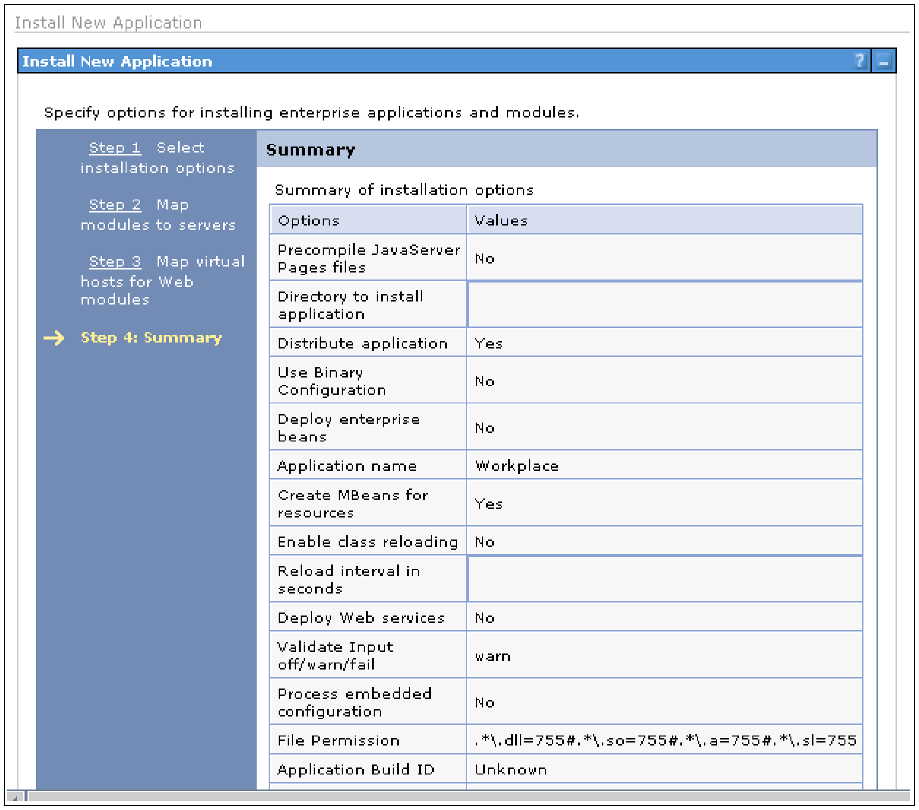
⊞ Apply Multiple Mappings

Select	Web module	Virtual host
☐	Workplace	default_host

Previous | Next | Cancel

Figure 5-36 Install New Application window

7. Again, no changes are needed in this window. Click **Next**. You should see the window shown in Figure 5-37.


Install New Application

Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options
Step 2 Map modules to servers
Step 3 Map virtual hosts for Web modules
Step 4: Summary

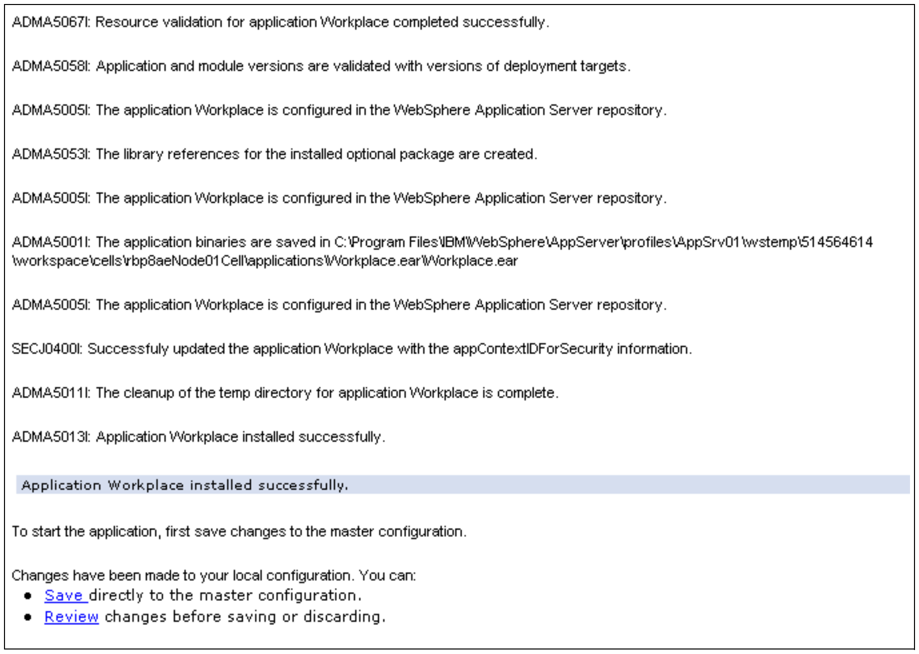
Summary

Summary of installation options

Options	Values
Precompile JavaServer Pages files	No
Directory to install application	
Distribute application	Yes
Use Binary Configuration	No
Deploy enterprise beans	No
Application name	Workplace
Create MBeans for resources	Yes
Enable class reloading	No
Reload interval in seconds	
Deploy Web services	No
Validate Input off/warn/fail	warn
Process embedded configuration	No
File Permission	.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
Application Build ID	Unknown

Figure 5-37 Install New Application window

8. No changes are needed in this window. Click **Finish**. The install process is displayed, as shown in Figure 5-38 on page 175.

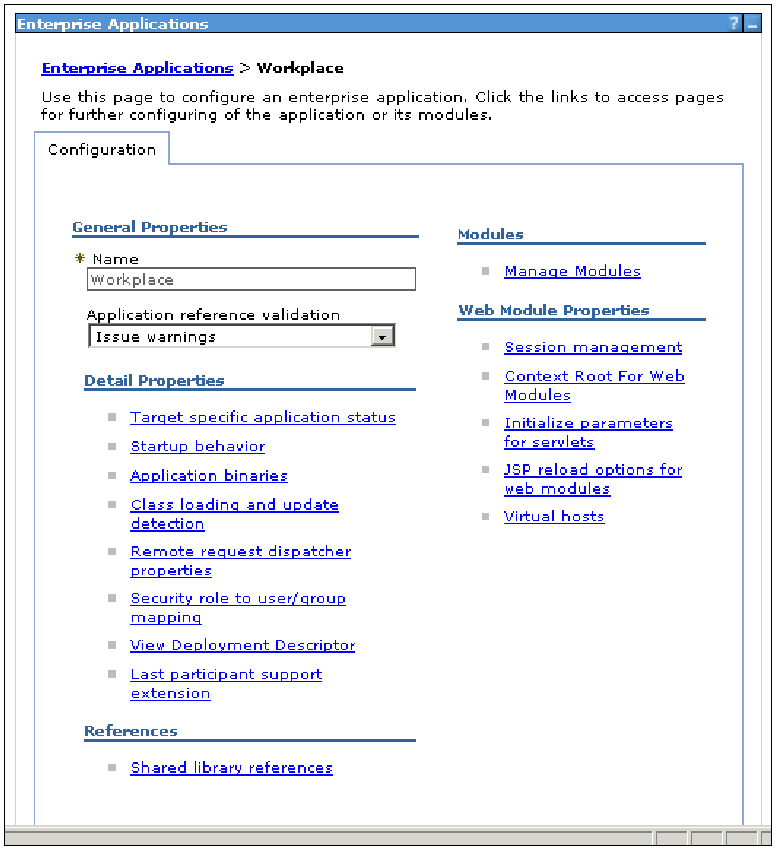


```
ADMA5067I: Resource validation for application Workplace completed successfully.

ADMA5058I: Application and module versions are validated with versions of deployment targets.

ADMA5005I: The application Workplace is configured in the WebSphere Application Server repository.

ADMA5053I: The library references for the installed optional package are created.

ADMA5005I: The application Workplace is configured in the WebSphere Application Server repository.

ADMA5001I: The application binaries are saved in C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\wstemp\514564614
\workspace\cells\rbp8aeNode01Cell\applications\Workplace.ear\Workplace.ear

ADMA5005I: The application Workplace is configured in the WebSphere Application Server repository.

SECJ0400I: Successfuly updated the application Workplace with the appContextIDForSecurity information.

ADMA5011I: The cleanup of the temp directory for application Workplace is complete.

ADMA5013I: Application Workplace installed successfully.

Application Workplace installed successfully.

To start the application, first save changes to the master configuration.

Changes have been made to your local configuration. You can:
 • Save directly to the master configuration.
 • Review changes before saving or discarding.
```

Figure 5-38 Install progression window

9. After the installation is completed, click **Save**.

10. Go to the **Applications** menu on the left side of the window. Click **Enterprise Applications**, then click **Workplace**. You should see the window shown in Figure 5-39.


Enterprise Applications

Enterprise Applications > Workplace

Use this page to configure an enterprise application. Click the links to access pages for further configuring of the application or its modules.

Configuration

General Properties

* Name

Workplace

Application reference validation

Issue warnings

Detail Properties

- Target specific application status
- Startup behavior
- Application binaries
- Class loading and update detection
- Remote request dispatcher properties
- Security role to user/group mapping
- View Deployment Descriptor
- Last participant support extension

References

- Shared library references

Modules

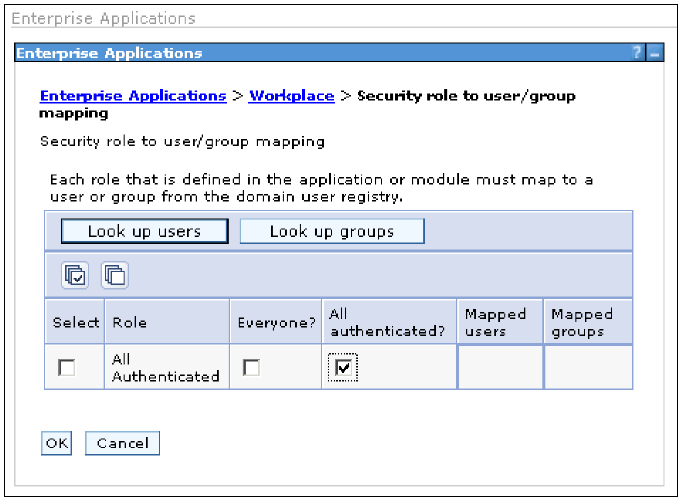
- Manage Modules

Web Module Properties

- Session management
- Context Root For Web Modules
- Initialize parameters for servlets
- JSP reload options for web modules
- Virtual hosts

Figure 5-39 Workplace application properties

11. Click **Security role to user/group mapping**. You should see the window shown in Figure 5-40 on page 177. After the installation of the Workplace application, there is no check mark in this window. You should check **All authenticated?**, as shown in Figure 5-40 on page 177. Click **OK**.


Enterprise Applications

Enterprise Applications

Enterprise Applications > Workplace > Security role to user/group mapping

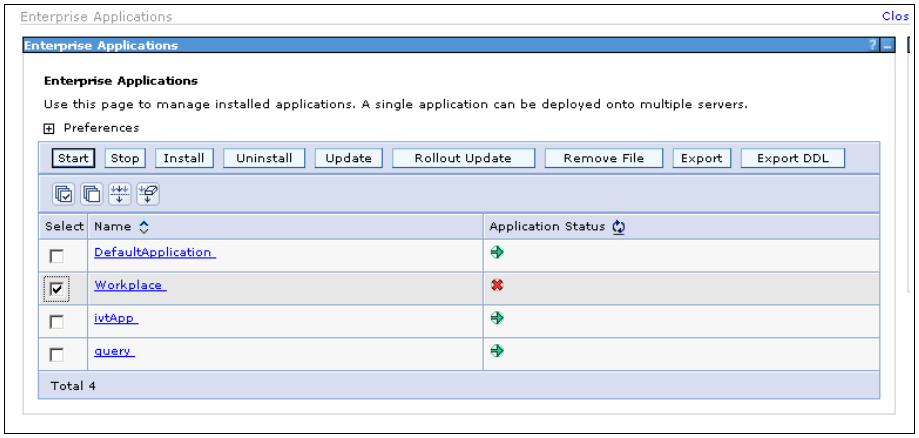
Security role to user/group mapping

Each role that is defined in the application or module must map to a user or group from the domain user registry.

Select	Role	Everyone?	All authenticated?	Mapped users	Mapped groups
☐	All Authenticated	☐	☑		

Figure 5-40   Security role mapping

12. We start the Workplace application, which actually represents the FileNet P8 Application Engine. Go to the **Applications** menu on the left of the window and click **Enterprise Applications**. You should see the window shown in Figure 5-41.


Enterprise Applications

Enterprise Applications

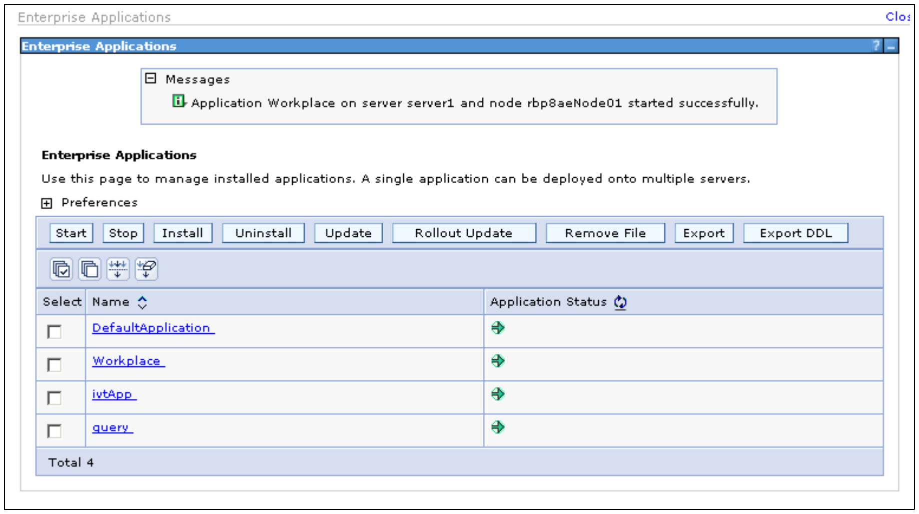
Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple servers.

Select	Name	Application Status
☐	DefaultApplication	➔
☑	Workplace	✖
☐	ivtApp	➔
☐	query	➔
Total 4		

Figure 5-41   Enterprise Applications List

13. Check the box to the left side of the **Workplace** application, as shown in Figure 5-41 on page 177, and then click **Start**. After a while, you should see a successful startup message, as shown in Figure 5-42.


Enterprise Applications
Enterprise Applications

Messages

Application Workplace on server server1 and node rbp8aeNode01 started successfully.

Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple servers.

Preferences
Start Stop Install Uninstall Update Rollout Update Remove File Export Export DDL

Select	Name	Application Status
☐	DefaultApplication	➜
☐	Workplace	➜
☐	ivtApp	➜
☐	query	➜
Total 4		

Figure 5-42 Workplace application started successfully

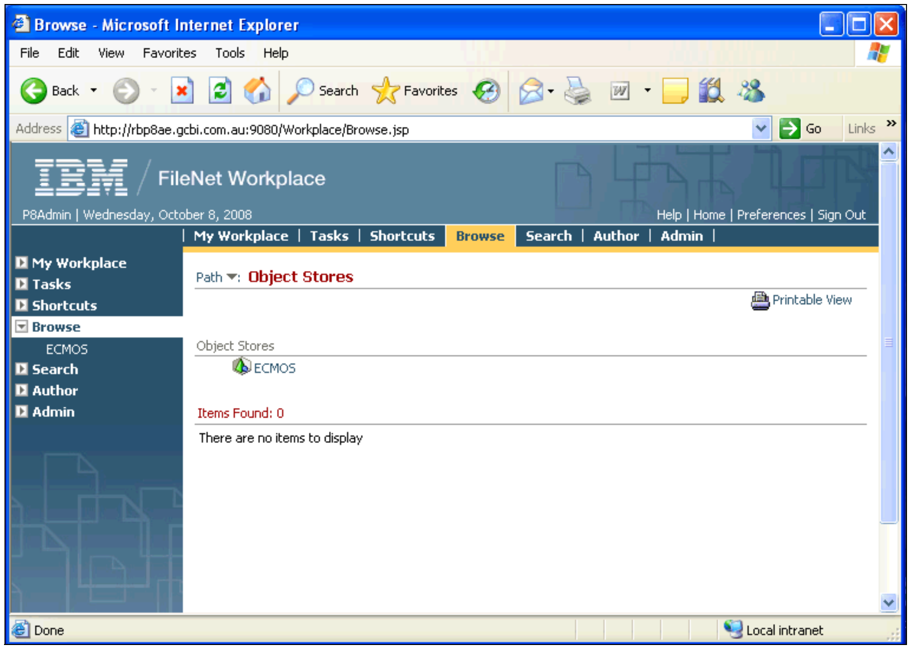
14. At this time, you are ready to use the FileNet P8 Application Engine.

5.2.10 Integration test for P8 AE and SPNEGO

Now you can test your configuration by accessing the FileNet P8 Application Engine, which is called *Workplace*.

Do these steps:

1. Log on to the Active Directory domain on your local Windows workstation. In our example, we use the p8admin user account that is already registered on Active Directory.
2. Access your P8 AE server by using your Web browser. In our example, we access `http://rbp8ae.gcbi.com.au:9060/Workplace`. Because SPNEGO is working underneath the surface, you can access the Workplace application without any additional logon procedures. You should see the window shown in Figure 5-43 on page 179.


Browse - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://rbp8ae.gcbi.com.au:9080/Workplace/Browse.jsp
Go Links
IBM FileNet Workplace
P8Admin | Wednesday, October 8, 2008
Help | Home | Preferences | Sign Out
My Workplace | Tasks | Shortcuts | Browse | Search | Author | Admin
My Workplace
Tasks
Shortcuts
Browse
ECMOS
Search
Author
Admin
Path: Object Stores
Printable View
Object Stores
ECMOS
Items Found: 0
Done
Local intranet

There are no items to display

Figure 5-43 Successful SPNEGO and FileNet P8 Workplace integration

This concludes our discussion of the setup and configuration of SPNEGO-based single sign-on for the IBM FileNet P8 environment.

6

Single sign-on using Tivoli Federated Identity Manager

In this chapter, we describe how Tivoli Federated Identity Manager (TFIM) can be used to enhance the Web service security capabilities of the FileNet P8 platform. By using Federated Identity Manager, we provide additional security token support for accessing the FileNet platform through Web services. We also allow greater control of identity propagation between partners within a SOA environment.

In this chapter, we cover the following topics:

Important: The solution described in this chapter represents a prototype implementation to demonstrate the type of integration that can be performed with IBM Tivoli Federated Identity Manager. If this type of solution is desired for a customer environment, a consulting solution would have to be arranged to develop integration components specific to the customers unique SOA application and environment.

6.1 Overview

In this section, we provide an overview of Tivoli Federated identity Manager and its role in enhancing the FileNet P8 platform to support additional token types in Web service requests. This information relates to the GCBI customer solution outlined in Chapter 3, “Customer overview” on page 55 and more particularly to a partner relationship using the FileNet P8 Content Engine Web services interface. The overview covers these topics:

- “Partners involved” on page 182
- “Installation overview” on page 183
- “Prerequisites and assumptions” on page 184

6.1.1 Partners involved

GCBI would like to set up a partner relationship with the Queensland Insurance Solutions Agency (QISA). QISA will require access to certain GCBI documents to perform approval of mortgage loans as part of GCBI’s loan approval process. In order to do this, QISA will access GCBI documents through GCBI’s FileNet P8 Content Engine Web services interface.

In order for this approach to go ahead, the two companies have decided that the basis of their environment will consist of Web service requests using SOAP over HTTP(S) and SAML 2.0 security tokens. This requires the addition of Federated Identity Manager to allow SAML token support in the FileNet P8 environment. In order for GCBI to not have to manage the identities of the QISA employees, we anticipate that they will use Federated Identity Manager to perform user mapping as well. The mapping rule to be used by GCBI will map users in the format *user@qisa.com.au* to the single user *qisapartner*.

The two companies have set up the following guidelines to ensure each knows their responsibilities in regards to the partnership:

- QISA provides its own Content Engine WS client that its employees will use to access the GCBI FileNet P8 platform.
- QISA will be responsible for authentication, authorization, and audit within its client application.
- QISA users will be represented in the GCBI FileNet registry as the single user *qisapartner,* which is really a representation of the group to which they belong.
- SAML 2.0 tokens will be used between the QISA client and the GCBI FileNet Content Engine Web service.

- The SAML 2.0 tokens sent by the QISA client will contain the user in the format user@qisa.com.au.
- GCBI will be responsible for mapping the QISA user from the SAML 2.0 token into the local GCBI user *qisapartner*.

Now that we have outlined the partner details and explained each partner's responsibilities, we need to explain the GCBI Federated Identity Manager environment.

6.1.2 Installation overview

In order to implement this approach, GCBI needs to set up the following Federated Identity Manager environment in addition to their existing FileNet P8 infrastructure. The new Federated Identity Manager environment consists of Tivoli Federated Identity Manager V6.2 deployed on WebSphere Application Server V6.1.14. The Federated Identity Manager components required for this approach are the Federated Identity Manager runtime and Federated Identity Manager management service. The Web services security management component is not required.

Figure 6-1 shows the complete system overview.

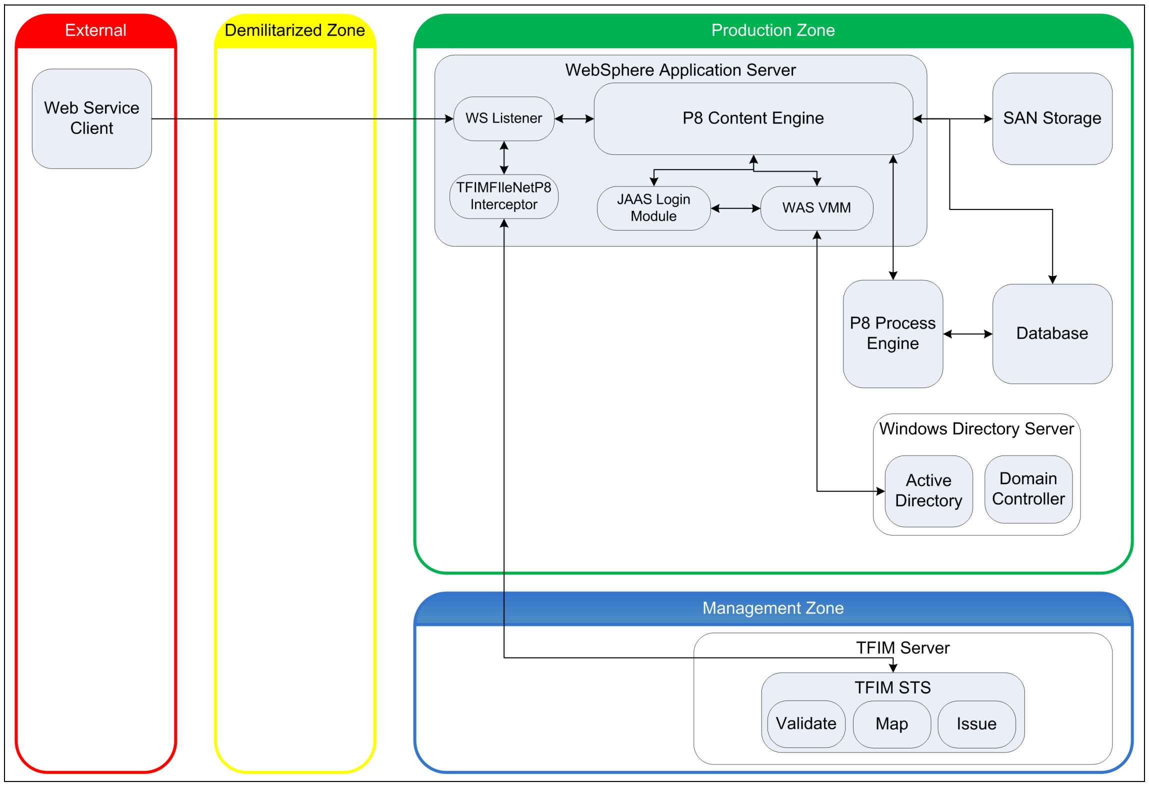


Figure 6-1 GCBI FileNet P8 and Federated Identity Manager infrastructure

6.1.3 Prerequisites and assumptions

The following prerequisites and assumptions are required in order to successfully implement the scenario:

- The FileNet P8 Content Engine Web service (CEWS) interface is deployed and working as expected.
- The Federated Identity Manager environment is installed and configured as per the *IBM Tivoli Federated Identity Manager Version 6.2 Installation and Configuration Guide*, SC23-6190, including Federated Identity Manager domain configuration.
- A Web service client has been developed and works as expected with the CEWS using the standard Username Token as an authentication mechanism.

- The working CEWS client can be modified or configured to utilize an external system (such as Federated Identity Manager WSSM) to send SAML 2.0 tokens in the security header of the request.
- This implementation only deals with the configuration of the Web service consumer who receives a valid SAML 2.0 token.
- Each party has entered into a legally binding agreement to ensure that each knows and maintains their responsibilities for this partnership.

Now that we have outlined the prerequisites and assumptions within this environment, we explain the Federated Identity Manager FileNet P8 solution.

Note: Note that the CEWS client described above may be an application written against either the Content Engine Java API or .NET API. Using one of these higher level APIs (which are able to sit on top of the CEWS layer) is generally a preferable approach, rather than writing directly against the Content Engine Web service layer.

6.2 IBM FileNet P8 and Federated Identity Manager solution

In this section, we describe the integration flow between FileNet P8 and Federated Identity Manager in order to allow multiple security token types for the Content Engine Web services interface. This approach also addresses the SOA identity propagation needs of GCBI. Each of the components in this solution are outlined in 6.3, “Solution components” on page 187.

Figure 6-2 shows the solution.

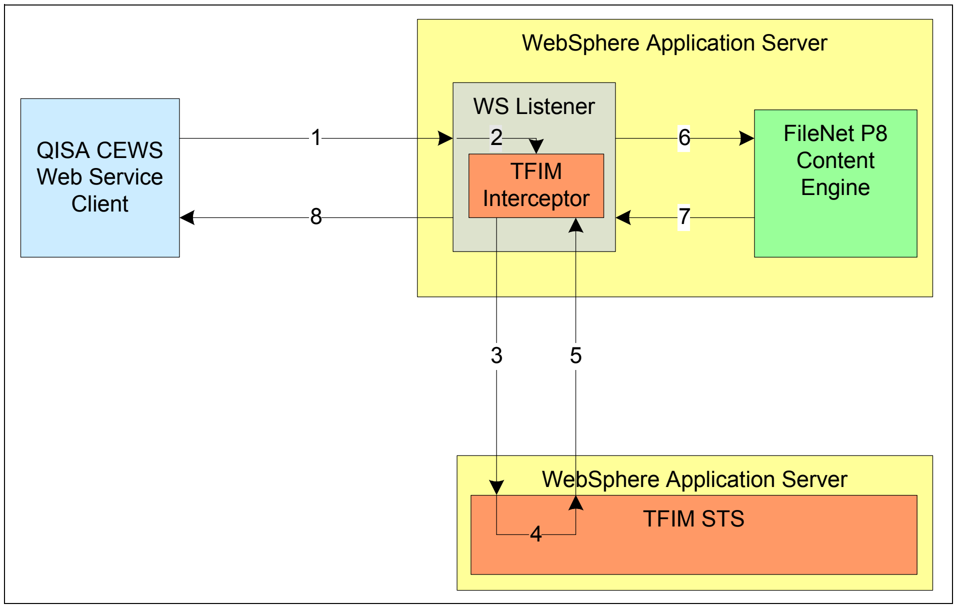

WebSphere Application Server
WS Listener
QISA CEWS Web Service Client
1
2
TFIM Interceptor
6
FileNet P8 Content Engine
8
7
3
5
WebSphere Application Server
TFIM STS
4

Figure 6-2 Federated Identity Manager FileNet P8 solution

Where:

1. The QISA Web service client sends a request to the GCBI FileNet P8 Content Engine Web service containing a SAML 2.0 token containing the user name in the format user@qisa.com.au.
2. The Federated Identity Manager FileNet Interceptor intercepts the request and removes the SAML 2.0 security token.
3. The Federated Identity Manager FileNet Interceptor sends an Request Security Token (RST) to the Federated Identity Manager STS containing the appropriate AppliesTo and Issuer parameters as well as the original SAML 2.0 token.
4. The Federated Identity Manager STS performs a validation of the SAML 2.0 token from the RST, maps the external user to the internal user qisapartner, and issues a Username Token.
5. The Federated Identity Manager STS returns an Request Security Token Response (RSTR) to the Federated Identity Manager FileNet Interceptor, which includes the Federated Identity Manager STS issued Username Token.
6. The Federated Identity Manager FileNet Interceptor replaces the SAML 2.0 token contained within the security header of the original request with the

Username Token supplied by the Federated Identity Manager STS from the RSTR. The request is sent to the FileNet Content Engine Web service.

7. The Content Engine processes the Web service request, including authentication of the user using the Username Token in the security header.
8. The result of the request is returned to the Web service client.

For more information about the general Federated Identity Manager architecture, see 2.4, “Tivoli Federated Identity Manager architecture overview” on page 45.

6.3 Solution components

In this section, we outline the components required to enable this solution. Some of these modules are custom developed and are included for download with this book. The components outlined include:

- “Sample Web service client” on page 187
- “Federated Identity Manager FileNet Interceptor” on page 188
- “FileNet Content Engine Web Service” on page 189
- “Trust chain” on page 189
- “E-mail to user name and password mapping module” on page 190

Note: The code that has been developed in this book project is provided to you on an *as is* basis; no support whatsoever can be rendered for the components.

6.3.1 Sample Web service client

We have developed a Web service client using iBM Rational Application Developer V7.0 that is designed to run on WebSphere Application Server V6.1. The client is configured to send a SAML 2.0 token in the security header of the Web service request to the FileNet Content Engine Web service. This SAML 2.0 token is generated using the Federated Identity Manager WSSM component.

The client consists of two servlets, one JSP and the underlying WebSphere Application Server generated Web service client code. The client can create folders and documents within the FileNet P8 platform using the CEWS. While the client is an important part of the development and testing of this Federated Identity Manager FileNet scenario, the client code included with this book should only be used as a reference for testing.

6.3.2 Federated Identity Manager FileNet Interceptor

The FileNet Content Engine Web service runs within the *Content Engine Web Service listener*. The CEWS listener is a Web services framework that is application server agnostic. In our case, the CEWS listener is running on WebSphere Application Server and is part of our default FileNet Content Engine deployment.

The CEWS listener contains its own Web services stack, which it utilizes to process Web service requests. This means that we cannot use the existing Federated Identity Manager Web Services Security Management (WSSM) component to intercept our request and perform token exchange with the Federated Identity Manager Security Token Service (STS). Instead, we need to use an interceptor, which is invoked when a Web service request is sent to the CEWS listener.

The interceptor needs to be able to perform the following tasks:

1. Intercept the incoming CEWS request.
2. Remove the security header from the original request.
3. Make a WS-Trust request to Federated Identity Manager with the security header from the original request.
4. Have Federated Identity Manager perform validation on the received security token, perform a mapping, and return a Username Token.
5. Insert the Username Token into the CEWS request.
6. Send the request on to the Content Engine.

A suitable interceptor has been written to perform these functions and is included with this book. The interceptor is called the TFIMFileNetP8Interceptor and is contained in the `TFIMFileNetP8Interceptor.jar` file.

Note: Note that the Federated Identity Manager FileNet Interceptor described in this section is based on internal architecture details for the P8 V4.0.1 Content Engine. While this version of the interceptor does work with the V4.0.1 and V4.5.0 Content Engines, this internal area of the P8 Content Engine is likely to be rewritten in a future release, requiring a completely different implementation of this interceptor module. There is no support available from the IBM FileNet P8 team for this integration module prototype.

Again, if this type of solution is desired for a customer environment, a consulting solution would have to be arranged to develop integration components specific to the customers unique SOA application and environment.

6.3.3 FileNet Content Engine Web Service

The FileNet Content Engine Web Service default authentication module requires that a Username Token, including the password, be contained within the Web service request. This is because the Content Engine Web Service engine authenticates a user and performs a login to FileNet P8 using the user name and password found in the token. In order to satisfy this requirement, we must ensure that the Username Token sent from Federated Identity Manager contains a valid user name and password. This means that we must use a *mapping module* that populates the user name and password portion of the Security Token Service Universal User (STSUU) from some data source before the token is issued.

As an alternative to the default authentication module, the CEWS engine also provides an interface to replace the user name and password Web services authentication mechanism. This is known as the *Web services extensible authentication framework* (WS-EAF). The WS-EAF allows us to create a custom authentication module that could use a Username Token without a password as an identity assertion. This can mean we would not be required to obtain the password and place it in the STSUU before the Username Token is issued. Due to time restraints of our short project, we could not endeavour to use this approach.

6.3.4 Trust chain

In order to perform the token exchange using the Federated Identity Manager STS, a Federated Identity Manager *trust chain* must be created. A Federated Identity Manager trust chain contains a series of trust modules that are invoked when the trust server receives an incoming request. The trust chain that is invoked when the request is received depends on the *AppliesTo* and *Issuer* parameters of the trust chain.

This chain can contain modules that perform the following operations:

Validate	Performs validation of the incoming security token.
Map	Performs mapping of identities from one administrative domain to another.
Other	Performs operations such as authorization.
Issue	Performs the issue of outgoing security tokens.

The chain that we use in this example validates an incoming SAML 2.0 token, maps the user from an e-mail format into a partner identifier, and issues a Username Token.

6.3.5 E-mail to user name and password mapping module

In order to map the external user presented in the incoming security token (SAML 2.0) into an internal user for issuing the outgoing security token (Username Token) included in the FileNet CEWS request, we must create a Federated Identity Manager *mapping rule*.

The mapping rule performs the following:

- Map the incoming external user to an internal user within FileNet, for example, user@qisa.com.au ∅ qisapartner.
- Add the password of the qisapartner user to the STSUU ready to be included in the Username Token when it is issued by the Federated Identity Manager STS.

This task can be performed using XSLT. Example 6-1 shows the mapping rule used for this approach.

Example 6-1 Federated Identity Manager FileNet XSLT identity mapping rule

```
<!-- This stylesheet is for use in conjunction with the TFIM FileNet
integration -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xalan="http://xml.apache.org/xalan"
xmlns:mapping-ext="com.tivoli.am.fim.trustserver.sts.utilities.IDMappin
gExtUtils" xmlns:stsuuser="urn:ibm:names:ITFIM:1.0:stsuuser"
version="1.0" xmlns:fn="http://www.w3.org/2005/02/xpath-functions">
 <xsl:strip-space elements="*" />
 <xsl:output method="xml" version="1.0" encoding="utf-8" indent="yes"
/>

 <!-- Identity transform - copy all content -->
 <xsl:template match="@* | node()">
  <xsl:copy>
   <xsl:apply-templates select="@* | node()" />
  </xsl:copy>
 </xsl:template>

 <xsl:template match="//stsuuser:STSUniversalUser ">
  <xsl:copy>
   <xsl:apply-templates />
  </xsl:copy>
 </xsl:template>

 <!-- Transform user from form xxx@qisa.com.au to qisapartner -->
 <xsl:template match="//stsuuser:Principal">
```

```
  <stsuuser:Principal>
   <xsl:choose>
    <xsl:when
test="contains(//stsuuser:Principal/stsuuser:Attribute[@name='name']/st
suuser:Value/text(), 'qisa.com.au')">
     <stsuuser:Attribute name="name">
      <stsuuser:Value>qisapartner</stsuuser:Value>
     </stsuuser:Attribute>
     <stsuuser:Attribute name="Password"
type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText">
      <stsuuser:Value>password</stsuuser:Value>
     </stsuuser:Attribute>
    </xsl:when>
    <xsl:when
test="contains(//stsuuser:Principal/stsuuser:Attribute[@name='name']/st
suuser:Value/text(), 'P8Admin')">
     <stsuuser:Attribute name="name">
      <stsuuser:Value>P8Admin</stsuuser:Value>
     </stsuuser:Attribute>
     <stsuuser:Attribute name="Password"
type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText">
      <stsuuser:Value>filenet</stsuuser:Value>
     </stsuuser:Attribute>
    </xsl:when>
    <xsl:otherwise>
     <stsuuser:Attribute name="name">
      <stsuuser:Value>
       <xsl:value-of
select="//stsuuser:Principal/stsuuser:Attribute[@name='name']/stsuuser:
Value/text()" />
      </stsuuser:Value>
     </stsuuser:Attribute>
    </xsl:otherwise>
   </xsl:choose>
  </stsuuser:Principal>
 </xsl:template>

 <xsl:template match="//stsuuser:AttributeList">
  <xsl:copy>
   <xsl:apply-templates />
  </xsl:copy>
 </xsl:template>
```

```
</xsl:stylesheet>
```

Note: This mapping module assumes that the user *qisapartner* is a valid user in the FileNet P8 LDAP directory.

6.4 Configuration

In this section, we outline the configuration required on Federated Identity Manager and the FileNet P8 platform to allow Web services token exchange using Federated Identity Manager. This involves installing and configuring the TFIMFileNetP8Interceptor and configuring the Federated Identity Manager trust chain.

6.4.1 Configuring the TFIMFileNetP8Interceptor

In order to successfully intercept the Web service request and change the WS-Security header using Federated Identity Manager, we need to install and configure the TFIMFileNetP8Interceptor. To do this task, perform the following steps:

1. Copy the `ws-trust.wsdl` and the `TFIMFileNetP8Interceptor.jar` files into separate directories on the Content Engine. Remember the locations so that they can be used later.
2. At a command prompt, browse to the folder containing the `TFIMFileNetP8Interceptor.jar` file and unjar it so that the configuration file can be updated for your environment:

   ```
   <JAVA_BIN_PATH>\jar.exe xvf TFIMFileNetP8Interceptor.jar
   ```

   Note: `<JAVA_BIN_PATH>` is the path to the JRE on your application server instance. On Windows, using WebSphere Application Server, this is `C:\Program Files\IBM\WebSphere\AppServer\java\bin`.

   You should see output similar to the one shown in Example 6-2.

Example 6-2 Output from TFIMFileNetP8Interceptor unjar operation

```
C:\TFIMFileNetP8Interceptor>"c:\Program
Files\IBM\WebSphere\AppServer\java\bin\jar.exe" xvf TFIMFileNetP8Interceptor.jar
 inflated: META-INF/MANIFEST.MF inflated:
com/tivoli/soa/tfim/interceptors/TFIMFileNetP8InterceptorInputMessage.class
 inflated: com/tivoli/soa/tfim/interceptors/TFIMFileNetP8Interceptor.class
```

```
 inflated:
com/tivoli/soa/tfim/interceptors/TFIMFileNetP8InterceptorOutputMessage.class
 inflated: com/tivoli/soa/tfim/interceptors/SecurityTokenService.class
 inflated: .classpath
 inflated: .project
 inflated: TFIMFileNetP8Interceptor.properties
```

3. Modify the `TFIMFileNetP8Interceptor.properties` file extracted from the jar based on the configuration options shown in Table 6-1.

Table 6-1 TFIMFileNetP8Interceptor configuration items

Configuration item	**Description**
trustServiceUrl	The URL of the TFIM STS endpoint, for example, http://tfim:9080/TrustServer/SecurityTokenService.
issuerAddress	The name of the issuer of the request to the TFIM STS. The default is urn:tfim:filenet:interceptor. This must match your TFIM trust chain configuration.
requestType	Defines the request type sent in the request to the TFIM STS. The default is `http://schemas.xmlsoap.org/ws/2005/02/trust/Validate`.
appliesTo	Defines the applitesTo parameter sent in the request to the TFIM STS. The default is /wsi/FNCEWS40SOAP. This should be specific to the Web service that is being requested.
logLevel	The level at which Java logging should take place. THe options are OFF, FINE, FINER, FINEST, and ALL. The default is ALL.
logFileHandler	The path to the file to write the logs to. The default is `c:/wsdl/log.txt`.
pathToLocalWSDL	The path to the local ws-trust.wsdl file. The default is `c:/wsdl/ws-trust.wsdl`.

4. Delete the old `TFIMFileNetP8Interceptor.jar` file and jar up the updated contents so that the updated configuration file is now contained within the jar as follows:

```
<JAVA_BIN_PATH>\jar.exe cf TFIMFileNetP8Interceptor.jar *
```

Note: `<JAVA_BIN_PATH>` is the path to the JRE on your application server instance. On Windows using WebSphere Application Server, this is `C:\Program Files\IBM\WebSphere\AppServer\java\bin`.

5. Copy the new `TFIMFileNetP8Interceptor.jar` into the `<APP_PATH>\FileNetEngine.ear\wsi-ws.war\WEB-INF\lib` directory.

Note: `<APP_PATH>` is the path to the installed application on your application server instance. On Windows using WebSphere Application Server, this is `C:\Program Files\IBM\WebSphere\AppServer\profiles\<AppServer>\installedApps\<CellName>`.

6. Open the `was-servletconf.xml` file from within the `<APP_PATH>\FileNetEngine.ear\wsi-ws.war\conf` directory.
7. Find the line:

```
<interceptor
class="com.systinet.transport.content_encoding.GzipContentEncodingIn
terceptor" classSpace="root.wasp-impl"
name="gzipContentEncodingInterceptor"/>
```

   and add the following line below it:

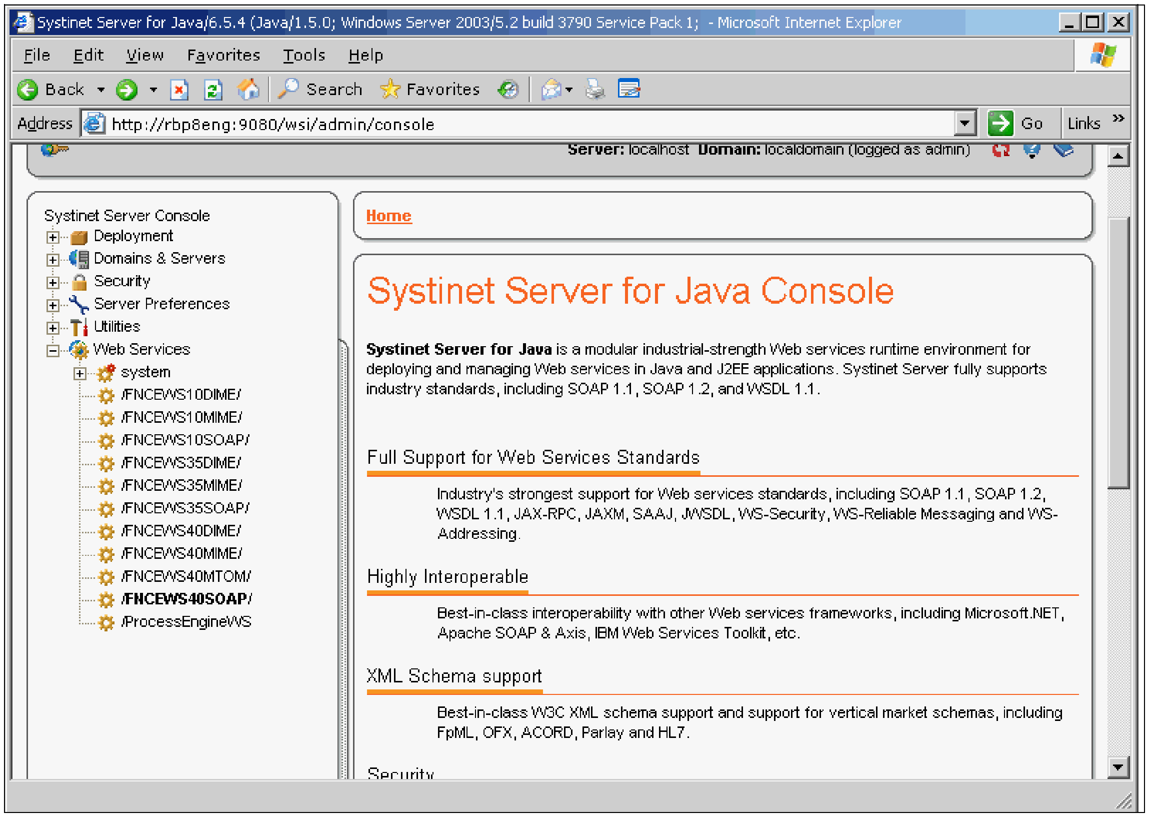
```
<interceptor
class="com.tivoli.soa.tfim.interceptors.TFIMFileNetP8Interceptor"
classSpace="root.wasp-impl" name="TFIMFileNetP8Interceptor"/>
```

8. Save the file.
9. Restart WebSphere Application Server.
10. Open the CEWS listener admin console by browsing to:

```
http://ce_host:ce_port/wsi/
```

   and log in as a user with administrative privileges (the default user is admin).

11. On the left hand side, expand the option that says **Web Services** and click the appropriate Web service, as shown in Figure 6-3. For this example, we are using **FNCEWS40SOAP**.



Systinet Server for Java Console

Systinet Server for Java is a modular industrial-strength Web services runtime environment for deploying and managing Web services in Java and J2EE applications. Systinet Server fully supports industry standards, including SOAP 1.1, SOAP 1.2, and WSDL 1.1.

Full Support for Web Services Standards

Industry's strongest support for Web services standards, including SOAP 1.1, SOAP 1.2, WSDL 1.1, JAX-RPC, JAXM, SAAJ, JWSDL, WS-Security, WS-Reliable Messaging and WS-Addressing.

Highly Interoperable

Best-in-class interoperability with other Web services frameworks, including Microsoft.NET, Apache SOAP & Axis, IBM Web Services Toolkit, etc.

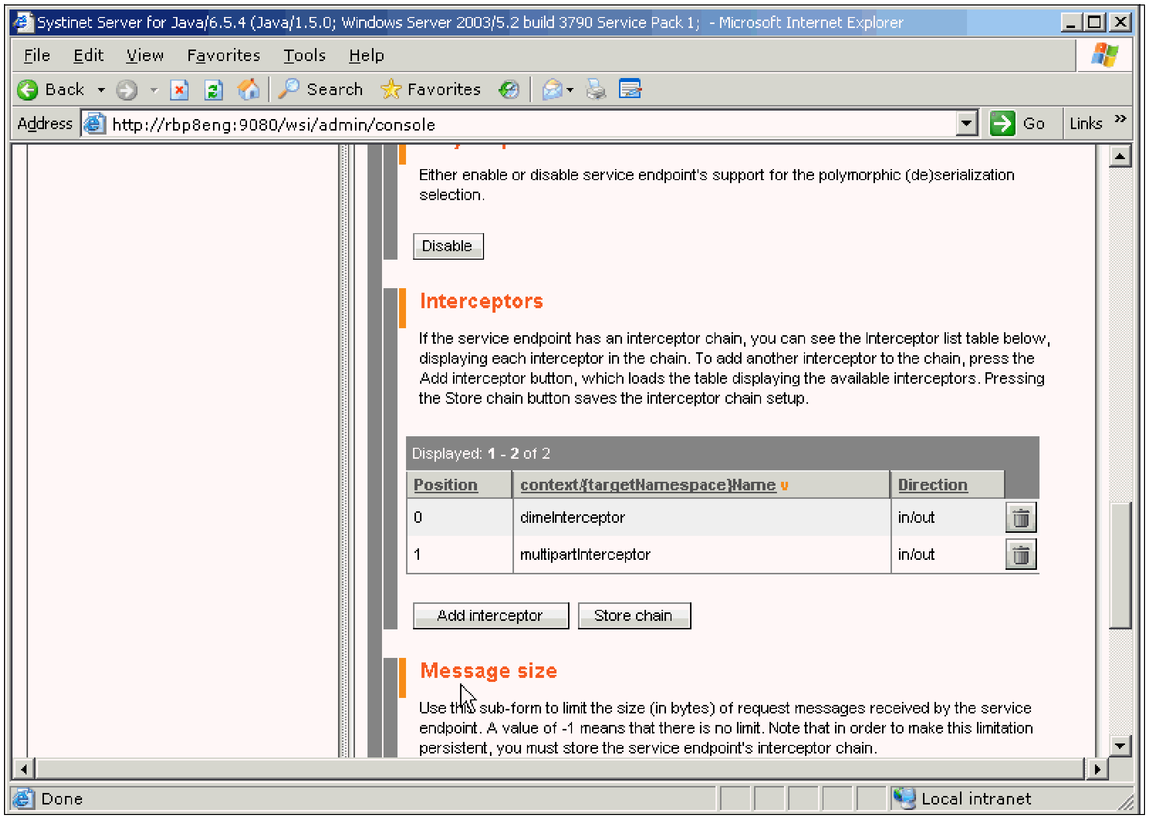
XML Schema support

Best-in-class W3C XML schema support and support for vertical market schemas, including FpML, OFX, ACORD, Parlay and HL7.

Security

Figure 6-3 FileNet P8 Content Engine Web Services listener window

12. Scroll down to the Interceptors section and click **Add interceptor**, as shown in Figure 6-4.


Systinet Server for Java/6.5.4 (Java/1.5.0; Windows Server 2003/5.2 build 3790 Service Pack 1; - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://rbp8eng:9080/wsi/admin/console
Go Links
Disable
Add interceptor
Store chain
Done
Local intranet

Either enable or disable service endpoint's support for the polymorphic (de)serialization selection.

Interceptors

If the service endpoint has an interceptor chain, you can see the Interceptor list table below, displaying each interceptor in the chain. To add another interceptor to the chain, press the Add interceptor button, which loads the table displaying the available interceptors. Pressing the Store chain button saves the interceptor chain setup.

Displayed: **1 - 2** of 2

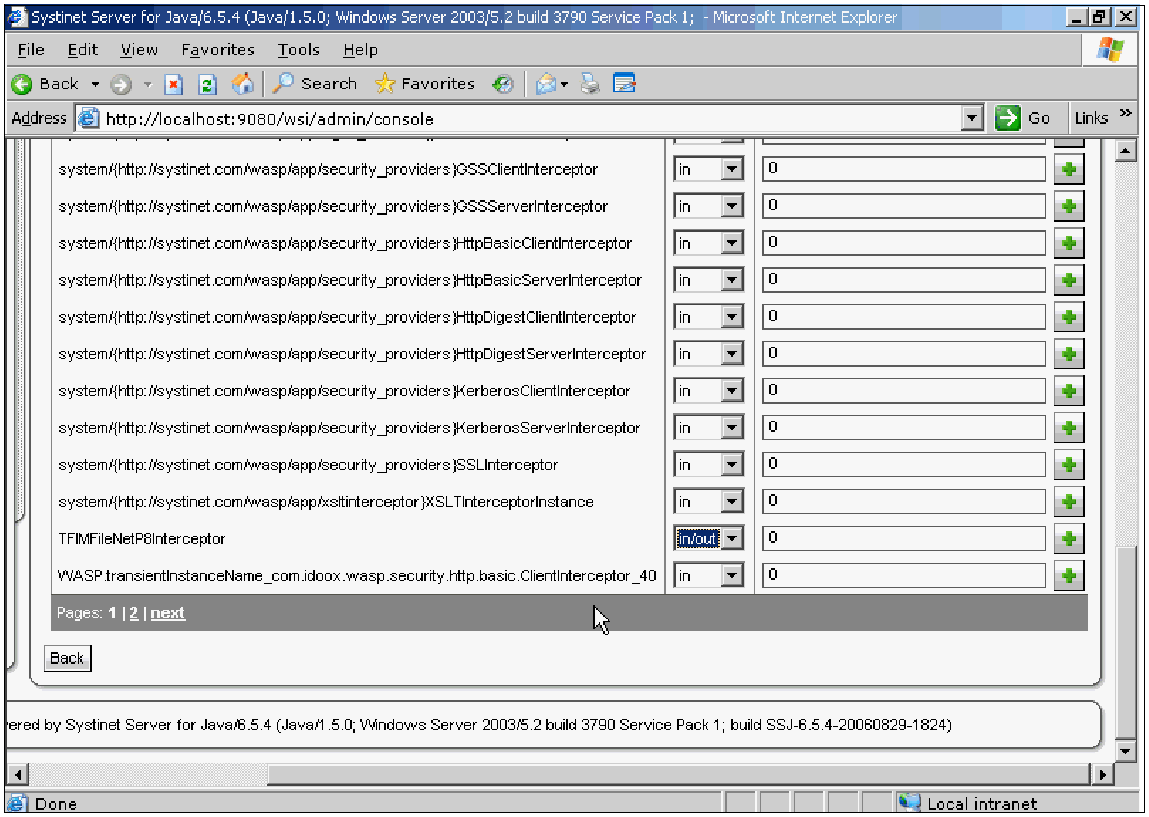
Position	context/{targetNamespace}Name	Direction
0	dimeInterceptor	in/out
1	multipartInterceptor	in/out

Message size

Use this sub-form to limit the size (in bytes) of request messages received by the service endpoint. A value of -1 means that there is no limit. Note that in order to make this limitation persistent, you must store the service endpoint's interceptor chain.

Figure 6-4 FNCEWS40SOAP Web services Interceptors configuration window

13. Locate the TFIMFileNetP8Interceptor on the Interceptors page, modify the direction to be in/out, the position to be 0, and click the **+** button, as shown in Figure 6-5.


Systinet Server for Java/6.5.4 (Java/1.5.0; Windows Server 2003/5.2 build 3790 Service Pack 1; - Microsoft Internet Explorer

Address http://localhost:9080/wsi/admin/console

system/{http://systinet.com/wasp/app/security_providers}GSSClientInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}GSSServerInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}HttpBasicClientInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}HttpBasicServerInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}HttpDigestClientInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}HttpDigestServerInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}KerberosClientInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}KerberosServerInterceptor	in	0
system/{http://systinet.com/wasp/app/security_providers}SSLInterceptor	in	0
system/{http://systinet.com/wasp/app/xsltinterceptor}XSLTInterceptorInstance	in	0
TFIMFileNetP8Interceptor	in/out	0
WASP.transientInstanceName_com.idoox.wasp.security.http.basic.ClientInterceptor_40	in	0

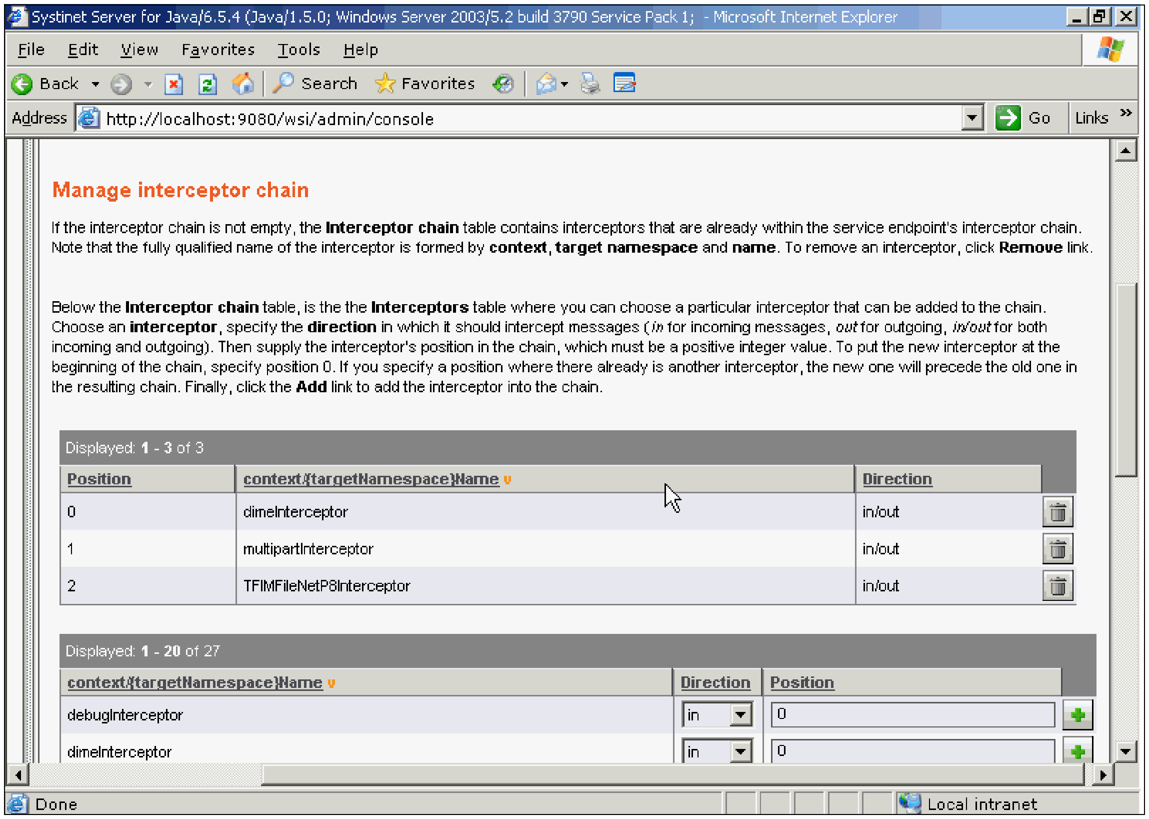
Pages: 1 | 2 | next

Back

ered by Systinet Server for Java/6.5.4 (Java/1.5.0; Windows Server 2003/5.2 build 3790 Service Pack 1; build SSJ-6.5.4-20060829-1824)

Figure 6-5 P8 FileNet Content Engine Web Services listener interceptors window

14. The TFIMFileNetP8Interceptor should now appear in the Interceptor chain, as shown in Figure 6-6. Scroll to the bottom of the window and click **Back**.


Systinet Server for Java/6.5.4 (Java/1.5.0; Windows Server 2003/5.2 build 3790 Service Pack 1; - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:9080/wsi/admin/console
Go Links
Done
Local intranet

Manage interceptor chain

If the interceptor chain is not empty, the **Interceptor chain** table contains interceptors that are already within the service endpoint's interceptor chain. Note that the fully qualified name of the interceptor is formed by **context**, **target namespace** and **name**. To remove an interceptor, click **Remove** link.

Below the **Interceptor chain** table, is the the **Interceptors** table where you can choose a particular interceptor that can be added to the chain. Choose an **interceptor**, specify the **direction** in which it should intercept messages (*in* for incoming messages, *out* for outgoing, *in/out* for both incoming and outgoing). Then supply the interceptor's position in the chain, which must be a positive integer value. To put the new interceptor at the beginning of the chain, specify position 0. If you specify a position where there already is another interceptor, the new one will precede the old one in the resulting chain. Finally, click the **Add** link to add the interceptor into the chain.

Displayed: **1 - 3** of 3

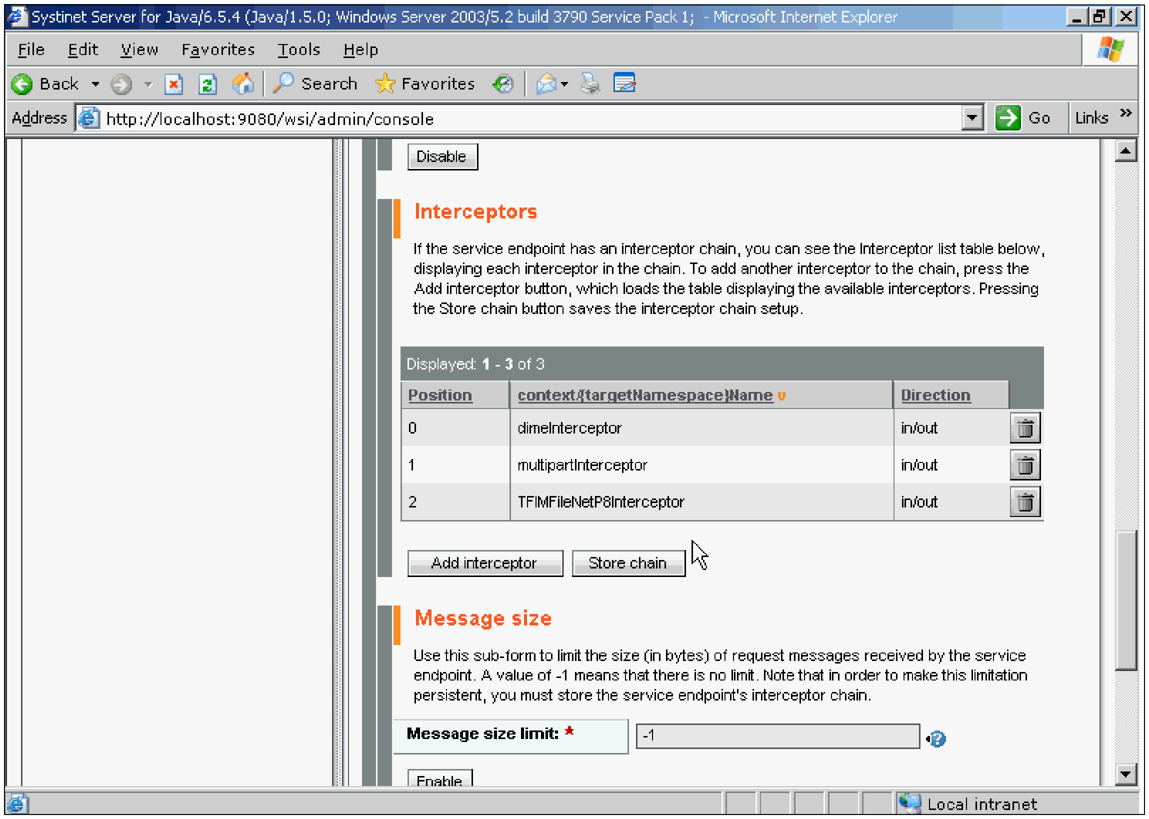
Position	context/{targetNamespace}Name	Direction
0	dimeInterceptor	in/out
1	multipartInterceptor	in/out
2	TFIMFileNetP8Interceptor	in/out

Displayed: **1 - 20** of 27

context/{targetNamespace}Name	Direction	Position
debugInterceptor	in	0
dimeInterceptor	in	0

Figure 6-6 FileNet P8 Content Engine Web Service listener interceptor chain

15. In the Interceptors section, click **Store Chain**, as shown in Figure 6-7 on page 199.


Systinet Server for Java/6.5.4 (Java/1.5.0; Windows Server 2003/5.2 build 3790 Service Pack 1; - Microsoft Internet Explorer

http://localhost:9080/wsi/admin/console

Interceptors

If the service endpoint has an interceptor chain, you can see the Interceptor list table below, displaying each interceptor in the chain. To add another interceptor to the chain, press the Add interceptor button, which loads the table displaying the available interceptors. Pressing the Store chain button saves the interceptor chain setup.

Displayed: **1** - **3** of 3

Position	context/{targetNamespace}Name	Direction
0	dimeInterceptor	in/out
1	multipartInterceptor	in/out
2	TFIMFileNetP8Interceptor	in/out

Message size

Use this sub-form to limit the size (in bytes) of request messages received by the service endpoint. A value of -1 means that there is no limit. Note that in order to make this limitation persistent, you must store the service endpoint's interceptor chain.

Message size limit: * -1

Figure 6-7 Complete interceptor chain including the TFIMFileNetP8Interceptor

We have now installed and configured the TFIMFileNetP8Interceptor. The TFIMFileNetP8Interceptor will now be invoked for incoming and outgoing messages to the FNCEWS40SOAP Web service.The next step is to create and configure the Federated Identity Manager trust chain.

6.4.2 Configuring a trust chain

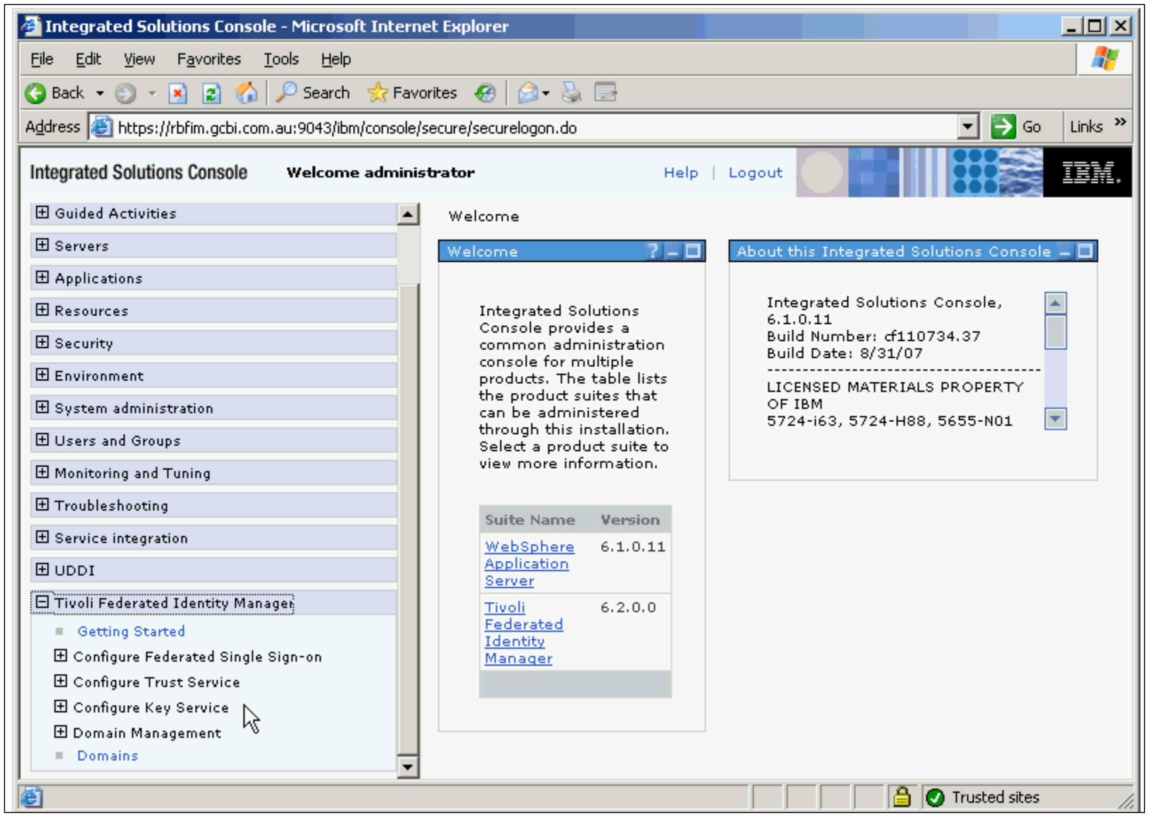
In order to perform the token exchange, we need to make a request to the Federated Identity Manager Trust Service. A Federated Identity Manager trust chain is invoked when an STS request matching the supplied appliesTo and Issuer, among other things, in the RST is received by the trust service. The resulting trust chain, which is invoked, will output an RSTR containing an appropriate token, as per the chain.

Perform the following steps to configure the Federated Identity Manager FileNet trust chain:

1. On the Federated Identity Manager machine, log in to the WebSphere Application Server Integrated Solutions Console using a browser and enter the following URL:

   ```
   http://<fim_host>:<fim_admin_port>/ibm/console
   ```

2. Log in as a user with administrative privileges.
3. In the left hand side menu, expand the **Tivoli Federated Identity Manager** section, as shown in Figure 6-8.



Integrated Solutions Console provides a common administration console for multiple products. The table lists the product suites that can be administered through this installation. Select a product suite to view more information.

Suite Name	Version
WebSphere Application Server	6.1.0.11
Tivoli Federated Identity Manager	6.2.0.0

Integrated Solutions Console, 6.1.0.11
Build Number: cf110734.37
Build Date: 8/31/07

LICENSED MATERIALS PROPERTY OF IBM
5724-i63, 5724-H88, 5655-N01

Figure 6-8 Integrated Solutions Console

4. Expand **Configure Trust Service** and select **Trust Service Chains**, as shown in Figure 6-9 on page 201.

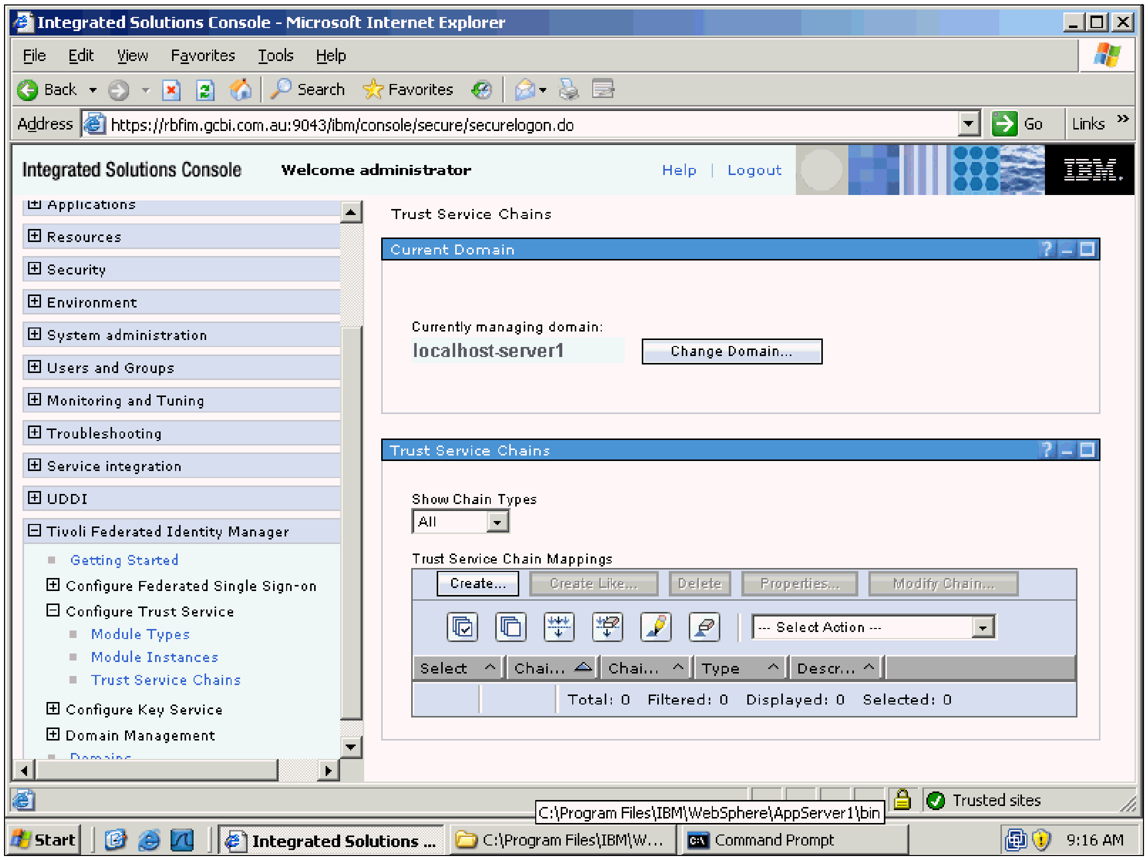
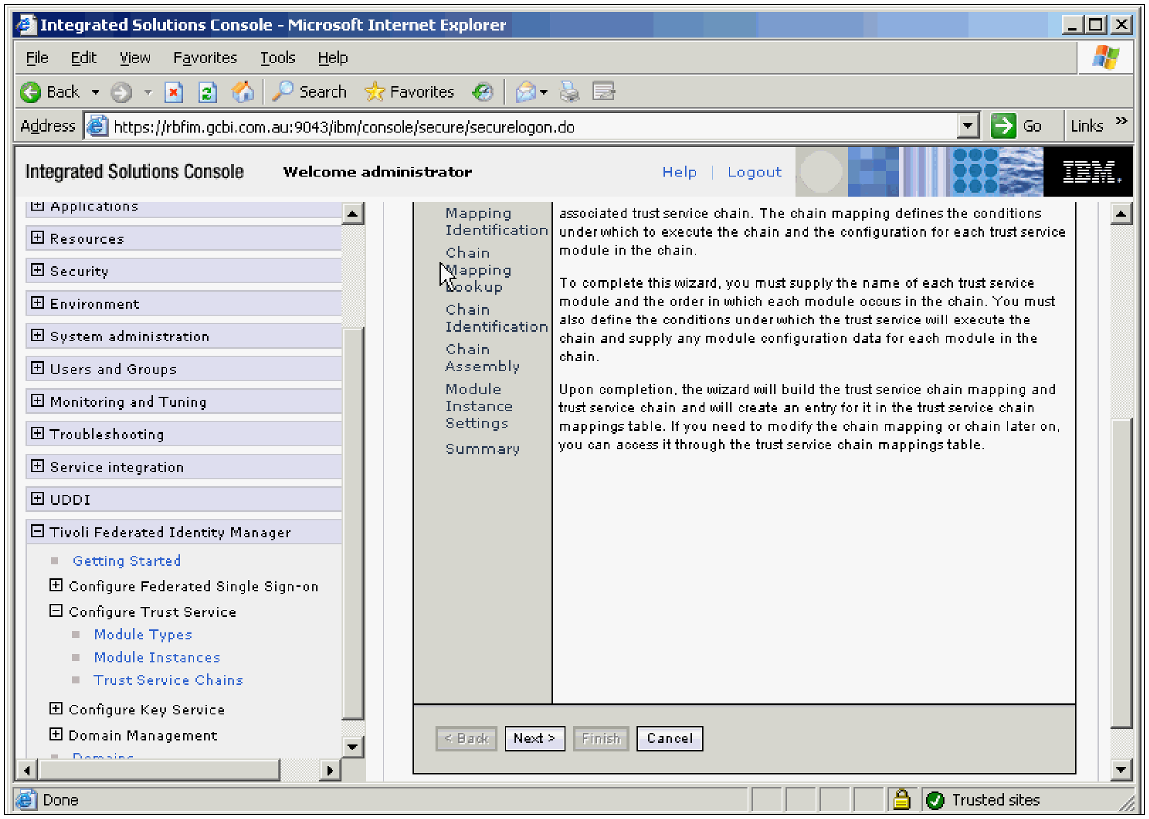


Figure 6-9 Trust Service Chains

5. The Trust Service Chains window is displayed. Click **Create**, as shown in Figure 6-9.

6. The Trust Service Chain Mapping Wizard Introduction window is displayed. Scroll to the bottom and click **Next**, as shown in Figure 6-10.



associated trust service chain. The chain mapping defines the conditions under which to execute the chain and the configuration for each trust service module in the chain.

To complete this wizard, you must supply the name of each trust service module and the order in which each module occurs in the chain. You must also define the conditions under which the trust service will execute the chain and supply any module configuration data for each module in the chain.

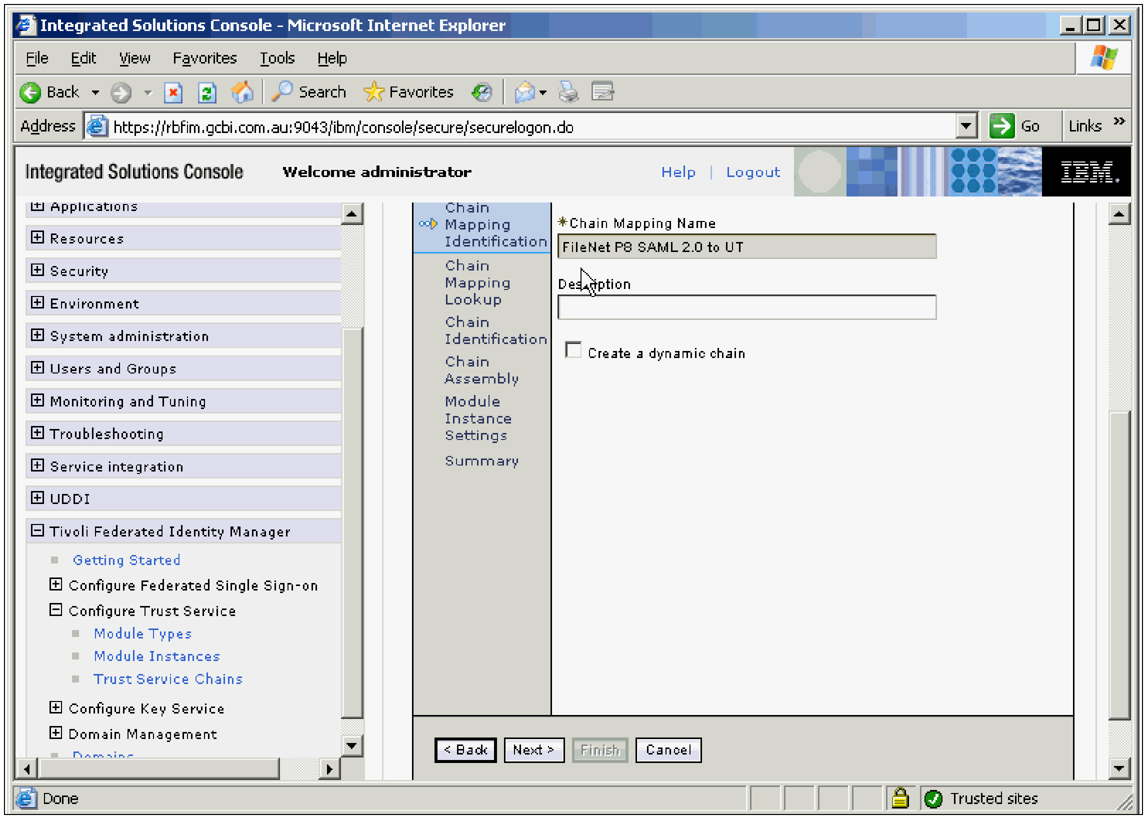
Upon completion, the wizard will build the trust service chain mapping and trust service chain and will create an entry for it in the trust service chain mappings table. If you need to modify the chain mapping or chain later on, you can access it through the trust service chain mappings table.

Figure 6-10 Trust Service Chain Mapping Wizard Introduction window

7. In the Chain Mapping Identification window, enter the following:

Chain Mapping Name	FileNet P8 SAML 2.0 to UT.

Click **Next**, as shown in Figure 6-11 on page 203.



*Chain Mapping Name: FileNet P8 SAML 2.0 to UT

Description:

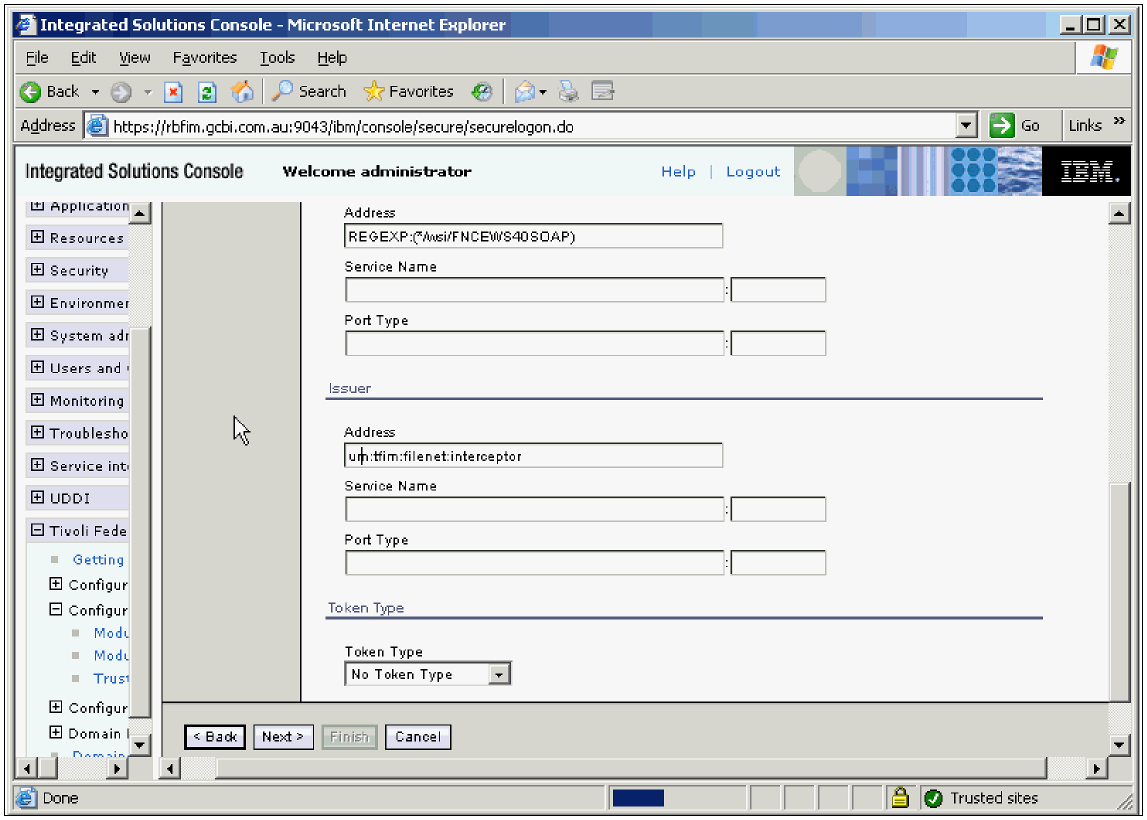
☐ Create a dynamic chain

Figure 6-11 Chain Mapping Lookup window

8. In the Chain Mapping Lookup window, enter the following:

AppliesTo Address	REGEXP:(.*/wsi/FNCEWS40SOAP).
Issuser Address	urn:tfim:filenet:interceptor.

Click the **Next** button, as shown in Figure 6-12.



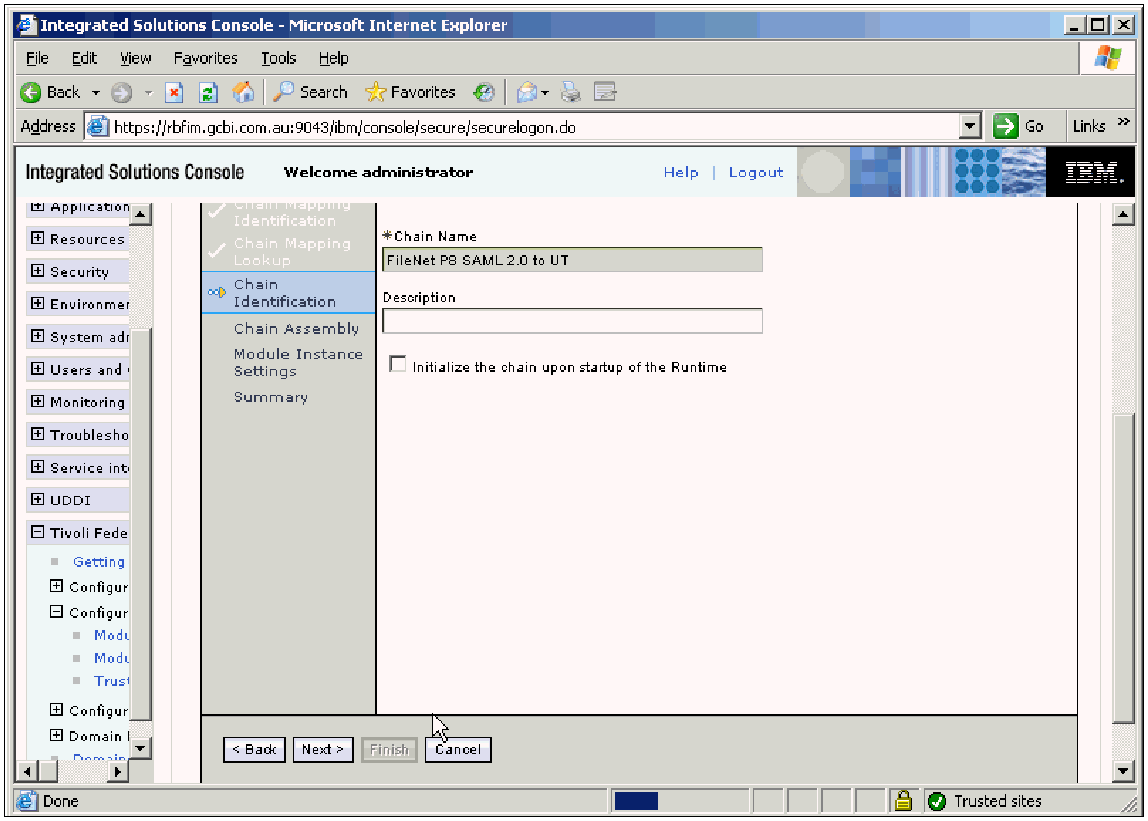
Address
REGEXP:(.*/wsi/FNCEWS40SOAP)
Service Name
Port Type

Issuer
Address
urn:tfim:filenet:interceptor
Service Name
Port Type

Token Type
Token Type
No Token Type

Figure 6-12 Chain Mapping Lookup window

9. In the Chain Identification window, click **Next**, as shown in Figure 6-13 on page 205.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address https://rbfim.gcbi.com.au:9043/ibm/console/secure/securelogon.do
Integrated Solutions Console
Welcome administrator
Help | Logout
Chain Mapping Identification
Chain Mapping Lookup
Chain Identification
Chain Assembly
Module Instance Settings
Summary
< Back
Next >
Finish
Cancel
Done
Trusted sites

*Chain Name: FileNet P8 SAML 2.0 to UT

Description:

☐ Initialize the chain upon startup of the Runtime

Figure 6-13 Chain Identification window

10. In the Chain Assembly window, create a chain consisting of the following:
   – SAML 2.0 Token Validate:

Module Instance	Default SAML 2.0 Token
Mode	Validate

   Click **Add selected module instance to chain**

   – Default Map Module:

Module Instance	Default Map Module
Mode	Map

   Click **Add selected module instance to chain**

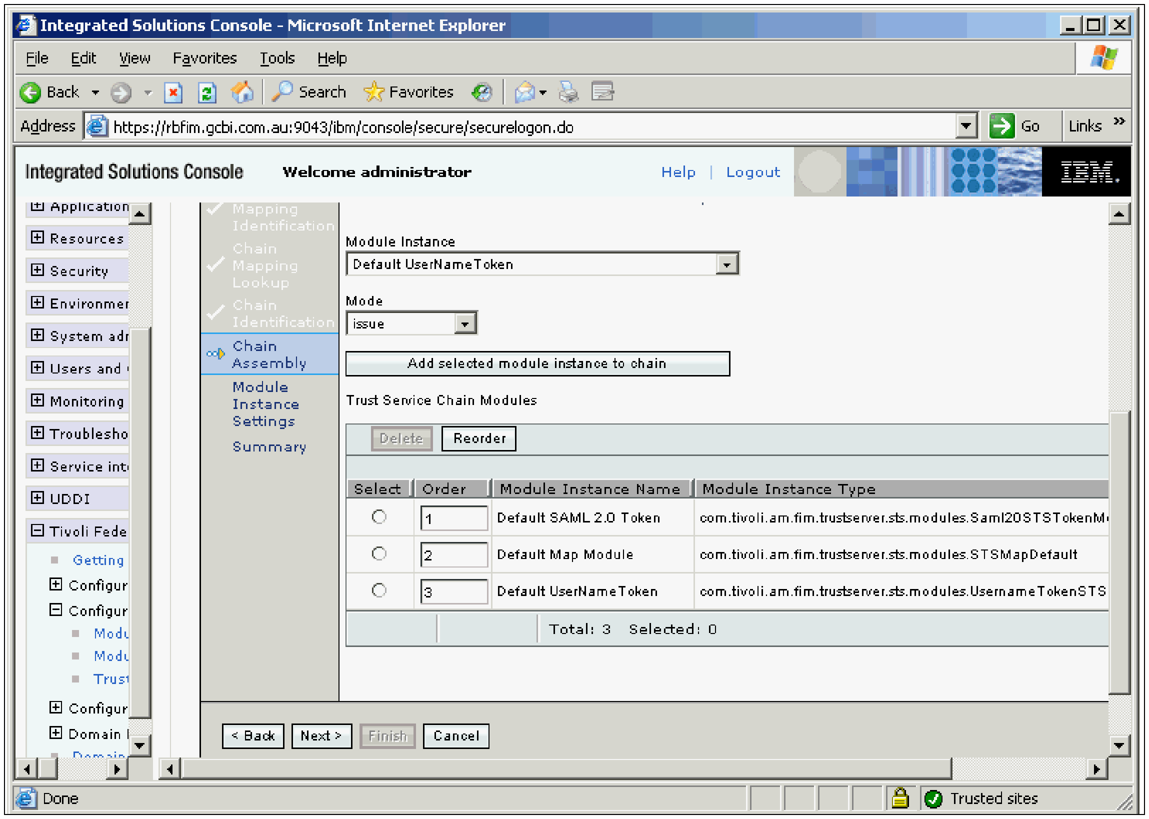
   – Default UserName Token Issue:

Module Instance	Default Username Token
Mode	Issue

Click **Add selected module instance to chain**.

The chain is complete. Click **Next**, as shown in Figure 6-14.

Note: The incoming token can be of any type token that the client can send and that Federated Identity Manager can validate. In this case, we chose to use SAML 2.0.



Module Instance: Default UserNameToken

Mode: issue

Add selected module instance to chain

Trust Service Chain Modules

Select	Order	Module Instance Name	Module Instance Type
○	1	Default SAML 2.0 Token	com.tivoli.am.fim.trustserver.sts.modules.Saml20STSTokenM
○	2	Default Map Module	com.tivoli.am.fim.trustserver.sts.modules.STSMapDefault
○	3	Default UserNameToken	com.tivoli.am.fim.trustserver.sts.modules.UsernameTokenSTS

Total: 3 Selected: 0

Figure 6-14 Chain Assembly window

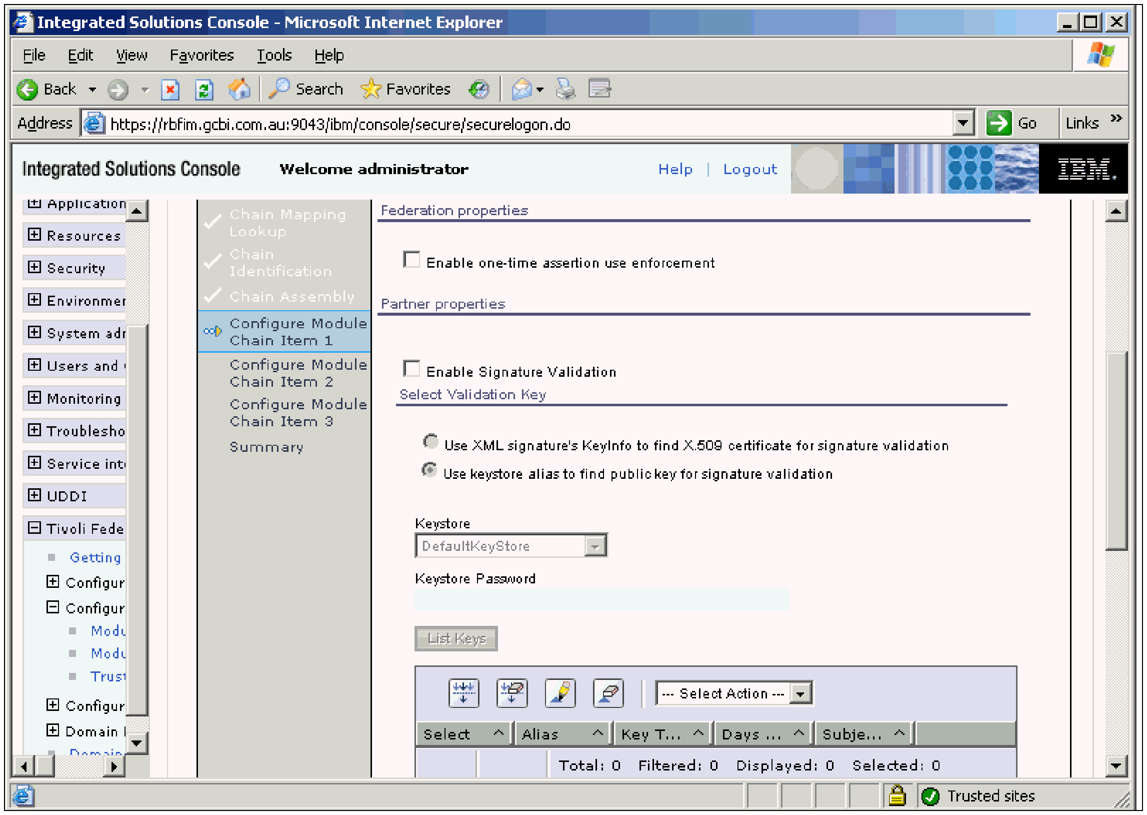
11. In the Security Assertion Markup Language (SAML) Module Configuration window, perform the following steps:

    a. Uncheck **Enable one-time assertion user enforcement**.

    b. Uncheck **Enable Signature Validation**.

    Click **Next**, as shown in Figure 6-15 on page 207.

Note: This example does not use signature validation or one-time assertion enforcement. We recommend that signed SAML assertions and one-time assertion enforcement is used in a production environment.


Integrated Solutions Console - Microsoft Internet Explorer

Address https://rbfim.gcbi.com.au:9043/ibm/console/secure/securelogon.do

Integrated Solutions Console Welcome administrator Help | Logout

Federation properties

☐ Enable one-time assertion use enforcement

Partner properties

☐ Enable Signature Validation

Select Validation Key

○ Use XML signature's KeyInfo to find X.509 certificate for signature validation

● Use keystore alias to find public key for signature validation

Keystore: DefaultKeyStore

Keystore Password

List Keys

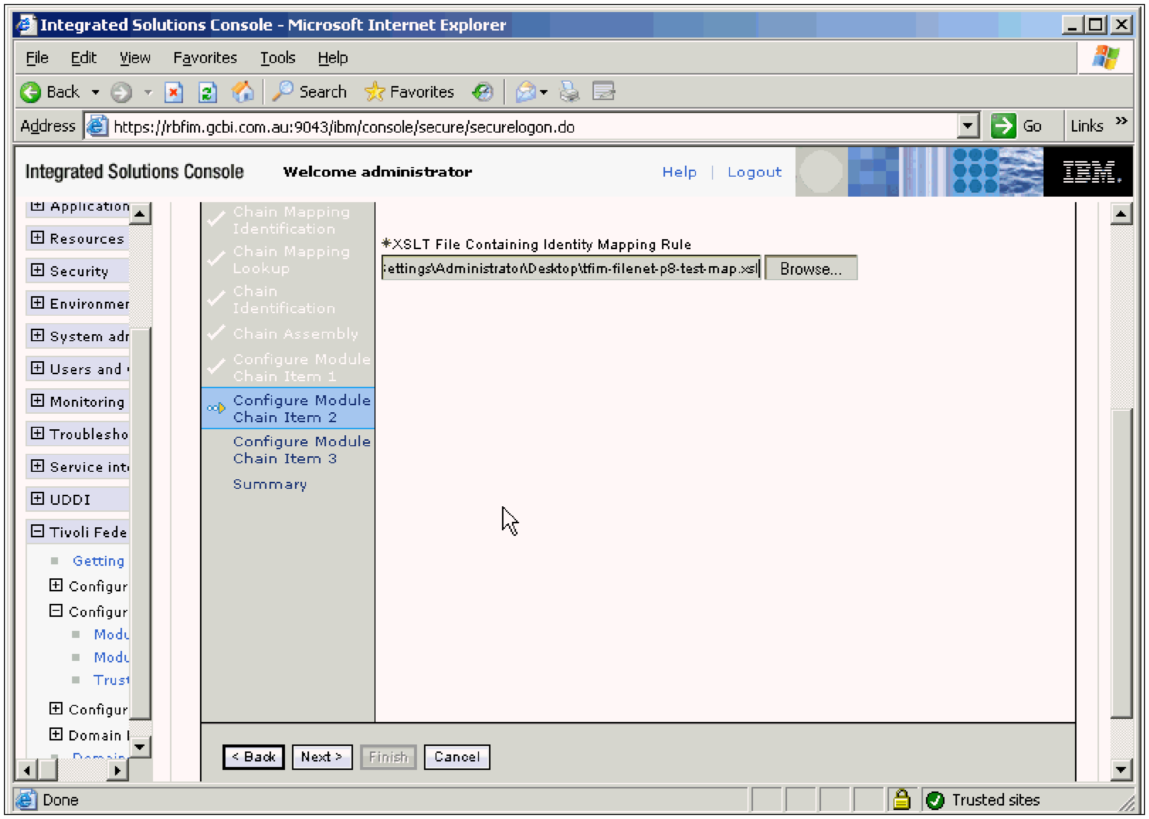
Select	Alias	Key T...	Days ...	Subje...

Total: 0 Filtered: 0 Displayed: 0 Selected: 0

Figure 6-15 Security Assertion Markup Language (SAML) Module Configuration window

12. in the Default Map Module window, perform the following steps:
    - Click **Browse**.
    - Browse to the `tfim-filenet-p8-test-map.xsl` file.
    - Click **Open**.

Click the **Next** button, as shown in Figure 6-16.



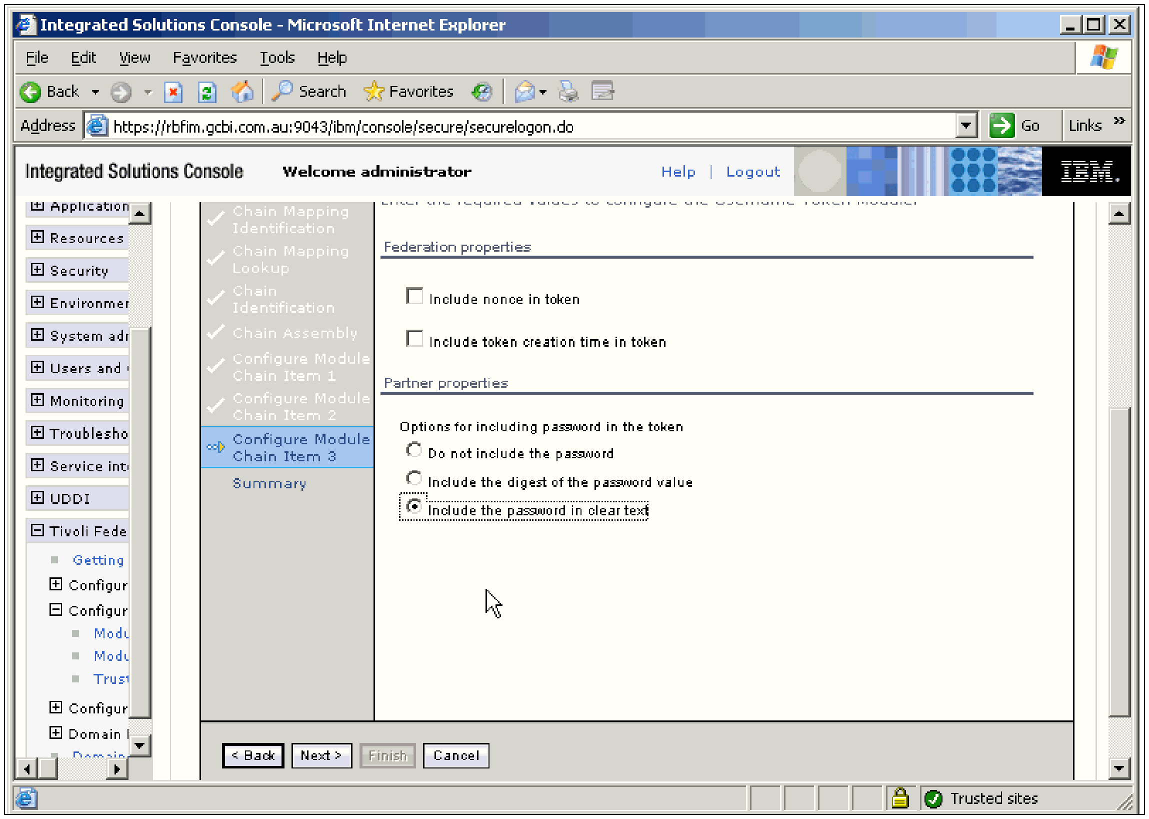
*XSLT File Containing Identity Mapping Rule

ettings\Administrator\Desktop\tfim-filenet-p8-test-map.xsl

Figure 6-16 Default Map Module window

13. In the Username Token Module Configuration window, perform the following:
   - Uncheck **Include nonce in token**.
   - Uncheck **Include token creation time in token**.
   - Select **Include the password in clear text**.

Click **Next**, as shown in Figure 6-17.



Federation properties

- [ ] Include nonce in token
- [ ] Include token creation time in token

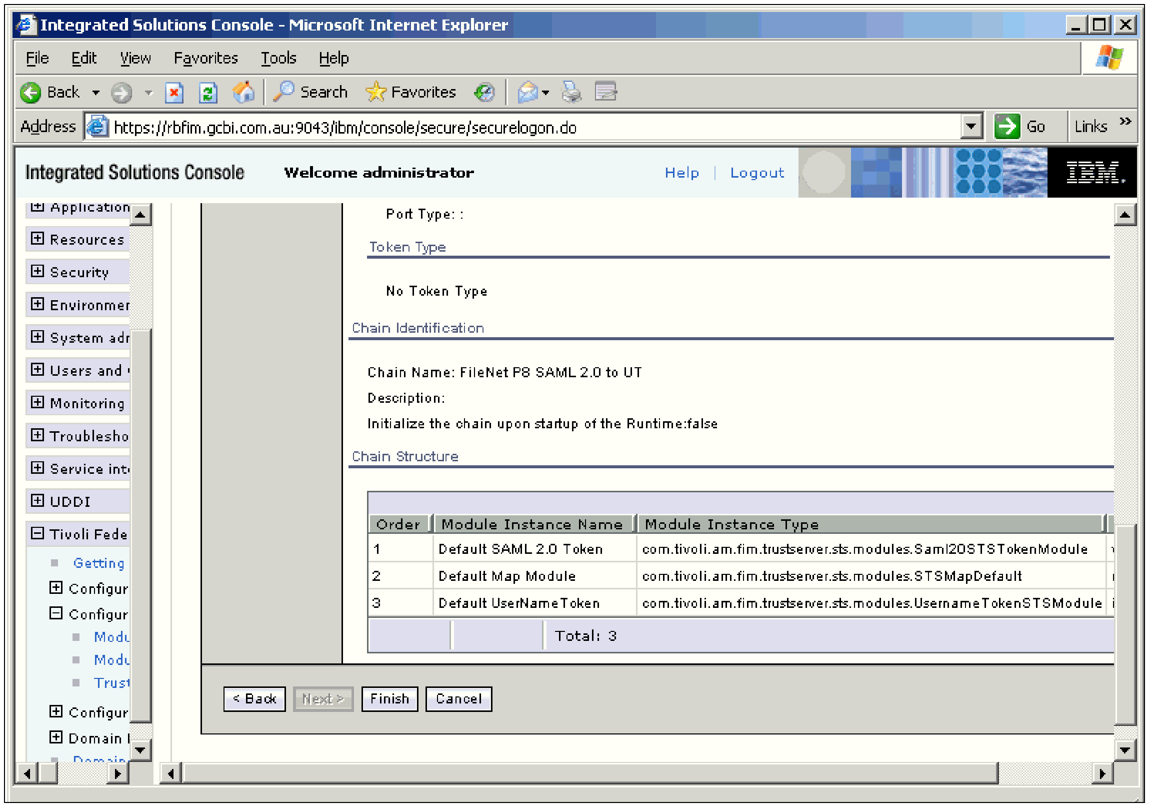
Partner properties

Options for including password in the token

( ) Do not include the password
( ) Include the digest of the password value
(•) Include the password in clear text

Figure 6-17 Username Token Module Configuration window

14. The Summary Page will display all of the information entered during the Trust Service Chain creation wizard. Check and make sure that the information is correct and click **Finish**, as shown in Figure 6-18.



Port Type: :

Token Type

No Token Type

Chain Identification

Chain Name: FileNet P8 SAML 2.0 to UT

Description:

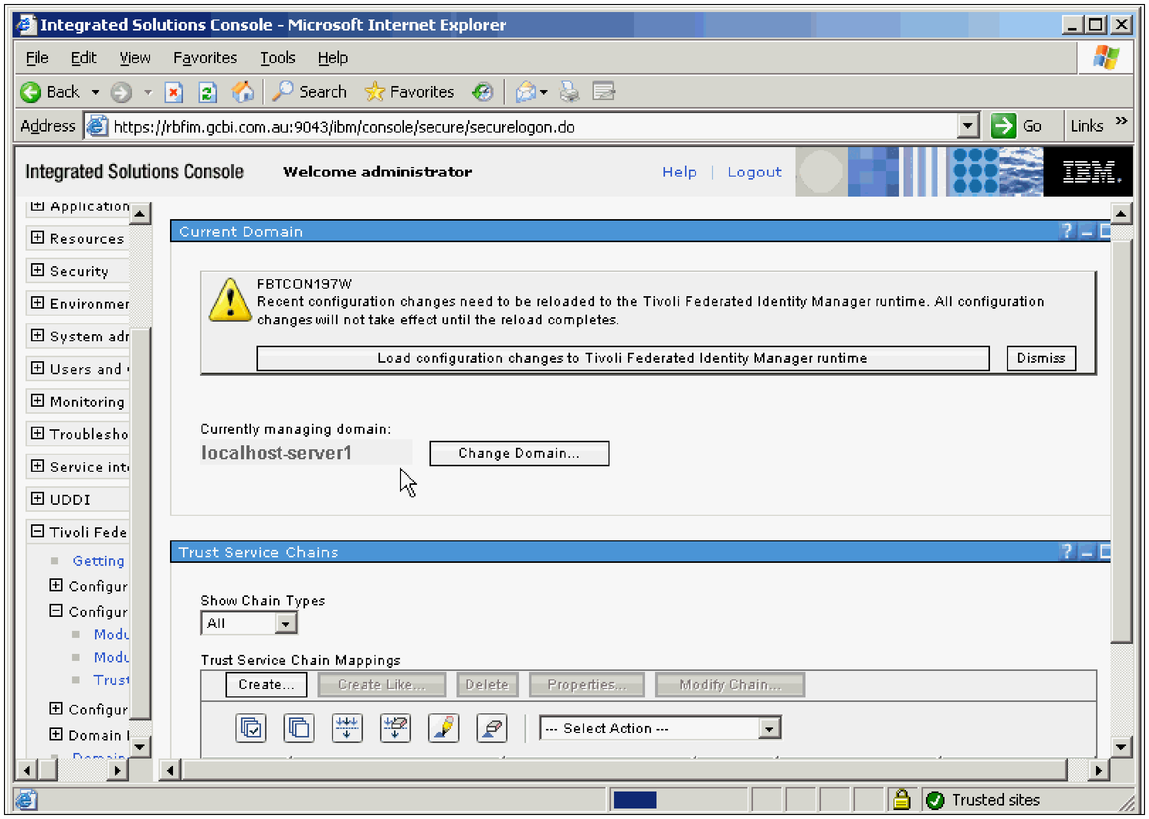
Initialize the chain upon startup of the Runtime:false

Chain Structure

Order	Module Instance Name	Module Instance Type
1	Default SAML 2.0 Token	com.tivoli.am.fim.trustserver.sts.modules.Saml20STSTokenModule
2	Default Map Module	com.tivoli.am.fim.trustserver.sts.modules.STSMapDefault
3	Default UserNameToken	com.tivoli.am.fim.trustserver.sts.modules.UsernameTokenSTSModule
		Total: 3

Figure 6-18 Summary Page window

15. Click **Load configuration changes to Tivoli Federated Identity Manager runtime**, as shown in Figure 6-19.



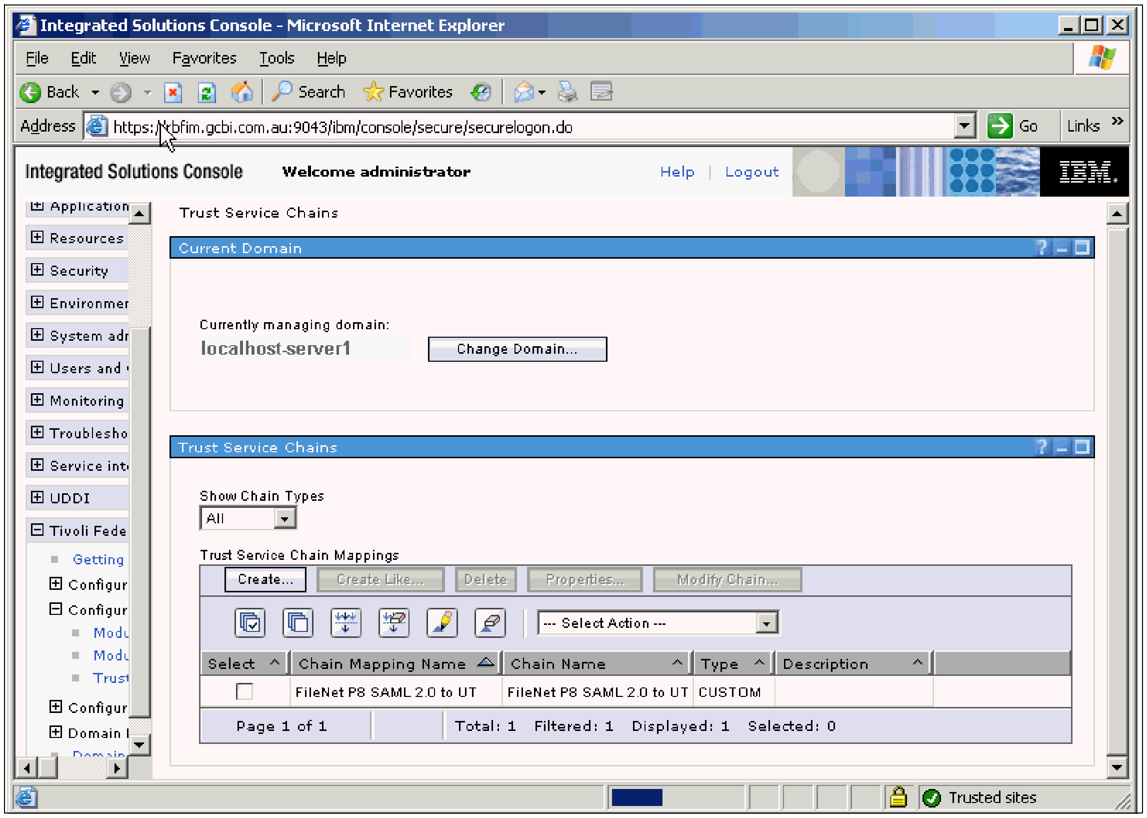
FBTCON197W
Recent configuration changes need to be reloaded to the Tivoli Federated Identity Manager runtime. All configuration changes will not take effect until the reload completes.

Load configuration changes to Tivoli Federated Identity Manager runtime | Dismiss

Currently managing domain:
localhost-server1 | Change Domain...

Figure 6-19 Reload Federated Identity Manager Runtime configuration changes

16. The Federated Identity Manager Trust Chain creation and configuration is complete, as shown in Figure 6-20.



Trust Service Chains

Current Domain

Currently managing domain: localhost-server1

Change Domain...

Trust Service Chains

Show Chain Types: All

Trust Service Chain Mappings

Create... | Create Like... | Delete | Properties... | Modify Chain...

--- Select Action ---

Select	Chain Mapping Name	Chain Name	Type	Description
☐	FileNet P8 SAML 2.0 to UT	FileNet P8 SAML 2.0 to UT	CUSTOM	

Page 1 of 1 | Total: 1 Filtered: 1 Displayed: 1 Selected: 0

Figure 6-20 Completed trust chain

6.5 Testing the approach

In this section, we describe some useful tools and techniques to test this approach.

6.5.1 Sample FileNet P8 CEWS test client

The FileNet P8 Federated Identity Manager CEWS test client shipped with this book can be used to test this scenario. The client was developed to run on WebSphere Application Server V6.1 and utilizes the Federated Identity Manager WSSM component to generate a SAML 2.0 token. The client is a simple series of

JSPs and servlets, which is accessed through a Web interface. The following high level steps need to be performed to install and configure the client:

1. Install and configure the Federated Identity Manager WSSM component onto the WebSphere Application Server instance that hosts the Federated Identity Manager CEWS test client. Refer to the *IBM Tivoli Federated Identity Manager Version 6.2 Installation and Configuration Guide*, SC23-6190, for more details.

   Note: On the client side, the Federated Identity Manager WSSM component is used to generate the incoming SAML 2.0 token.

2. Deploy the sample FileNet P8 Federated Identity Manager CEWS client into WebSphere Application Server. The deployment descriptor has already been configured for use with WSSM and requires authentication before users can access the application.
3. Create a user in the WebSphere Application Server directory, using the Integrated Solutions Console, that represents a user in the FileNet directory (or in the configured Federated Identity Manager mapping rule).
4. Within the Web service client configuration options on the Integrated Solutions Console, add All Authenticated to the AllAuthenticated role on the Security role to user/group mapping window.

We have now completed the high level FileNet P8 Federated Identity Manager CEWS test client installation and configuration process The client is accessible through `http://<hostname>:<host_port>/FNCEWS40Client/`.

Note: The test client was designed to send the Content Engine Web Service request to port 19080 on the localhost. In testing, the TCPMon program was used to capture this request and send it to the appropriate Content Engine using the appropriate port (default 9080).

6.5.2 Information flow

In this section, we show a sample of the messages that are passed in each stage of the Web service request.

Web service client to Federated Identity Manager FileNet P8 Interceptor

Example 6-3 shows a sample SOAP message that is sent from the Web service client to the CEWS. This message is intercepted by the Federated Identity Manager FileNet P8 Interceptor. Note that the request contains a SAML 2.0 assertion.

Example 6-3 Client Web service request containing SAML 2.0 assertion

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Header>
  <wsse:Security soapenv:mustUnderstand="1"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wss
ecurity-secext-1.0.xsd">
   <saml:Assertion ID="uuidde29bcff-011c-fdfe-40c0-cbe9e757b9b5"
IssueInstant="2008-10-08T20:33:09Z" Version="2.0"
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
    <saml:Issuer
Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
     urn:itfim:wssm:callbackhandler:jaas
    </saml:Issuer>
    <saml:Conditions NotBefore="2008-10-08T19:33:09Z"
NotOnOrAfter="2008-10-08T21:33:09Z" />
    <saml:Subject>
     <saml:NameID>jbloggs@qisa.com.au</saml:NameID>
    </saml:Subject>
    <saml:AuthnStatement AuthnInstant="2008-10-08T20:33:09Z">
     <saml:AuthnContext>
      <saml:AuthnContextClassRef>
       urn:oasis:names:tc:SAML:2.0:ac:classes:Password
      </saml:AuthnContextClassRef>
     </saml:AuthnContext>
    </saml:AuthnStatement>
    <saml:AttributeStatement>
     <saml:Attribute Name="com.ibm.ws.security.auth.WSCredentialImpl"
NameFormat="com.ibm.ws.security.auth.WSCredentialImpl">
      <saml:AttributeValue>
       com.ibm.ws.security.auth.WSCredentialImpl@66aa66aa
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="type"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>SOAP</saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="process.serverName"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>
       rbwsclientNode01Cell:rbwsclientNode01:server1
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="security.authMechOID"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>
       oid:1.3.18.0.2.30.2
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="java.naming.provider.url"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>
       corbaloc:iiop:rbwsclient:2809/WsnAdminNameService
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="host"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>rbwsclient</saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="u"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="expire"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>
       1223505187203
      </saml:AttributeValue>
      <saml:AttributeValue>
       1223505185078
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="port"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
      <saml:AttributeValue>8880</saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="u"
NameFormat="com.ibm.ws.security.token.AuthenticationTokenImpl">
      <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute
Name="com.ibm.wsspi.security.cred.longSecurityName"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
       uid=jbloggs@qisa.com.au,o=defaultWIMFileBasedRealm
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.uniqueId"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.securityName"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
       jbloggs@qisa.com.au
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.primaryGroupId"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue />
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.expiration"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
       1223505185078
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.forwardable"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>true</saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.oid"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
```

```
       oid:1.3.18.0.2.30.2
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="u"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
      </saml:AttributeValue>
     </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.realm"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
      <saml:AttributeValue>
       defaultWIMFileBasedRealm
      </saml:AttributeValue>
     </saml:Attribute>
    </saml:AttributeStatement>
   </saml:Assertion>
  </wsse:Security>
 </soapenv:Header>
 <soapenv:Body>
  <p857:ExecuteChangesRequest refresh="1"
xmlns:p857="http://www.filenet.com/ns/fnce/2006/11/ws/schema">
   <p857:ChangeRequest id="1" updateSequenceNumber="0">
    <p857:TargetSpecification classId="ObjectStore" objectId="ECMOS"
serializationDuplicate="0" />
    <p857:Action autoUniqueContainmentName="0" classId="folder"
defineSecurityParentage="0" xsi:type="p857:CreateAction" />
    <p857:ActionProperties>
     <p857:Property propertyId="FolderName" settable="0"
xsi:type="p857:SingletonString">
      <p857:Value>Case15</p857:Value>
     </p857:Property>
     <p857:Property propertyId="Parent" settable="0"
xsi:type="p857:SingletonObject">
      <p857:Value classId="Folder" itemIndex="0" objectStore="ECMOS"
path="/" serializationDuplicate="0" xsi:type="p857:ObjectSpecification"
/>
     </p857:Property>
    </p857:ActionProperties>
    <p857:RefreshFilter levelDependents="0" maxElements="0"
maxRecursion="0">
     <p857:ExcludeProperties>
      DateCreated
     </p857:ExcludeProperties>
     <p857:ExcludeProperties>
      DateLastModified
     </p857:ExcludeProperties>
    </p857:RefreshFilter>
   </p857:ChangeRequest>
  </p857:ExecuteChangesRequest>
 </soapenv:Body>
</soapenv:Envelope>
```

Federated Identity Manager FileNet P8 Interceptor to STS

Example 6-4 shows a sample SOAP message that is sent to the Federated Identity Manager STS from the Federated Identity Manager FileNet P8 Interceptor to perform token exchange.

Example 6-4 WS-Trust request sent from Federated Identity Manager FileNet P8 Interceptor to STS

```
<e:Envelope xmlns:e="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:d="http://www.w3.org/2001/XMLSchema"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns:wn0="http://schemas.xmlsoap.org/ws/2005/02/trust">
 <e:Body>
  <wn0:RequestSecurityToken i:type="wn0:RequestSecurityTokenType">
   <wn0:RequestType>
    http://schemas.xmlsoap.org/ws/2005/02/trust/Validate
   </wn0:RequestType>
   <wn0:Issuer
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
    <wsa:Address>urn:itfim:filenet:interceptor</wsa:Address>
   </wn0:Issuer>
   <wsp:AppliesTo
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy">
    <wsa:EndpointReference>
     <wsa:Address>
      http://rbp8eng.gcbi.com.au:9080/wsi/FNCEWS40SOAP
     </wsa:Address>
    </wsa:EndpointReference>
   </wsp:AppliesTo>
   <wn0:Base>
    <saml:Assertion ID="uuidde29bcff-011c-fdfe-40c0-cbe9e757b9b5"
IssueInstant="2008-10-08T20:33:09Z" Version="2.0"
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
     <saml:Issuer
Format="urn:oasis:names:tc:SAML:2.0:nameid-format:entity">
```

```
      urn:itfim:wssm:callbackhandler:jaas
     </saml:Issuer>
     <saml:Conditions NotBefore="2008-10-08T19:33:09Z"
NotOnOrAfter="2008-10-08T21:33:09Z" />
     <saml:Subject>
      <saml:NameID>jbloggs@qisa.com.au</saml:NameID>
     </saml:Subject>
     <saml:AuthnStatement AuthnInstant="2008-10-08T20:33:09Z">
      <saml:AuthnContext>
       <saml:AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:Password
       </saml:AuthnContextClassRef>
      </saml:AuthnContext>
     </saml:AuthnStatement>
     <saml:AttributeStatement>
      <saml:Attribute Name="com.ibm.ws.security.auth.WSCredentialImpl"
NameFormat="com.ibm.ws.security.auth.WSCredentialImpl">
       <saml:AttributeValue>
        com.ibm.ws.security.auth.WSCredentialImpl@66aa66aa
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="type"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        SOAP
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="process.serverName"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        rbwsclientNode01Cell:rbwsclientNode01:server1
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="security.authMechOID"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        oid:1.3.18.0.2.30.2
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="java.naming.provider.url"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        corbaloc:iiop:rbwsclient:2809/WsnAdminNameService
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="host"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        rbwsclient
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="u"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="expire"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        1223505187203
       </saml:AttributeValue>
       <saml:AttributeValue>
        1223505185078
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="port"
NameFormat="com.ibm.ws.security.token.SingleSignonTokenImpl">
       <saml:AttributeValue>
        8880
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="u"
NameFormat="com.ibm.ws.security.token.AuthenticationTokenImpl">
       <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute
       Name="com.ibm.wsspi.security.cred.longSecurityName"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
        uid=jbloggs@qisa.com.au,o=defaultWIMFileBasedRealm
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="com.ibm.wsspi.security.cred.uniqueId"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
```

```
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="com.ibm.wsspi.security.cred.securityName"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
        jbloggs@qisa.com.au
       </saml:AttributeValue>
      </saml:Attribute>
     <saml:Attribute Name="com.ibm.wsspi.security.cred.primaryGroupId"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue />
      </saml:Attribute>
      <saml:Attribute Name="com.ibm.wsspi.security.cred.expiration"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
        1223505185078
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="com.ibm.wsspi.security.cred.forwardable"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
        true
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="com.ibm.wsspi.security.cred.oid"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
        oid:1.3.18.0.2.30.2
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="u"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
user:defaultWIMFileBasedRealm/uid=jbloggs@qisa.com.au,o=defaultWIMFileB
asedRealm
       </saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="com.ibm.wsspi.security.cred.realm"
NameFormat="com.ibm.ws.security.token.AuthorizationTokenImpl">
       <saml:AttributeValue>
        defaultWIMFileBasedRealm
       </saml:AttributeValue>
      </saml:Attribute>
     </saml:AttributeStatement>
    </saml:Assertion>
   </wn0:Base>
  </wn0:RequestSecurityToken>
 </e:Body>
</e:Envelope>
```

STS to Federated Identity Manager P8 Interceptor Response

Example 6-5 shows a sample SOAP message response that the Federated Identity Manager STS Interceptor sends to the FileNetP8Interceptor, containing the Username Token.

Example 6-5 WS-Trust response sent from STS to Federated Identity Manager FileNet P8 Interceptor

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Header />
 <soapenv:Body>
  <wst:RequestSecurityTokenResponse Context=""
wsu:Id="uuidde28eb66-011c-1755-85a3-8258a1caa7e7"
xmlns:wst="http://schemas.xmlsoap.org/ws/2005/02/trust"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-utility-1.0.xsd">
   <wsp:AppliesTo
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy">
    <wsa:EndpointReference>
     <wsa:Address>
      http://rbp8eng.gcbi.com.au:9080/wsi/FNCEWS40SOAP
     </wsa:Address>
    </wsa:EndpointReference>
   </wsp:AppliesTo>
   <wst:RequestedSecurityToken>
    <wss:UsernameToken
wsu:Id="usernamede28ecbe-011c-19b6-be5c-8258a1caa7e7"
xmlns:wss="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-utility-1.0.xsd">
     <wss:Username>qisapartner</wss:Username>
    <wss:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText">
      password
     </wss:Password>
    </wss:UsernameToken>
   </wst:RequestedSecurityToken>
   <wst:RequestedAttachedReference
xmlns:wss="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd">
    <wss:SecurityTokenReference>
     <wss:Reference URI="#usernamede28ecbe-011c-19b6-be5c-8258a1caa7e7"
ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-user
name-token-profile-1.0#UsernameToken"
xmlns:wss="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd" />
    </wss:SecurityTokenReference>
   </wst:RequestedAttachedReference>
   <wst:Status>
    <wst:Code>
     http://schemas.xmlsoap.org/ws/2005/02/trust/status/valid
    </wst:Code>
   </wst:Status>
  </wst:RequestSecurityTokenResponse>
 </soapenv:Body>
</soapenv:Envelope>
```

Federated Identity Manager FileNet P8 Interceptor to CEWS

Example 6-6 shows a sample SOAP message that is sent to the CEWS from the Federated Identity Manager FileNet P8 Interceptor. Note that the request now contains a Username Token.

Example 6-6 Client Web service request containing Username Token

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Header>
  <wsse:Security xmlns="http://schemas.xmlsoap.org/ws/2002/12/secext"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wss
ecurity-secext-1.0.xsd">
```

```
   <wss:UsernameToken
xmlns:wss="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-secext-1.0.xsd"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wsse
curity-utility-1.0.xsd"
wsu:Id="usernamede28ecbe-011c-19b6-be5c-8258a1caa7e7">
    <wss:Username>qisapartner</wss:Username>
    <wss:Password
Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-
token-profile-1.0#PasswordText">
     password
    </wss:Password>
   </wss:UsernameToken>
  </wsse:Security>
 </soapenv:Header>
 <soapenv:Body>
  <p857:ExecuteChangesRequest refresh="1"
xmlns:p857="http://www.filenet.com/ns/fnce/2006/11/ws/schema">
   <p857:ChangeRequest id="1" updateSequenceNumber="0">
    <p857:TargetSpecification classId="ObjectStore" objectId="ECMOS"
serializationDuplicate="0" />
    <p857:Action autoUniqueContainmentName="0" classId="folder"
defineSecurityParentage="0" xsi:type="p857:CreateAction" />
    <p857:ActionProperties>
     <p857:Property propertyId="FolderName" settable="0"
xsi:type="p857:SingletonString">
      <p857:Value>Case15</p857:Value>
     </p857:Property>
     <p857:Property propertyId="Parent" settable="0"
xsi:type="p857:SingletonObject">
      <p857:Value classId="Folder" itemIndex="0" objectStore="ECMOS"
path="/" serializationDuplicate="0" xsi:type="p857:ObjectSpecification"
/>
     </p857:Property>
    </p857:ActionProperties>
    <p857:RefreshFilter levelDependents="0" maxElements="0"
maxRecursion="0">
     <p857:ExcludeProperties>
      DateCreated
     </p857:ExcludeProperties>
     <p857:ExcludeProperties>
      DateLastModified
     </p857:ExcludeProperties>
    </p857:RefreshFilter>
   </p857:ChangeRequest>
  </p857:ExecuteChangesRequest>
 </soapenv:Body>
</soapenv:Envelope>
```

6.6 Conclusion

In this chapter, we showed how the combination of the FileNet P8 platform and Federated Identity Manager can be utilized to further enhance Web services security interoperability in SOA environments within and external to an organization using open standards.

We showed how Federated Identity Manager can extend the number of tokens supported by the FileNet P8 Web services platform. This was illustrated through the GCBI example and their partnership with QISA by adding SAML 2.0 token support to the Content Engine Web Service interface. We also showed how an external partner's users can be transported across organizational boundaries and mapped into local users through the use of Federated Identity Manager mapping modules.

A

Content Engine Web Service client

In this appendix, we describe the steps needed to install and configure the sample FileNet Content Engine Web Service (CEWS) SAML 2.0 Client using WebSphere Application Server and the Tivoli Federated Identity Manager Web services security management component.

Overview

The FileNet CEWS SAML 2.0 Client shipped with this book can be used to test the configuration of the Federated Identity Manager FileNet P8 Interceptor. This example client is configured to use the Federated Identity Manager Web services security management component, configured for the WebSphere Application Server instance that the client is running on, to generate the SAML 2.0 token sent by the client. In order to achieve this outcome, we assume that a WebSphere Application Server instance to run the client has been installed and created, and the Federated Identity Manager Web services security management component has been installed on the same WebSphere Application Server machine.

Installation and configuration instructions

In this section, we describe the instructions to install and configure the FileNet CEWS SAML 2.0 Client (FNCEWS40SAML20ClientEAR.ear) into a Federated Identity Manager Web services security management enabled WebSphere Application Server instance. This includes:

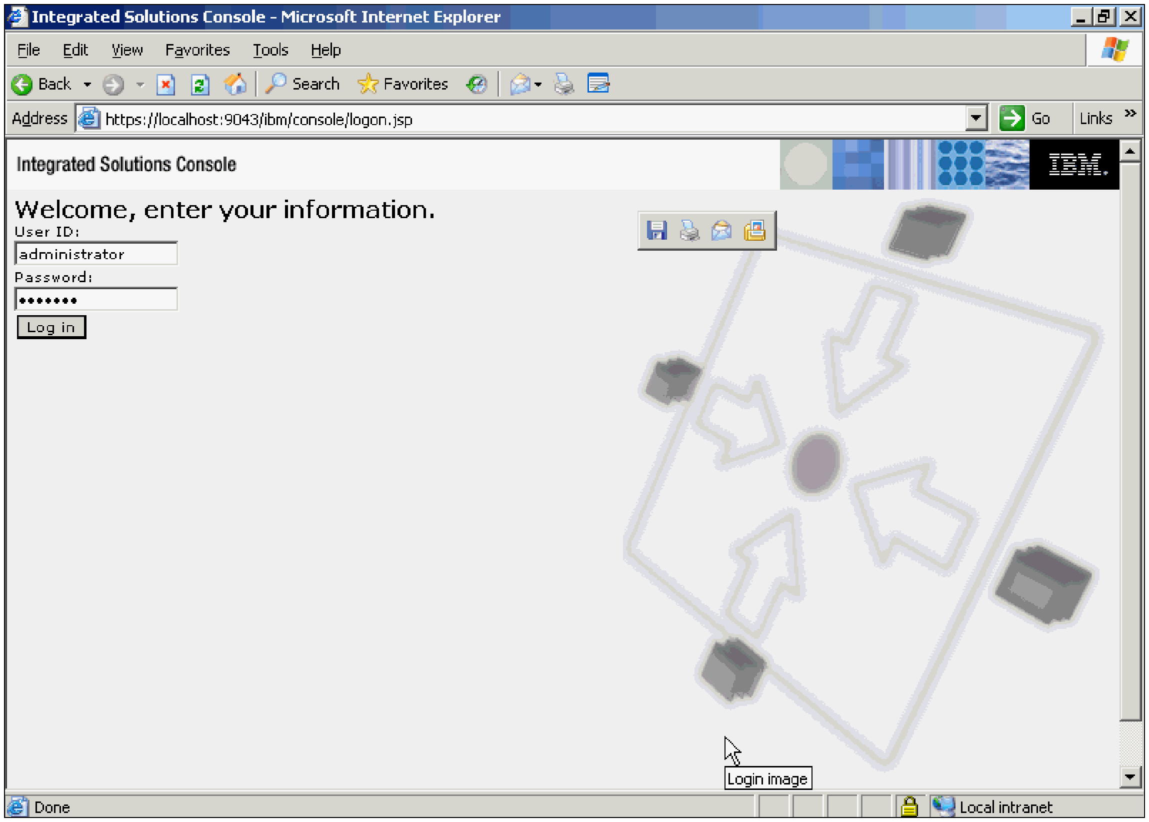
- "Setting the WebSphere variables for WSSM" on page 229
- "Configuring the WSSM shared library" on page 233
- "Configuring a classloader for the application server" on page 237
- "Installing the sample FileNet CEWS SAML 2.0 Client application" on page 245
- "Setting up a FileNet CEWS SAML 2.0 Client user and mapping a role" on page 254
- "Configuring TCPMon to relay FileNet CEWS SAML 2.0 Client requests" on page 266

Note: This client requires WebSphere Application Server V6.1.0.11 to ensure that it can interoperate with the TFIMFileNetP8Interceptor included with this book.

Setting the WebSphere variables for WSSM

In order for the application to use Federated Identity Manager Web services security management to generate a SAML 2.0 token, we need to define a shared library in WebSphere Application Server. To do this, perform the following steps:

1. Start the WebSphere Application Server administrative console and log in, if necessary, as shown in Figure A-1.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9043/ibm/console/logon.jsp
Go Links
Integrated Solutions Console
IBM.
Login image
Done
Local intranet

Welcome, enter your information.

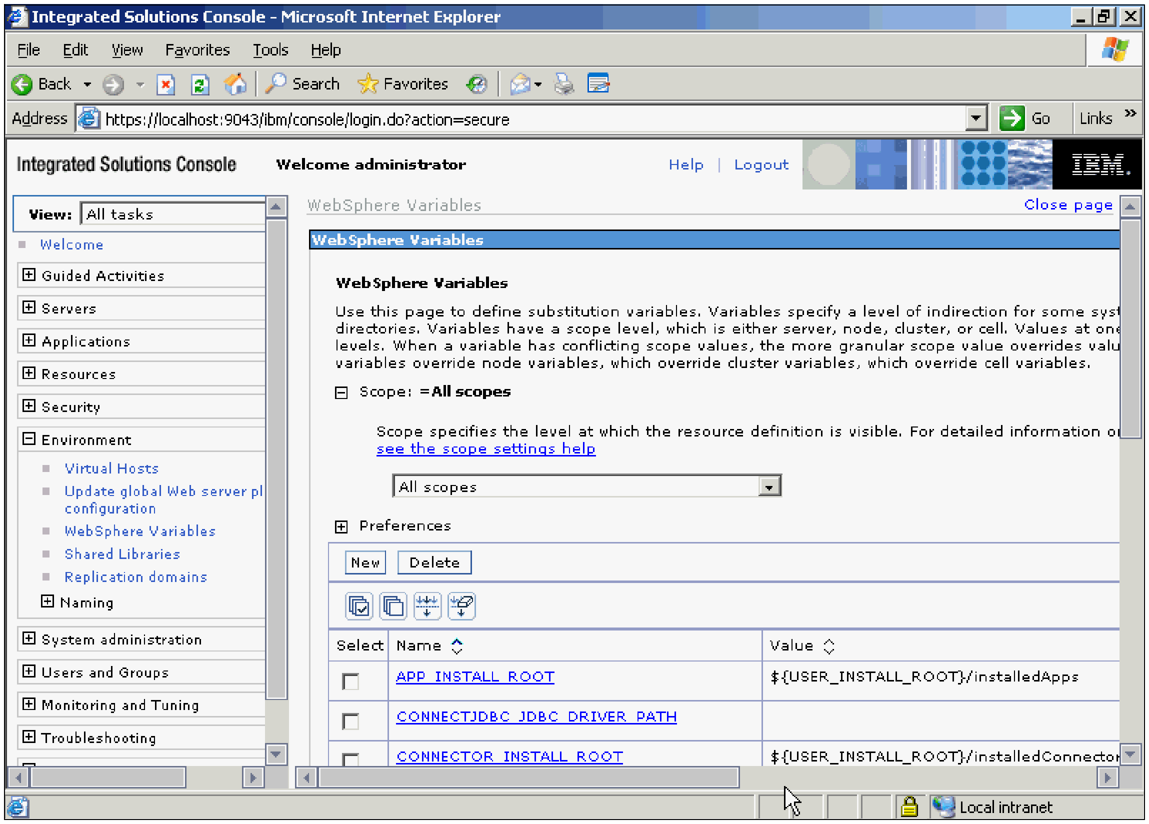
User ID: administrator

Password: •••••••

Log in

Figure A-1 Integrated Solutions Console

2. Select **Environment** ∅ **WebSphere Variables** to open the WebSphere Variables window, as shown in Figure A-2.



WebSphere Variables

Use this page to define substitution variables. Variables specify a level of indirection for some syst directories. Variables have a scope level, which is either server, node, cluster, or cell. Values at one levels. When a variable has conflicting scope values, the more granular scope value overrides valu variables override node variables, which override cluster variables, which override cell variables.

Scope: =**All scopes**

Scope specifies the level at which the resource definition is visible. For detailed information o see the scope settings help

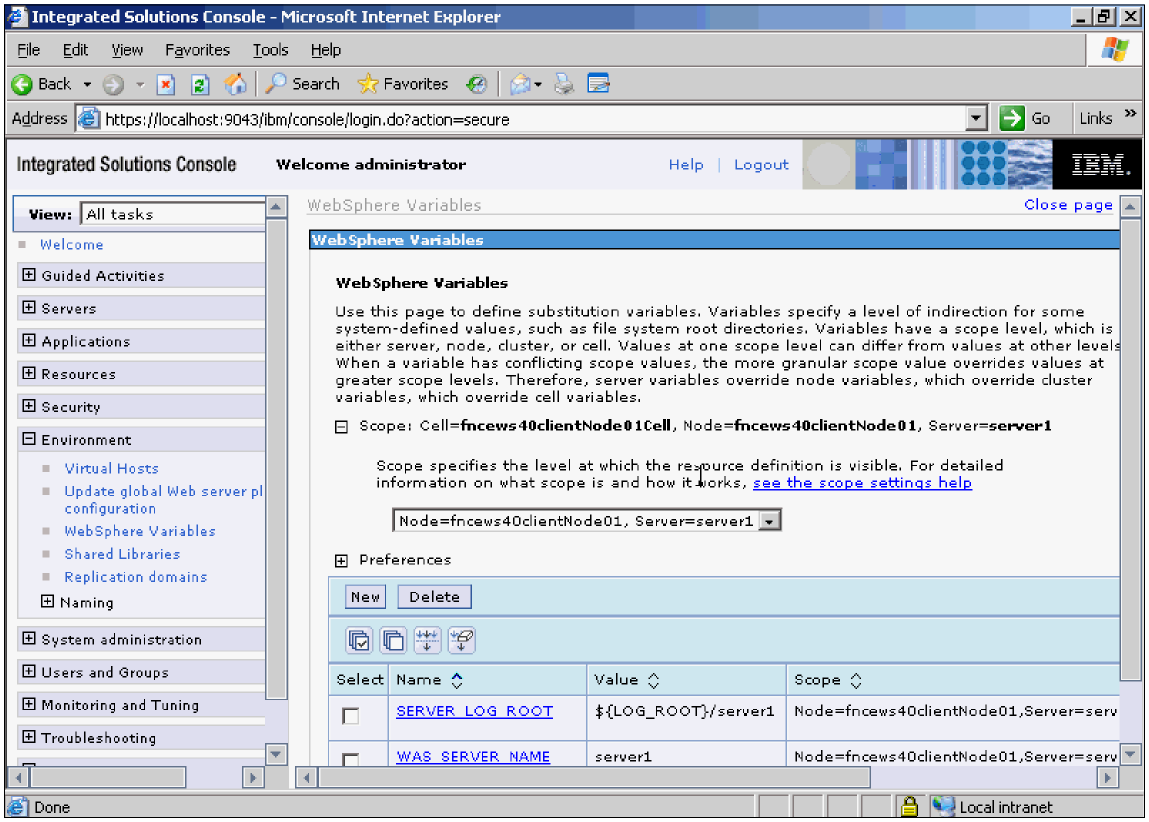
All scopes

Preferences

Select	Name	Value
☐	APP_INSTALL_ROOT	${USER_INSTALL_ROOT}/installedApps
☐	CONNECTJDBC_JDBC_DRIVER_PATH	
☐	CONNECTOR_INSTALL_ROOT	${USER_INSTALL_ROOT}/installedConnector

Figure A-2 WebSphere Variables window

3. Select a scope and click **New** to add an environment variable that specifies the installation directory for the Web services security management component of Tivoli Federated Identity Manager, as shown in Figure A-3.



WebSphere Variables

Use this page to define substitution variables. Variables specify a level of indirection for some system-defined values, such as file system root directories. Variables have a scope level, which is either server, node, cluster, or cell. Values at one scope level can differ from values at other levels. When a variable has conflicting scope values, the more granular scope value overrides values at greater scope levels. Therefore, server variables override node variables, which override cluster variables, which override cell variables.

Scope: Cell=**fncews40clientNode01Cell**, Node=**fncews40clientNode01**, Server=**server1**

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

Node=fncews40clientNode01, Server=server1

Select	Name	Value	Scope
☐	SERVER_LOG_ROOT	${LOG_ROOT}/server1	Node=fncews40clientNode01,Server=serv
☐	WAS_SERVER_NAME	server1	Node=fncews40clientNode01,Server=serv

Figure A-3 WebSphere Variables window with scope

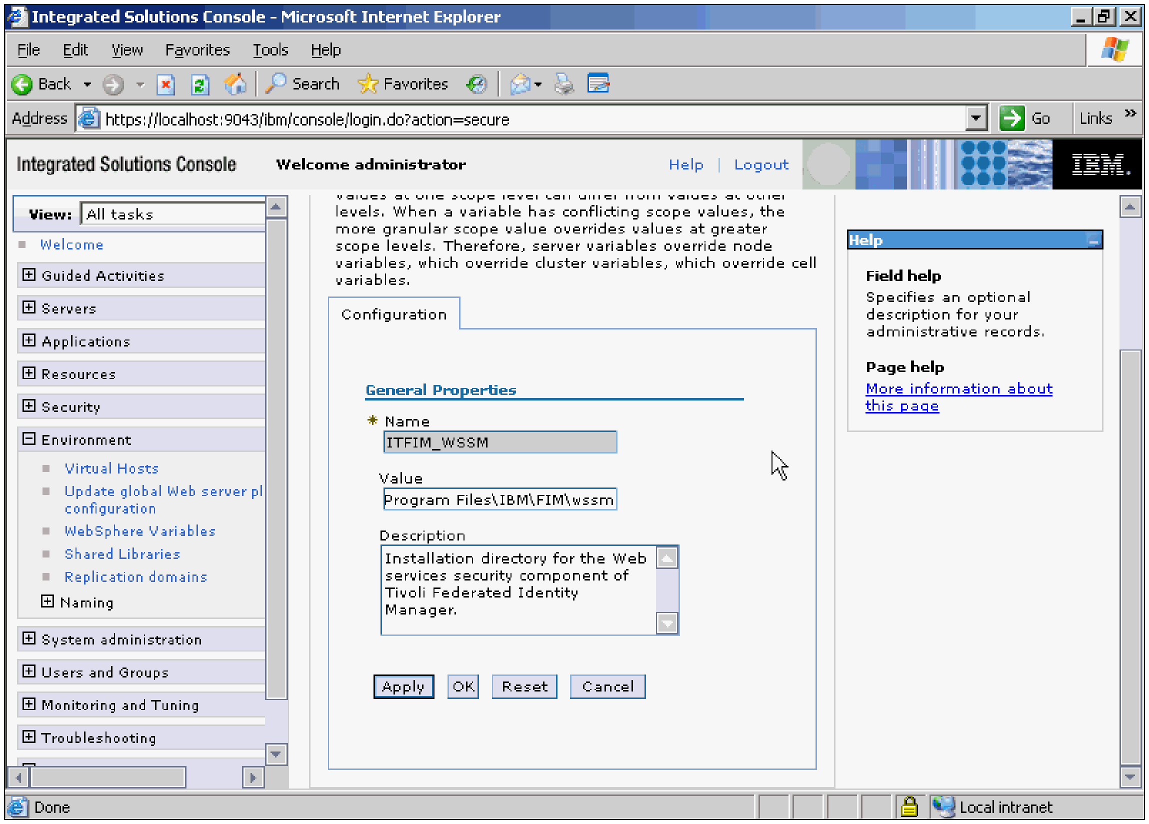
4. Enter the following information to define a variable called ITFIM_WSSM, as shown in Figure A-4.

   a. In the Name field, enter ITFIM_WSSM.

   b. In the Value field, enter the name of the directory where the Web services security management component is installed. The default for Windows is `C:\Program Files\IBM\FIM\wssm`.

   c. In the Description field, enter a description of the contents or purpose of the variable for future reference, for example, “Installation directory for the Web services security component of Tivoli Federated Identity Manager”.

   d. Click **OK** to add the variable.



values at one scope level can differ from values at other levels. When a variable has conflicting scope values, the more granular scope value overrides values at greater scope levels. Therefore, server variables override node variables, which override cluster variables, which override cell variables.

* Name
ITFIM_WSSM

Value
Program Files\IBM\FIM\wssm

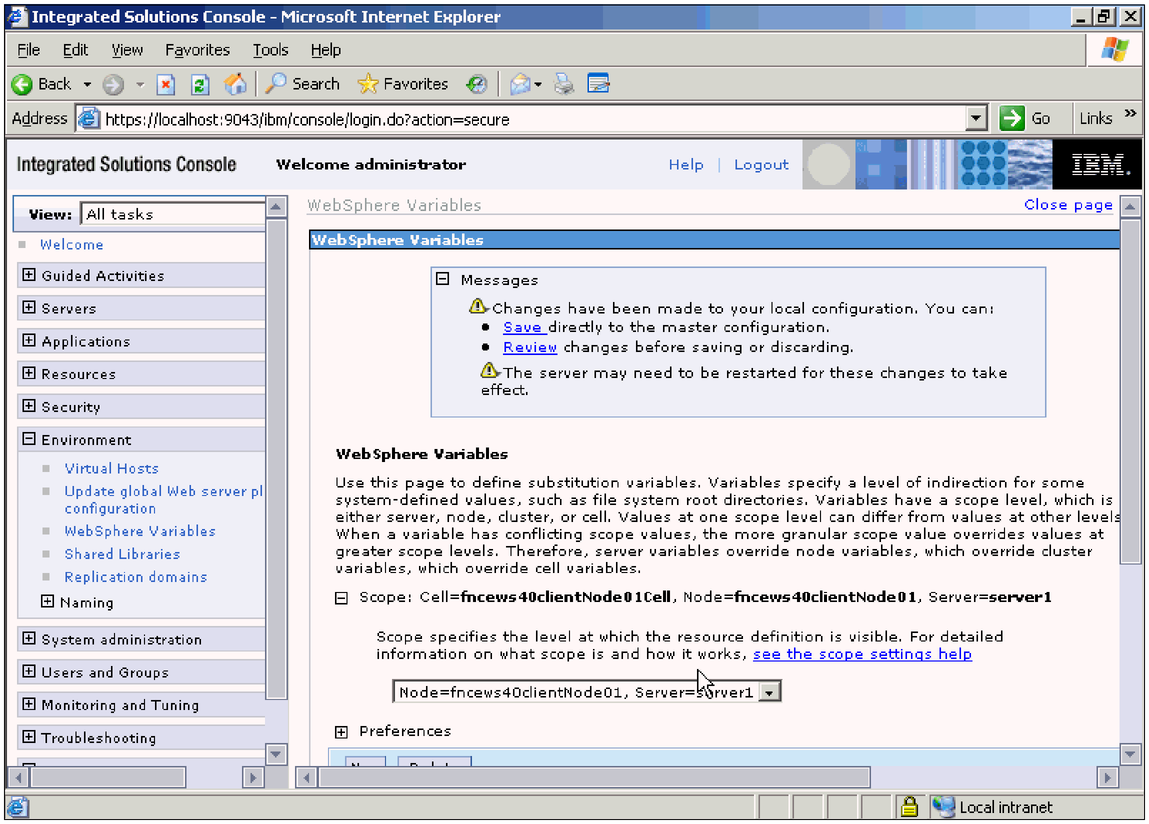
Description
Installation directory for the Web services security component of Tivoli Federated Identity Manager.

Field help
Specifies an optional description for your administrative records.

Page help

Figure A-4 ITFIM_WSSM variable configuration window

5. In the Messages pane at the top of the WebSphere Variables window, click **Save** to commit your changes, as shown in Figure A-5 on page 233.



Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

WebSphere Variables

Use this page to define substitution variables. Variables specify a level of indirection for some system-defined values, such as file system root directories. Variables have a scope level, which is either server, node, cluster, or cell. Values at one scope level can differ from values at other levels When a variable has conflicting scope values, the more granular scope value overrides values at greater scope levels. Therefore, server variables override node variables, which override cluster variables, which override cell variables.

Scope: Cell=**fncews40clientNode01Cell**, Node=**fncews40clientNode01**, Server=**server1**

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

Node=fncews40clientNode01, Server=server1

Figure A-5 Save configuration window

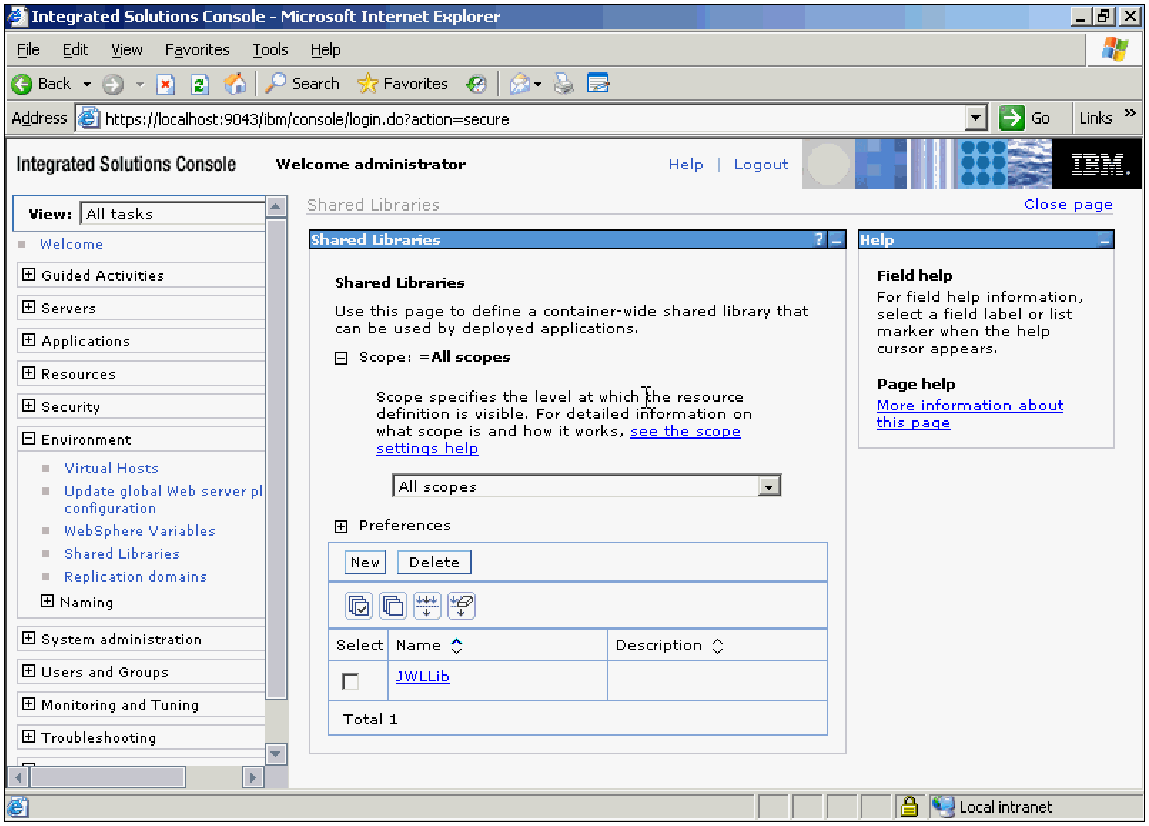
We have completed the WebSphere variable configuration. We now need to configure a WebSphere shared library.

Configuring the WSSM shared library

In order for the application to use Web services security management, we need to define a shared library in WebSphere Application Server. To do this, perform the following steps:

1. Start the WebSphere Application Server administrative console and log in, if necessary.

2. Select **Environment ∅ Shared Libraries** to open the Shared Libraries window, as shown in Figure A-6.



Shared Libraries

Use this page to define a container-wide shared library that can be used by deployed applications.

Scope: =**All scopes**

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

All scopes

Preferences

New Delete

Select	Name	Description
☐	JWLLib	

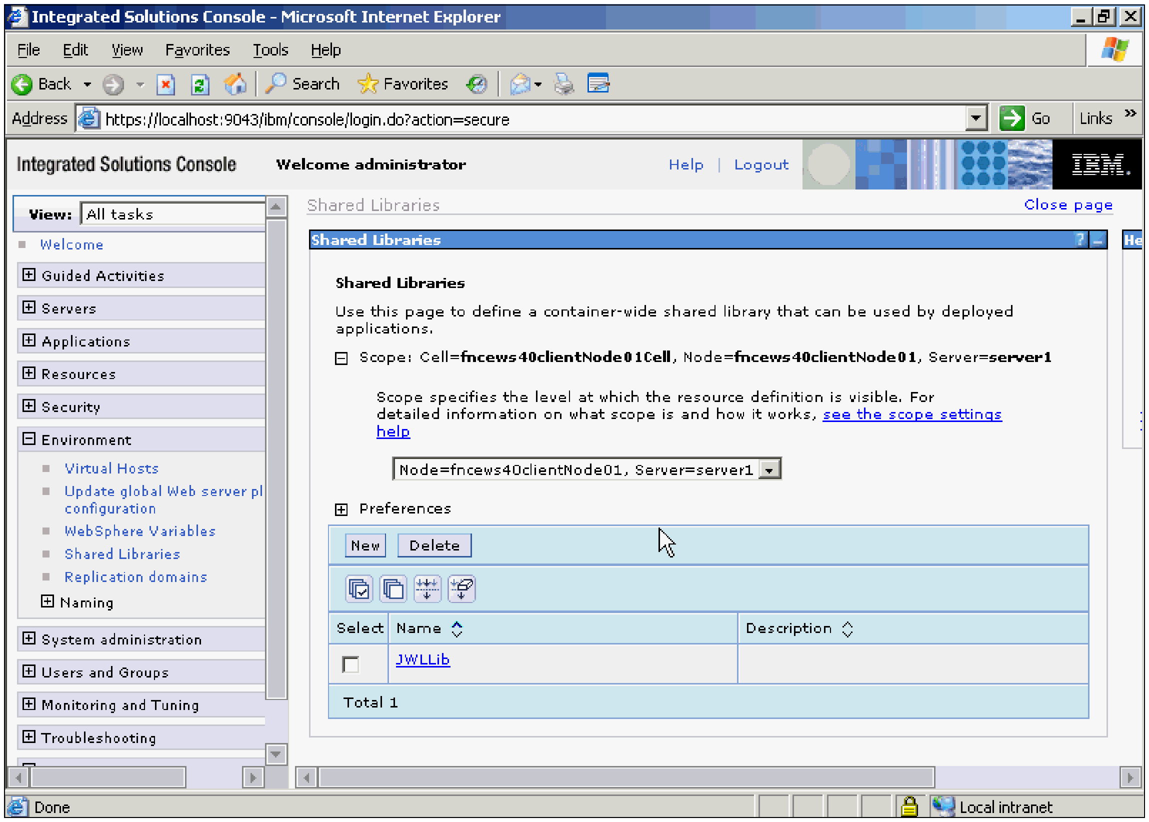
Total 1

Field help
For field help information, select a field label or list marker when the help cursor appears.

Page help
More information about this page

Figure A-6 Shared library window

3. Select a scope and click **New** to add an environment variable that specifies the installation directory for the Web services security management component of Tivoli Federated Identity Manager, as shown in Figure A-7.


Integrated Solutions Console - Microsoft Internet Explorer

Address https://localhost:9043/ibm/console/login.do?action=secure

Integrated Solutions Console Welcome administrator Help | Logout

View: All tasks

Shared Libraries Close page

Shared Libraries

Use this page to define a container-wide shared library that can be used by deployed applications.

Scope: Cell=**fncews40clientNode01Cell**, Node=**fncews40clientNode01**, Server=**server1**

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

Node=fncews40clientNode01, Server=server1

Preferences

New Delete

Select	Name	Description
☐	JWLLib	
Total 1		

Figure A-7 Shared library window with scope

4. Enter the following information to define the necessary shared library, as shown in Figure A-8 on page 236.

   a. In the Name field, enter ITFIM_WSSM. This name is needed later when configuring the class loader.

   b. In the Description field, enter a description of the contents or purpose of the shared library for future reference, for example, “Shared libraries needed by the Web services security management component of Tivoli Federated Identity Manager”.

   c. In the Classpath field, enter the following information:

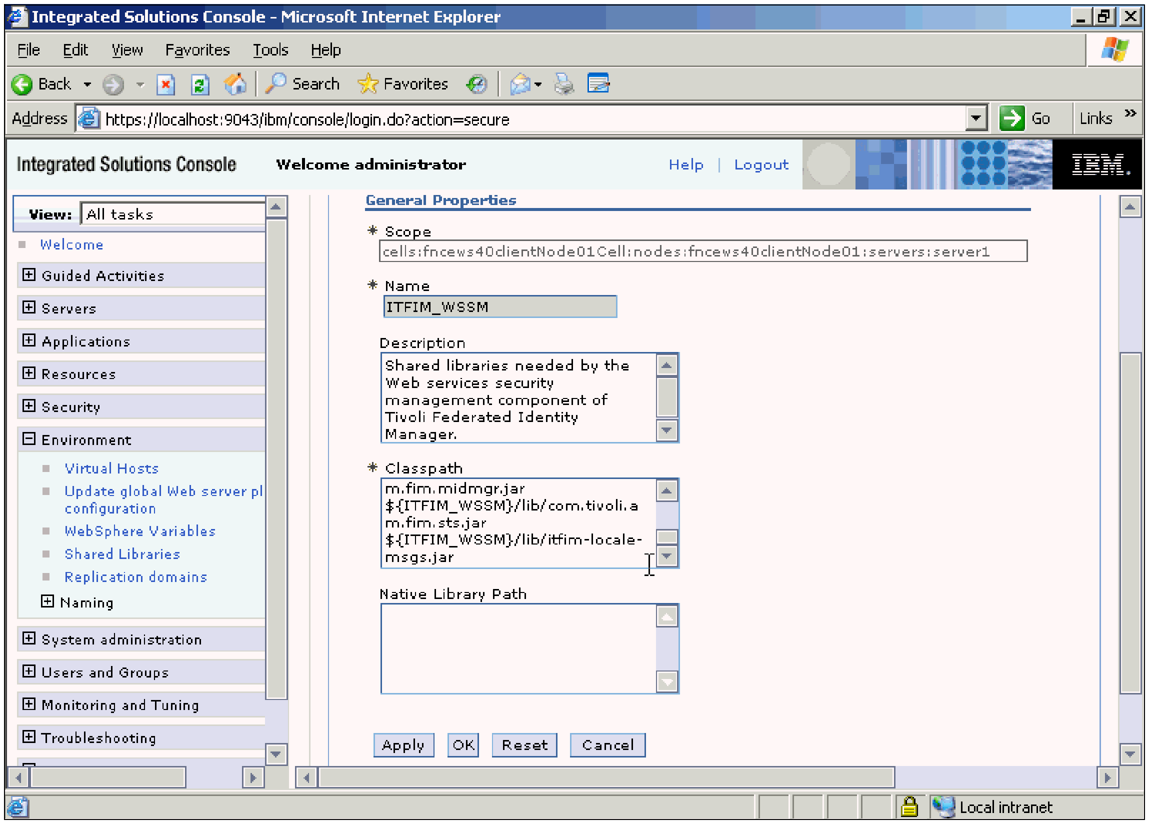
```
${ITFIM_WSSM}/etc ${ITFIM_WSSM}/lib/com.tivoli.am.fim.wssm.jar
${ITFIM_WSSM}/lib/com.tivoli.am.fim.common.jar
```

```
${ITFIM_WSSM}/lib/com.tivoli.am.fim.management.jar
${ITFIM_WSSM}/lib/com.tivoli.am.fim.war.sts.stubs.client.jar
${ITFIM_WSSM}/lib/com.tivoli.am.fim.midmgr.jar
${ITFIM_WSSM}/lib/com.tivoli.am.fim.sts.jar
${ITFIM_WSSM}/lib/itfim-locale-msgs.jar
```

Note: This list can also be found in the `C:\Program Files\IBM\FIM\wssm\etc\wssm.classpath` file.

d. Leave the Native Library Path field blank.

e. Click **OK** to add the shared library.



General Properties

* Scope
cells:fncews40clientNode01Cell:nodes:fncews40clientNode01:servers:server1

* Name
ITFIM_WSSM

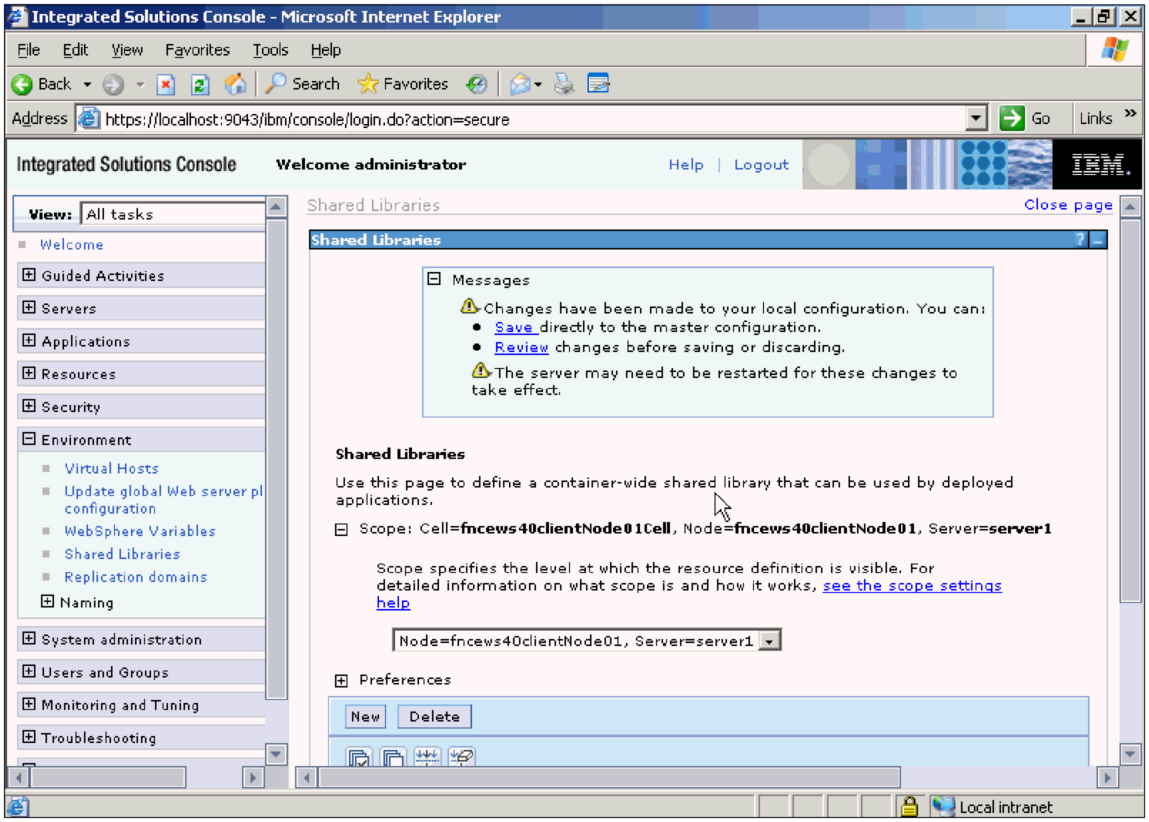
Description
Shared libraries needed by the Web services security management component of Tivoli Federated Identity Manager.

* Classpath
m.fim.midmgr.jar
${ITFIM_WSSM}/lib/com.tivoli.a
m.fim.sts.jar
${ITFIM_WSSM}/lib/itfim-locale-msgs.jar

Native Library Path

Figure A-8 ITFIM_WSSM shared library configuration window

5. In the Messages pane at the top of the Shared Libraries window, click **Save** to commit your changes, as shown in Figure A-9 on page 237.



Shared Libraries

Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Shared Libraries

Use this page to define a container-wide shared library that can be used by deployed applications.

Scope: Cell=**fncews40clientNode01Cell**, Node=**fncews40clientNode01**, Server=**server1**

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, see the scope settings help

Node=fncews40clientNode01, Server=server1

Figure A-9 Save configuration window

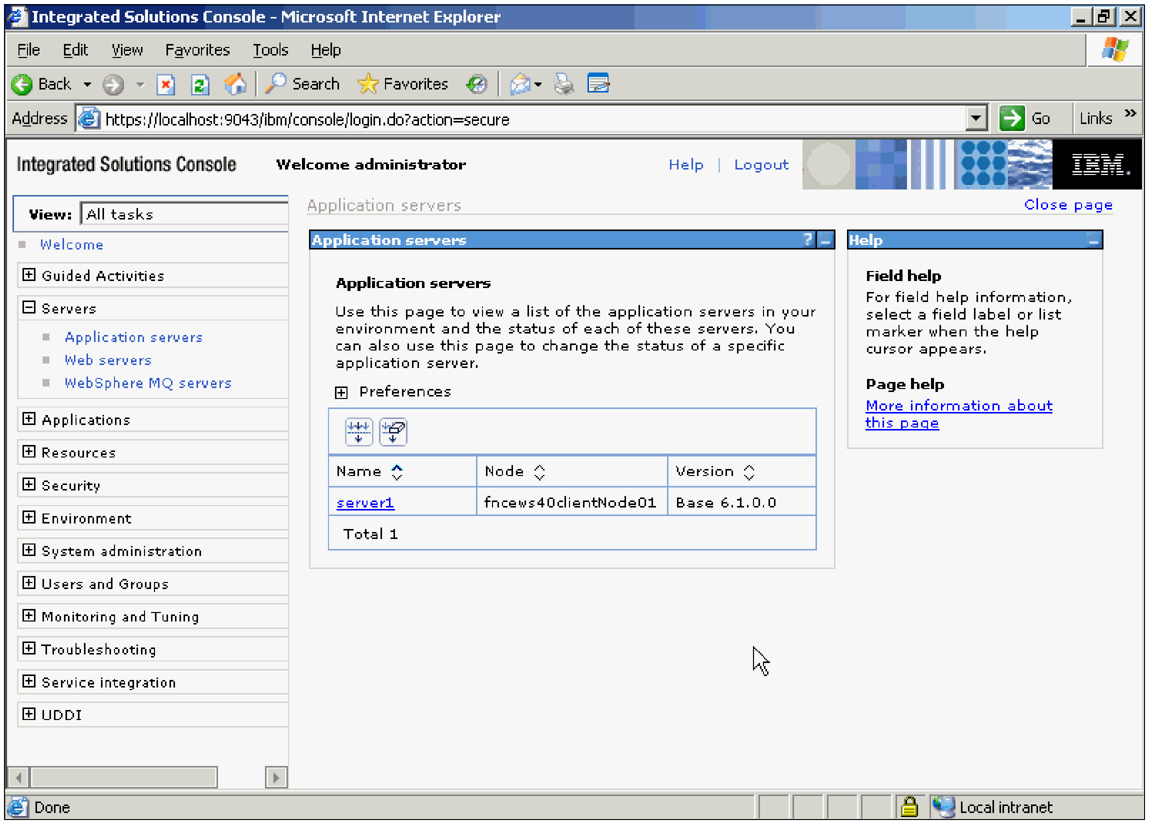
We have completed the WebSphere shared library configuration. We now need to configure the classloader for the application.

Configuring a classloader for the application server

To enable the application server to use the shared library, we must configure a class loader. To do this, perform the following steps:

1. Start the WebSphere Application Server administrative console and log in, if necessary.

2. Select **Servers** ∅ **Application Servers** and select the server associated with your application, such as *server1*, as shown in Figure A-10.



Application servers

Use this page to view a list of the application servers in your environment and the status of each of these servers. You can also use this page to change the status of a specific application server.

⊞ Preferences

Name	Node	Version
server1	fncews40clientNode01	Base 6.1.0.0
Total 1		

Field help
For field help information, select a field label or list marker when the help cursor appears.

Page help
More information about this page

Figure A-10 Application servers window

3. In the Server Infrastructure pane, expand the **Java and Process Management** container and click **Class loader**, as shown in Figure A-11.

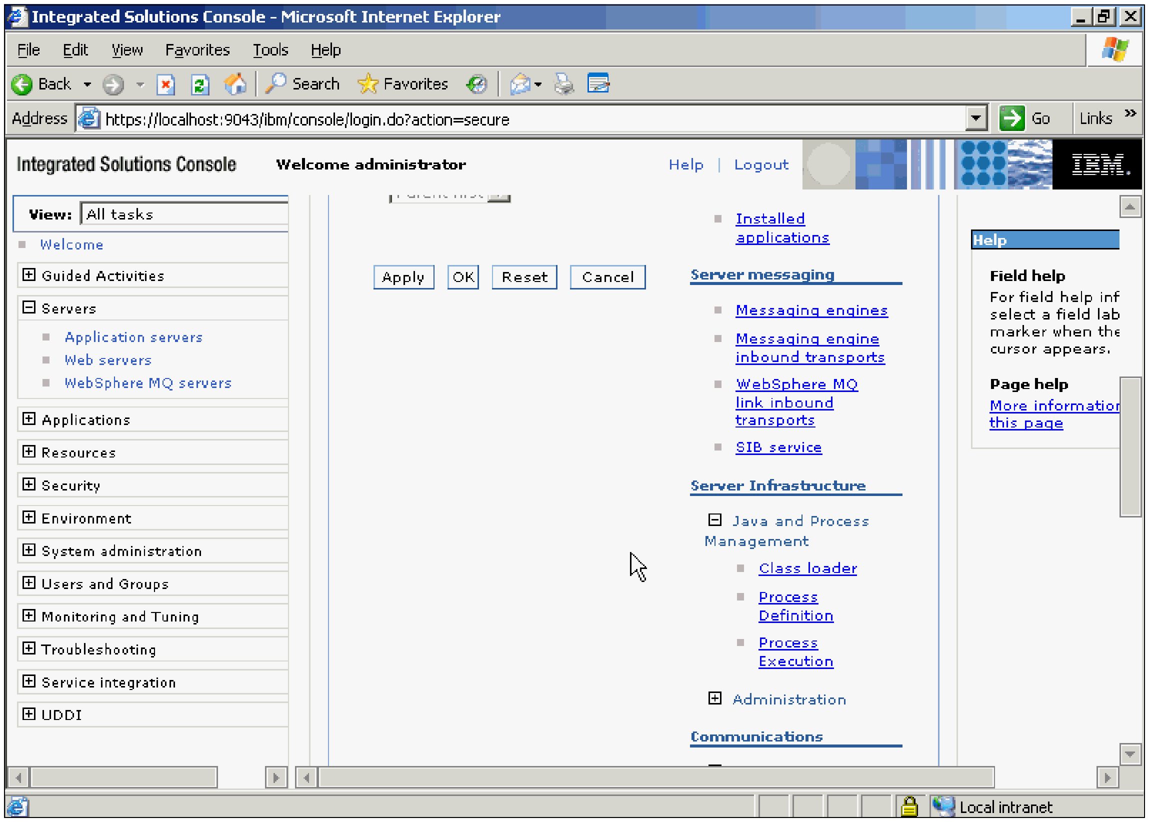
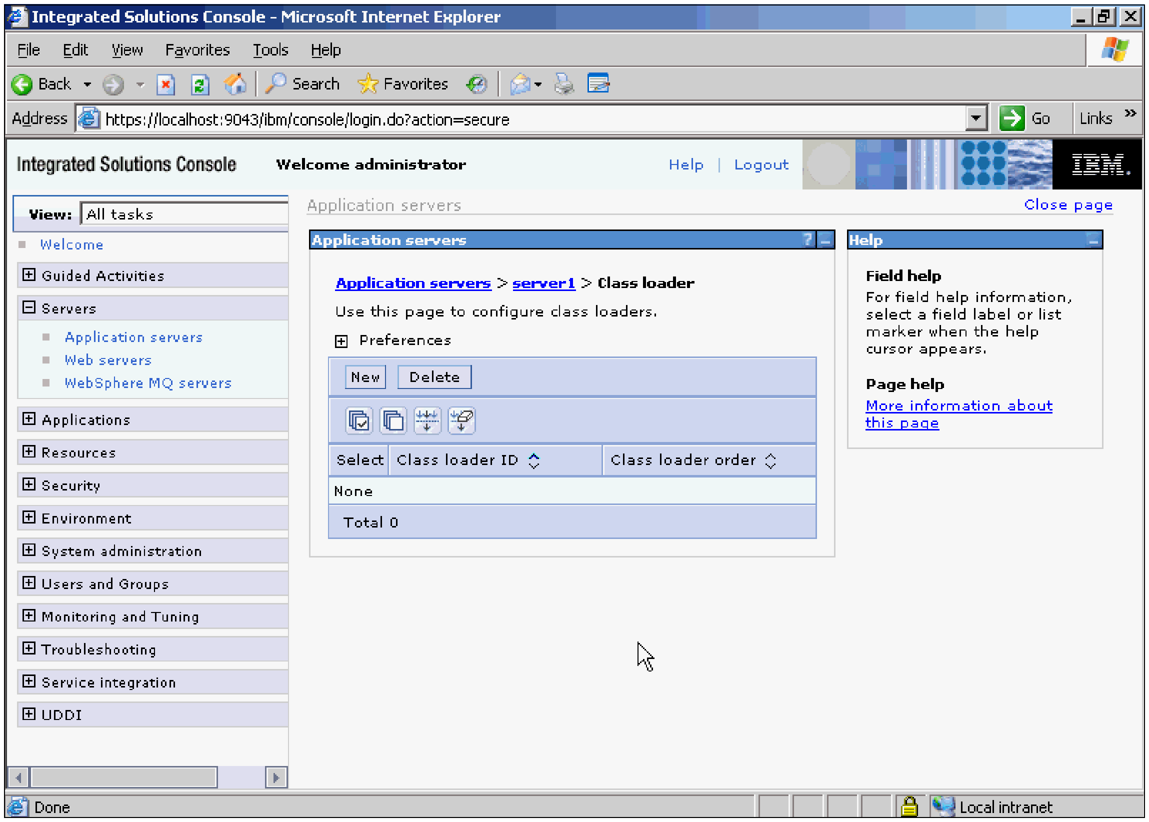


Figure A-11 Application class loader window

4. Click **New**, as shown in Figure A-12.



Application servers > server1 > Class loader

Use this page to configure class loaders.

Preferences

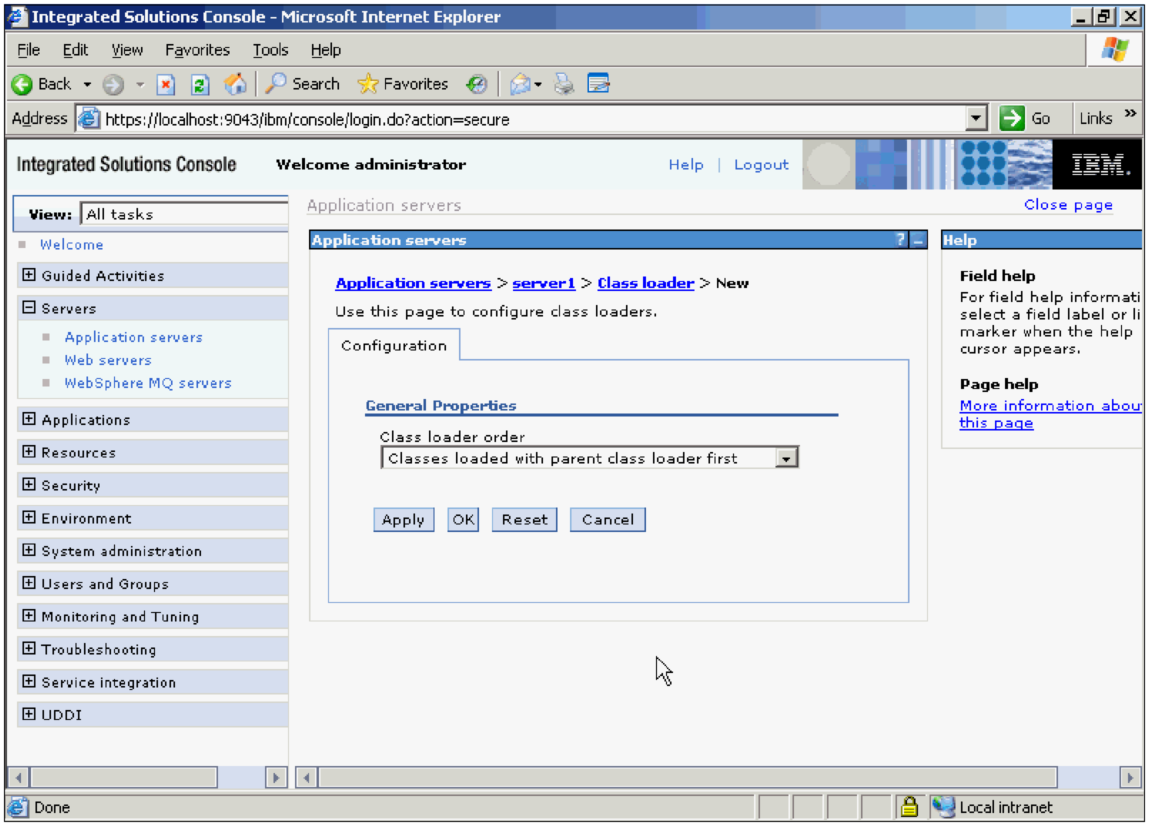
Select	Class loader ID	Class loader order
None		
Total 0		

Field help
For field help information, select a field label or list marker when the help cursor appears.

Page help
More information about this page

Figure A-12 Configure new class loader window

5. Do not make any changes; click **Apply**, as shown in Figure A-13.



Application servers > server1 > Class loader > New

Use this page to configure class loaders.

Configuration

General Properties

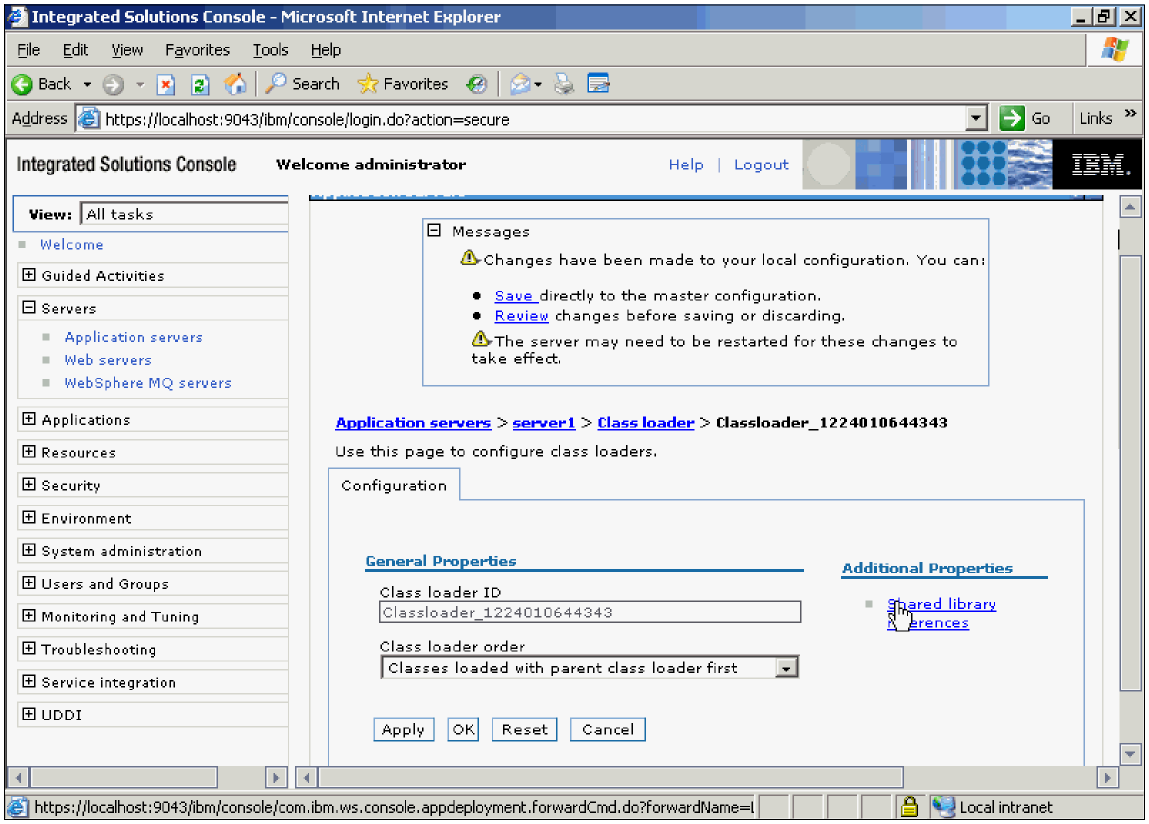
Class loader order
Classes loaded with parent class loader first

Field help
For field help informati select a field label or li marker when the help cursor appears.

Page help
More information abou this page

Figure A-13 Class loader configuration window

6. In the Additional Properties pane, click **Shared Library references**, as shown in Figure A-14.



Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Application servers > server1 > Class loader > Classloader_1224010644343

Use this page to configure class loaders.

Configuration

General Properties

Class loader ID: Classloader_1224010644343

Class loader order: Classes loaded with parent class loader first

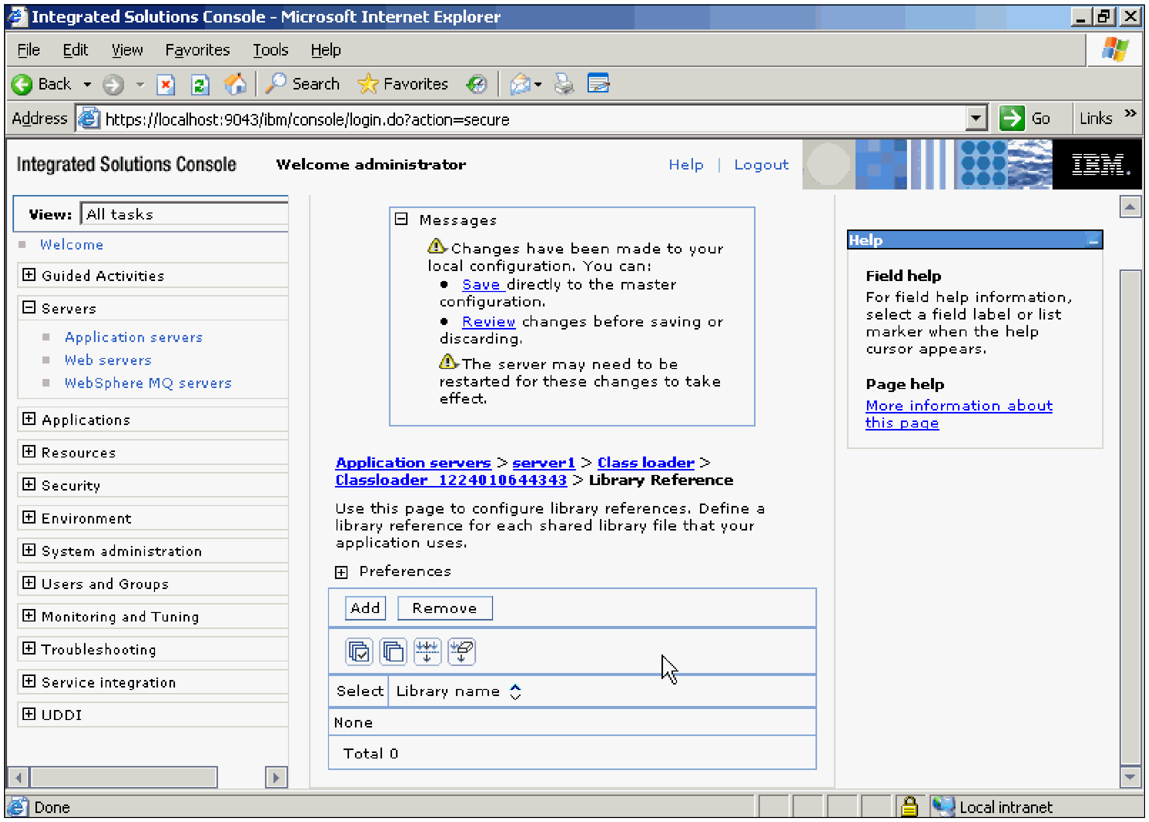
Additional Properties

- Shared library references

Apply
OK
Reset
Cancel
https://localhost:9043/ibm/console/com.ibm.ws.console.appdeployment.forwardCmd.do?forwardName=
Local intranet

Figure A-14 Shared library configuration window

7. Click **Add** to specify a shared library, as shown in Figure A-15.



Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Application servers > server1 > Class loader > Classloader_1224010644343 > Library Reference

Use this page to configure library references. Define a library reference for each shared library file that your application uses.

Preferences

Add Remove

Select	Library name
None	
Total 0	

Help

Field help

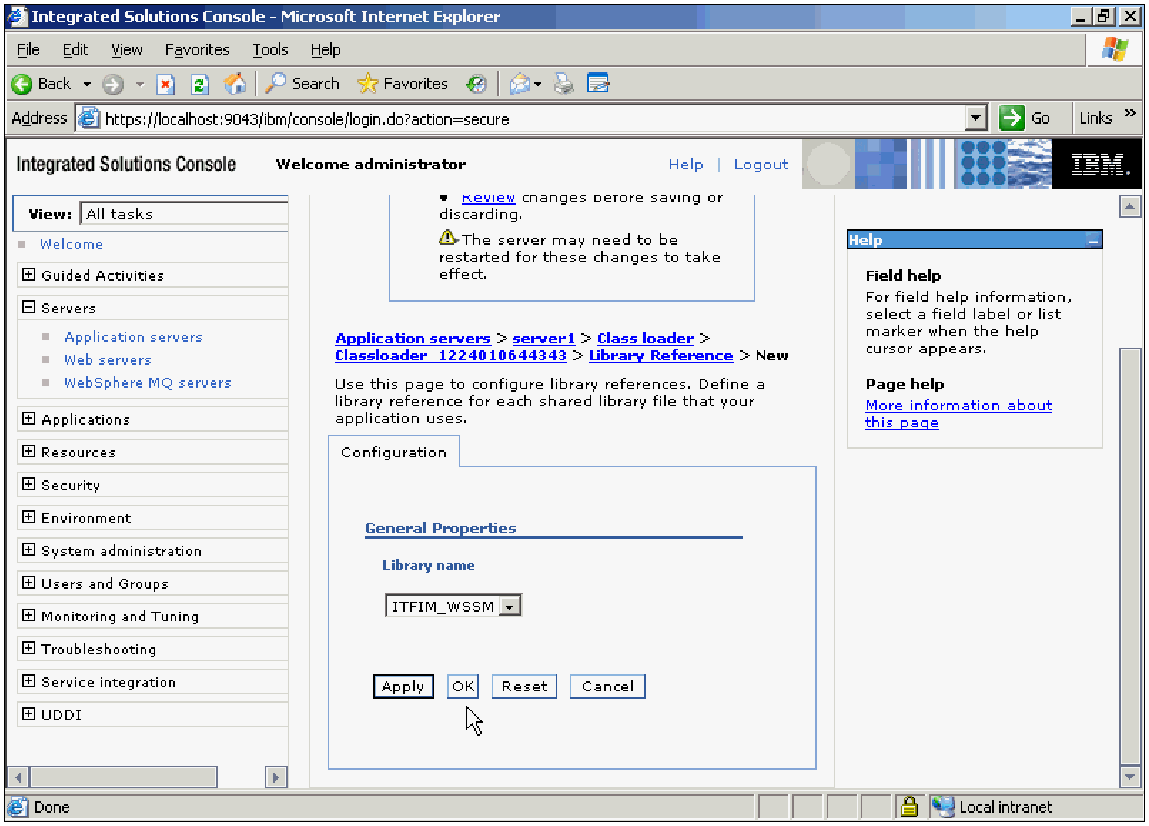
For field help information, select a field label or list marker when the help cursor appears.

Page help

More information about this page

Figure A-15 Add shared library window

8. In the Library name field, select the ITFIM_WSSM shared library previously defined and click **OK**, as shown in Figure A-16.



Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Application servers > server1 > Class loader > Classloader_1224010644343 > Library Reference > New

Use this page to configure library references. Define a library reference for each shared library file that your application uses.

Configuration

General Properties

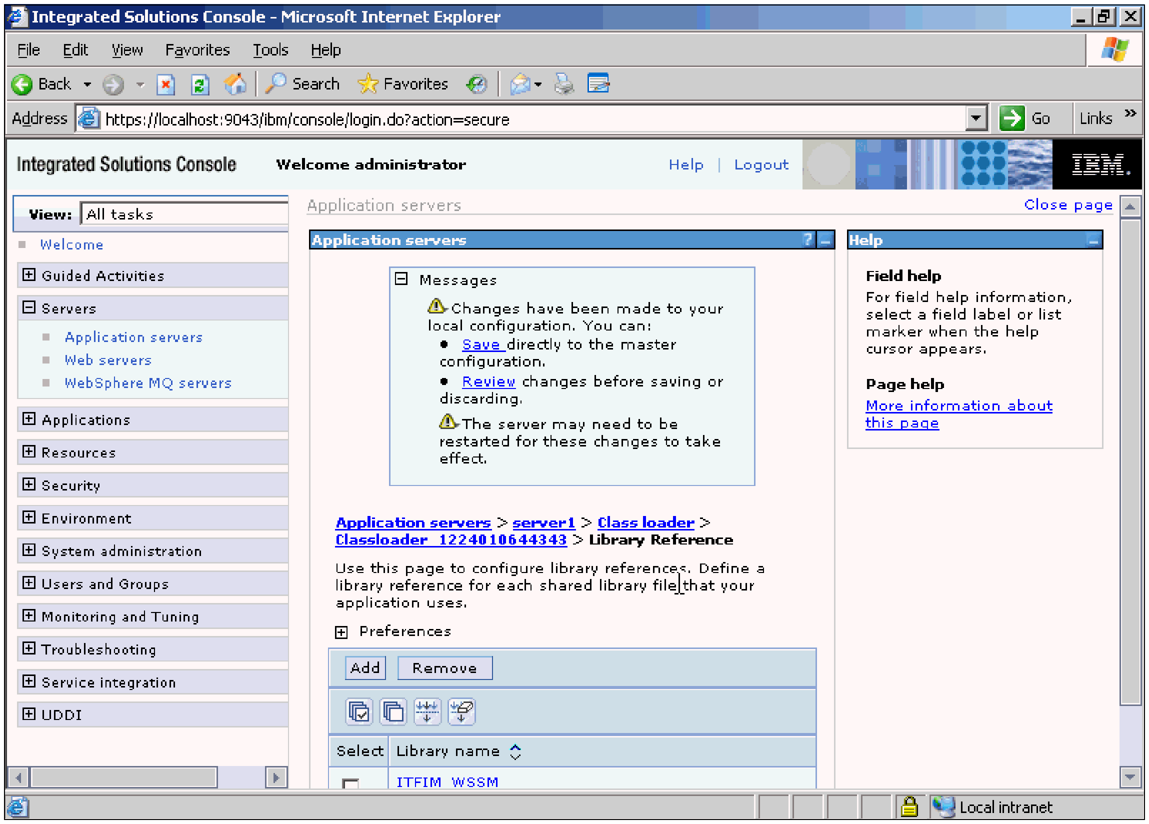
Library name: ITFIM_WSSM

Apply | OK | Reset | Cancel

For field help information, select a field label or list marker when the help cursor appears.

Figure A-16 Shared library configuration window

9. In the Messages pane at the top of the Application Servers window, click **Save** to commit your changes, as shown in Figure A-17 on page 245.


Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Application servers > server1 > Class loader > Classloader_1224010644343 > Library Reference

Use this page to configure library references. Define a library reference for each shared library file that your application uses.

Select	Library name
☐	ITFIM_WSSM

Figure A-17 Class loader save window

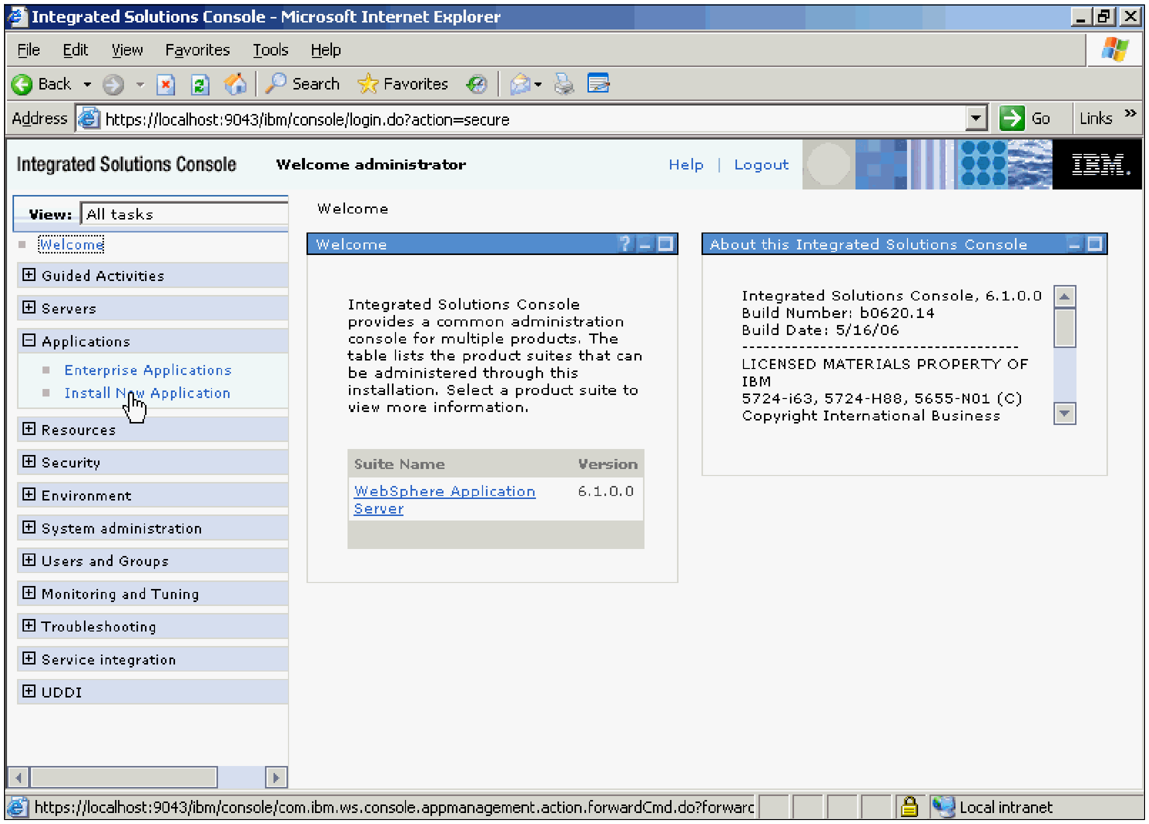
We have completed the classloader configuration. We now need to install the FileNet CEWS SAML 2.0 Client application.

Installing the sample FileNet CEWS SAML 2.0 Client application

Next, we need to install the client EAR file. We are assuming that the `FNCEWS40SAML20ClientEAR.ear` file has been copied to a temporary location on the client machine. To do this, perform the following steps:

1. Start the WebSphere Application Server administrative console and log in, if necessary.

2. Select **Applications** ∅ **Install New Application**, as shown in Figure A-18.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9043/ibm/console/login.do?action=secure
Go Links
Integrated Solutions Console
Welcome administrator
Help | Logout
View: All tasks
Welcome
Guided Activities
Servers
Applications
Enterprise Applications
Install New Application
Resources
Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Welcome
About this Integrated Solutions Console
https://localhost:9043/ibm/console/com.ibm.ws.console.appmanagement.action.forwardCmd.do?forwarc
Local intranet

Integrated Solutions Console provides a common administration console for multiple products. The table lists the product suites that can be administered through this installation. Select a product suite to view more information.

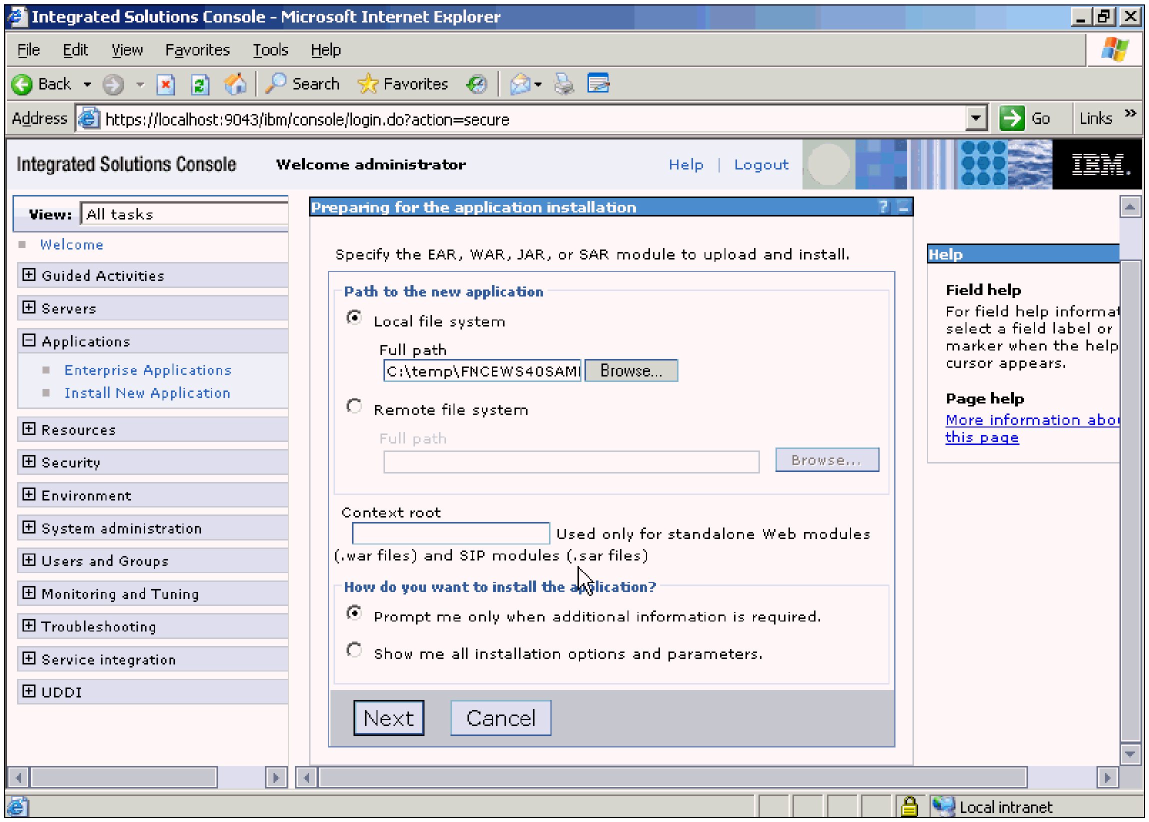
Suite Name	Version
WebSphere Application Server	6.1.0.0

Integrated Solutions Console, 6.1.0.0
Build Number: b0620.14
Build Date: 5/16/06

LICENSED MATERIALS PROPERTY OF IBM
5724-i63, 5724-H88, 5655-N01 (C)
Copyright International Business

Figure A-18 Install new application menu

3. Click **Browse** and browse to the `FNCEWS40SAML20ClientEAR.ear` file and then click **Next**, as shown in Figure A-19.


Integrated Solutions Console - Microsoft Internet Explorer

Address: https://localhost:9043/ibm/console/login.do?action=secure

Integrated Solutions Console — Welcome administrator — Help | Logout

View: All tasks

Preparing for the application installation

Specify the EAR, WAR, JAR, or SAR module to upload and install.

Path to the new application

(●) Local file system
Full path: C:\temp\FNCEWS40SAMI [Browse...]

( ) Remote file system
Full path: [Browse...]

Context root: Used only for standalone Web modules (.war files) and SIP modules (.sar files)

How do you want to install the application?

(●) Prompt me only when additional information is required.

( ) Show me all installation options and parameters.

[Next] [Cancel]

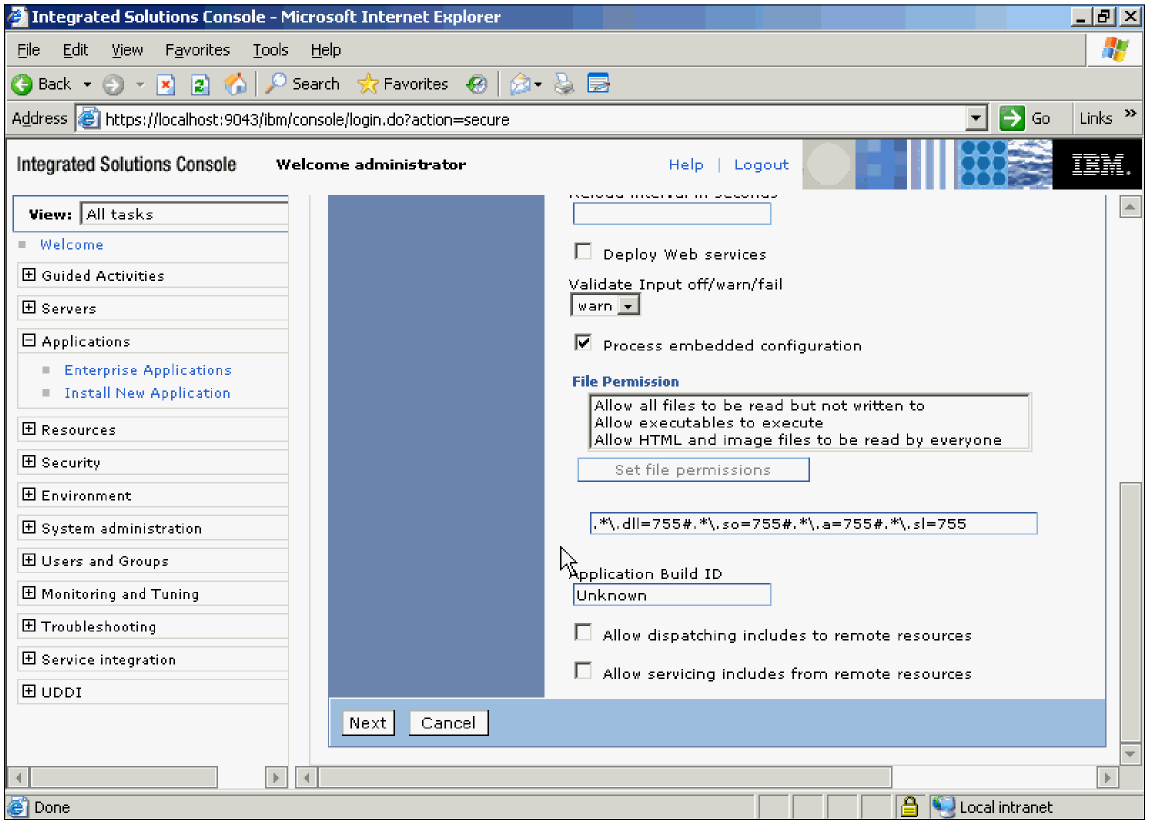
Help

Field help
For field help informat select a field label or marker when the help cursor appears.

Page help
More information abo this page

Figure A-19 Install new application window

4. Click **Next**, as shown in Figure A-20.



[ ] Deploy Web services

Validate Input off/warn/fail: warn

[✓] Process embedded configuration

File Permission

Allow all files to be read but not written to
Allow executables to execute
Allow HTML and image files to be read by everyone

Set file permissions

.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755

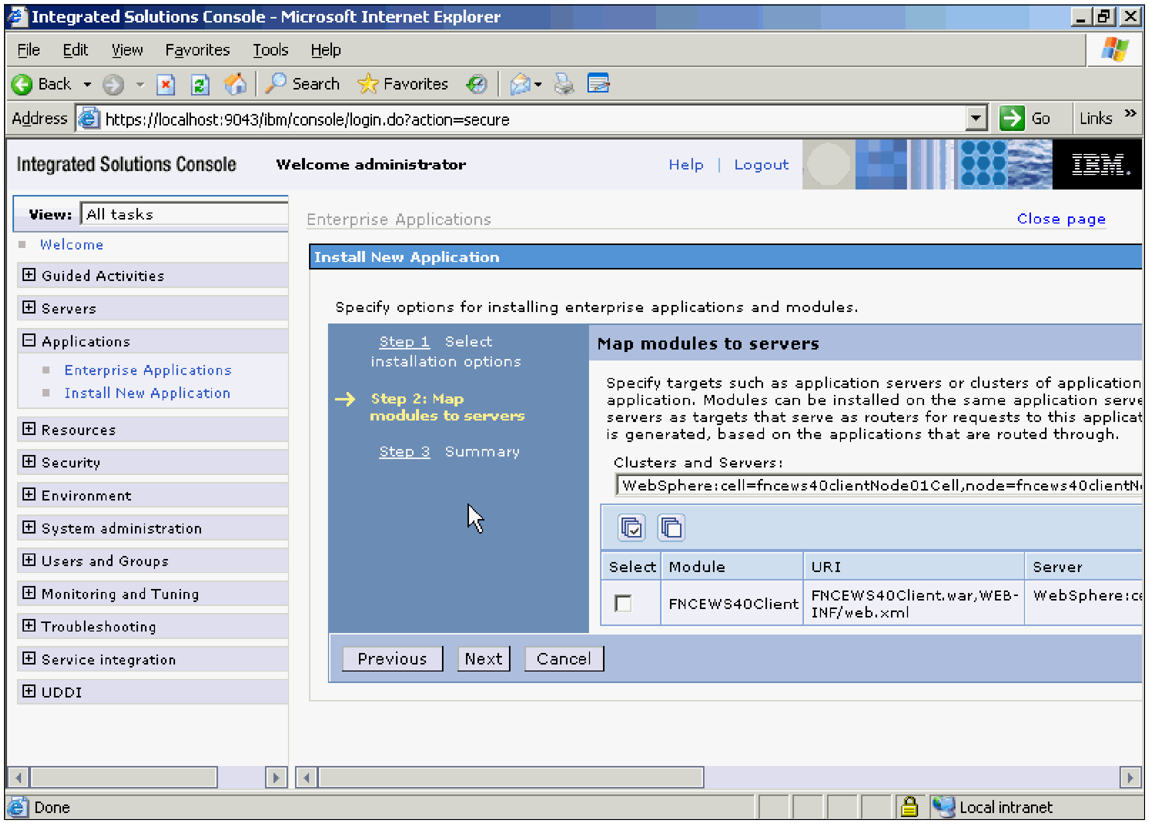
Application Build ID: Unknown

[ ] Allow dispatching includes to remote resources

[ ] Allow servicing includes from remote resources

Figure A-20 Application configuration options window

5. Click **Next**, as shown in Figure A-21.



Specify options for installing enterprise applications and modules.

Map modules to servers

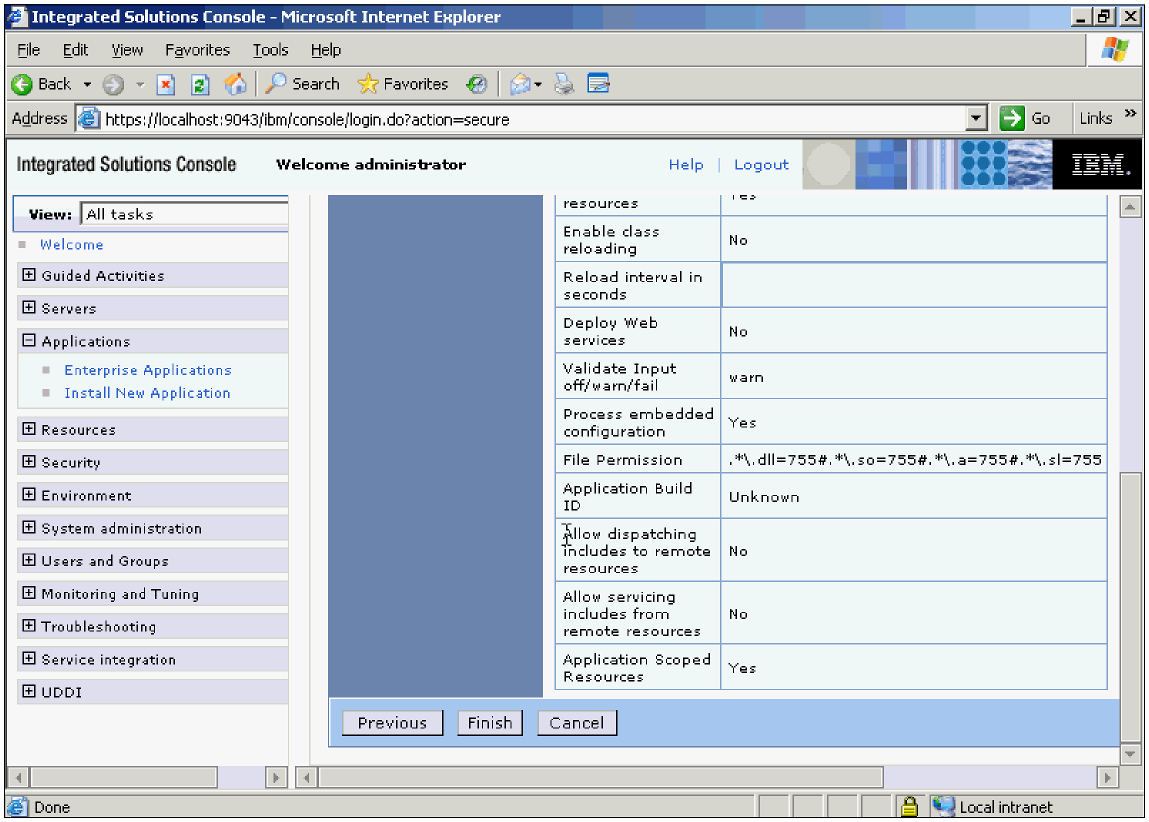
Specify targets such as application servers or clusters of application application. Modules can be installed on the same application serve servers as targets that serve as routers for requests to this applicat is generated, based on the applications that are routed through.

Clusters and Servers:
WebSphere:cell=fncews40clientNode01Cell,node=fncews40clientN

Select	Module	URI	Server
☐	FNCEWS40Client	FNCEWS40Client.war,WEB-INF/web.xml	WebSphere:ce

Figure A-21 Map module to servers window

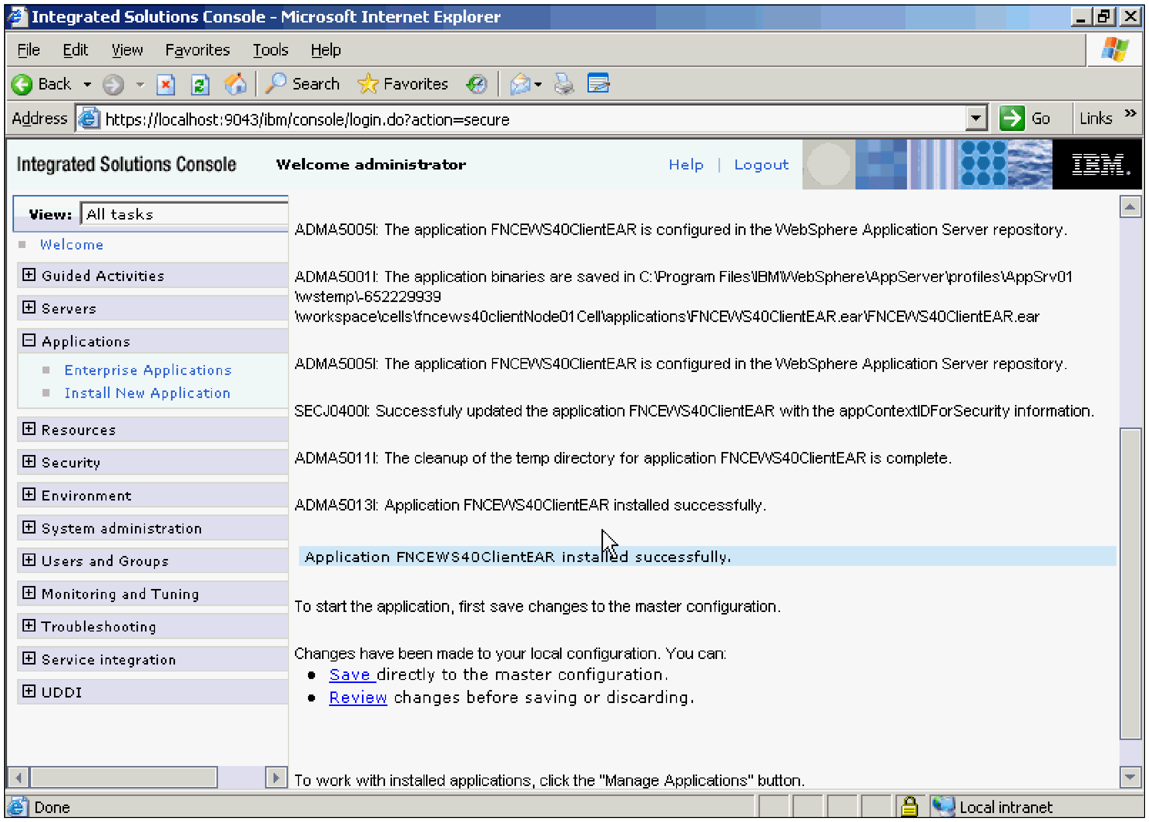
6. Click **Finish**, as shown in Figure A-22.



resources	Yes
Enable class reloading	No
Reload interval in seconds	
Deploy Web services	No
Validate Input off/warn/fail	warn
Process embedded configuration	Yes
File Permission	.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755
Application Build ID	Unknown
Allow dispatching includes to remote resources	No
Allow servicing includes from remote resources	No
Application Scoped Resources	Yes

Figure A-22 Application install summary window

7. Click **Save**, as shown in Figure A-23.



ADMA5005I: The application FNCEWS40ClientEAR is configured in the WebSphere Application Server repository.

ADMA5001I: The application binaries are saved in C:\Program Files\IBM\WebSphere\AppServer\profiles\AppSrv01\wstemp\-652229939\workspace\cells\fncews40clientNode01Cell\applications\FNCEWS40ClientEAR.ear\FNCEWS40ClientEAR.ear

ADMA5005I: The application FNCEWS40ClientEAR is configured in the WebSphere Application Server repository.

SECJ0400I: Successfuly updated the application FNCEWS40ClientEAR with the appContextIDForSecurity information.

ADMA5011I: The cleanup of the temp directory for application FNCEWS40ClientEAR is complete.

ADMA5013I: Application FNCEWS40ClientEAR installed successfully.

Application FNCEWS40ClientEAR installed successfully.

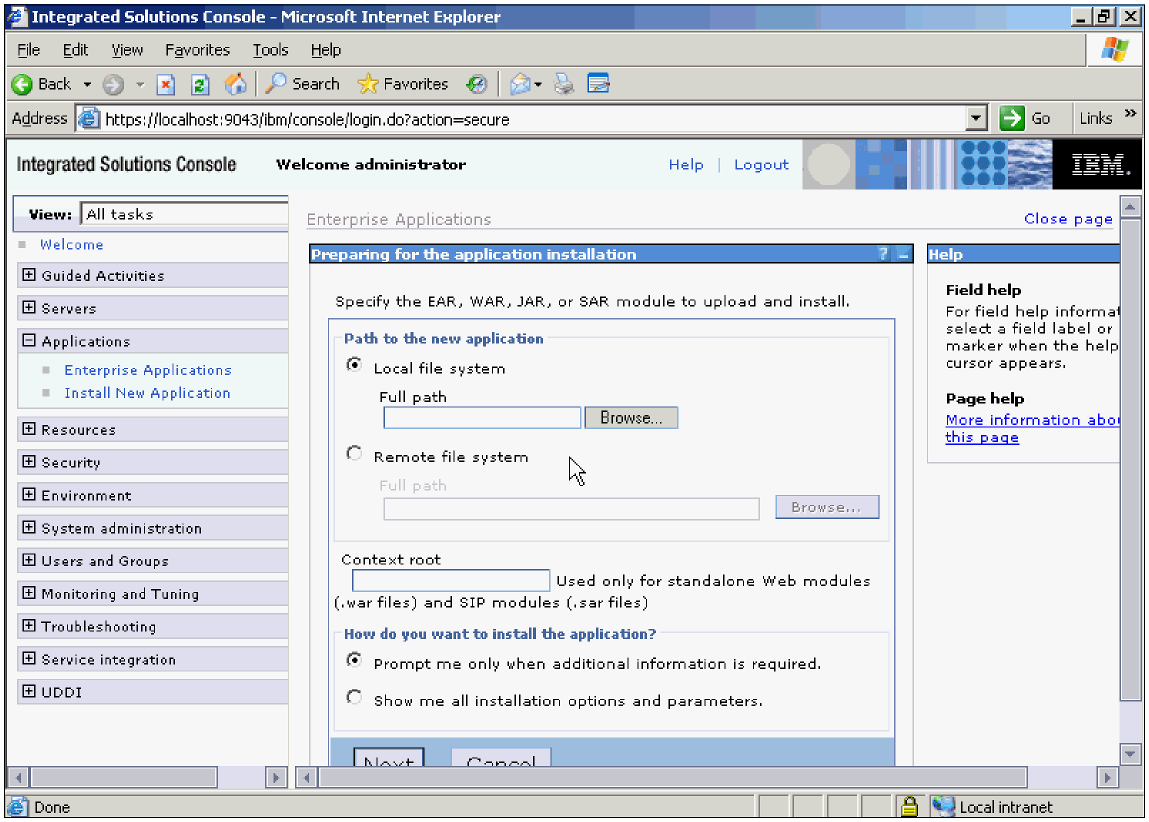
To start the application, first save changes to the master configuration.

Changes have been made to your local configuration. You can:
- Save directly to the master configuration.
- Review changes before saving or discarding.

To work with installed applications, click the "Manage Applications" button.

Figure A-23 Application successfully installed window

8. Click **Enterprise Applications**, as shown in Figure A-24.


Integrated Solutions Console - Microsoft Internet Explorer

Address https://localhost:9043/ibm/console/login.do?action=secure

Integrated Solutions Console Welcome administrator

Enterprise Applications

Preparing for the application installation

Specify the EAR, WAR, JAR, or SAR module to upload and install.

Path to the new application

[✓ Local file system]

Full path

Remote file system

Full path

Context root

Used only for standalone Web modules (.war files) and SIP modules (.sar files)

How do you want to install the application?

[✓ Prompt me only when additional information is required.]

Show me all installation options and parameters.

Help

Field help

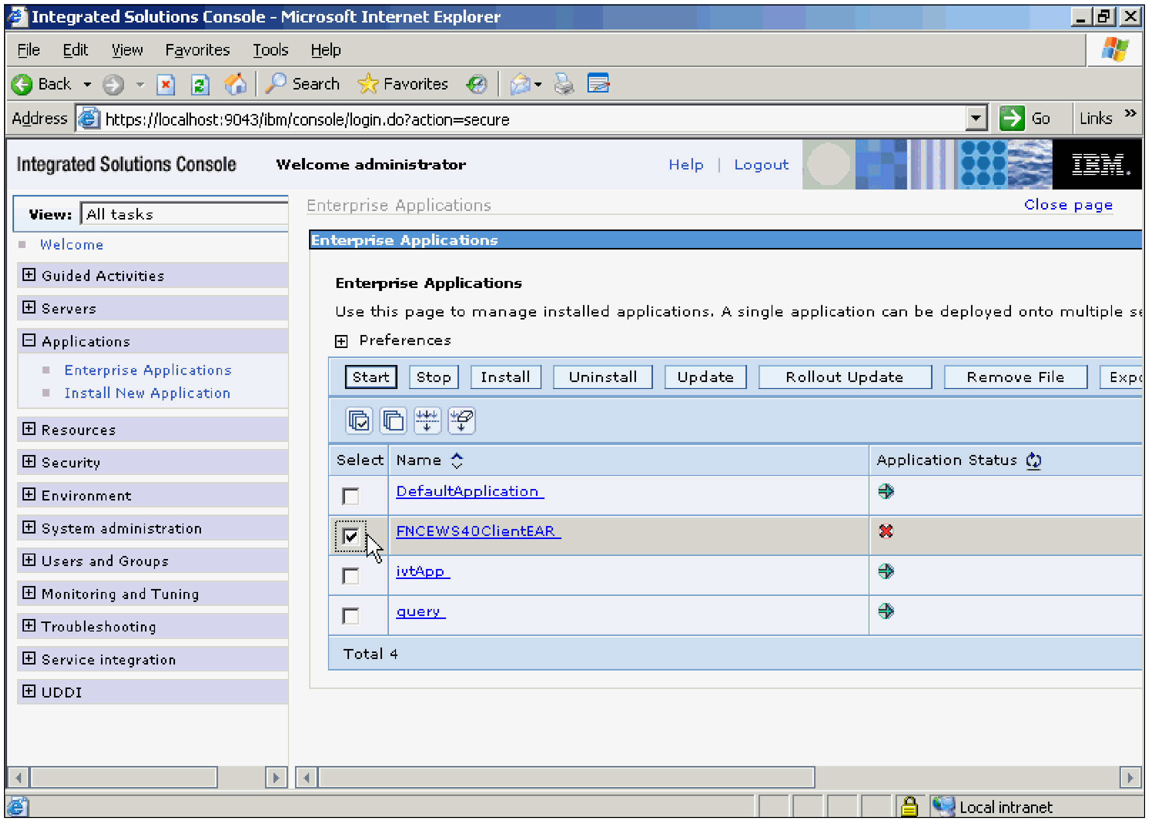
For field help informat select a field label or marker when the help cursor appears.

Page help

More information abo this page

Figure A-24 Enterprise Applications window

9. Select the FNCEWS40ClientEAR application and click **Start**, as shown in Figure A-25.



Enterprise Applications

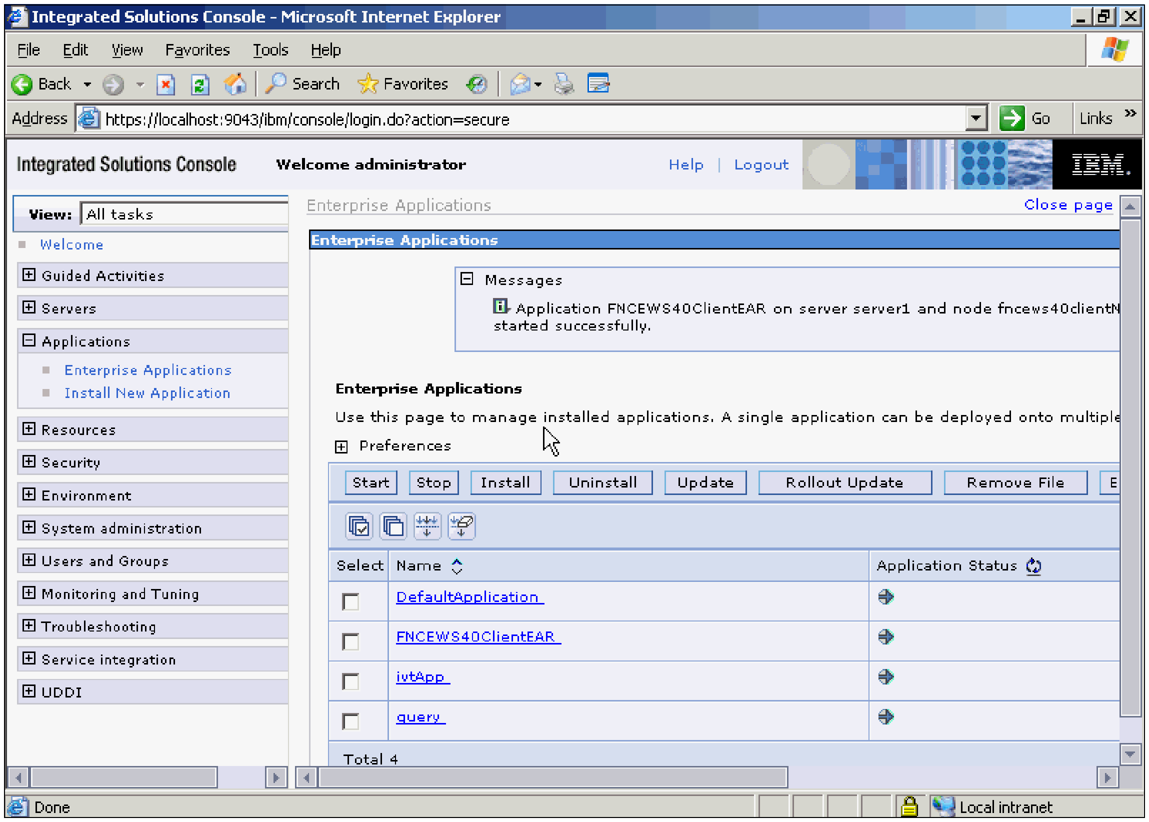
Use this page to manage installed applications. A single application can be deployed onto multiple se

Select	Name	Application Status
☐	DefaultApplication	➡
☑	FNCEWS40ClientEAR	✖
☐	ivtApp	➡
☐	query	➡

Total 4

Figure A-25 Enterprise applications window

10.The application has started successfully, as shown in Figure A-26.



Application FNCEWS40ClientEAR on server server1 and node fncews40clientN started successfully.

Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple

Preferences
Start Stop Install Uninstall Update Rollout Update Remove File E

Select	Name	Application Status
☐	DefaultApplication	➔
☐	FNCEWS40ClientEAR	➔
☐	ivtApp	➔
☐	query	➔

Total 4

Figure A-26 Application successfully started window

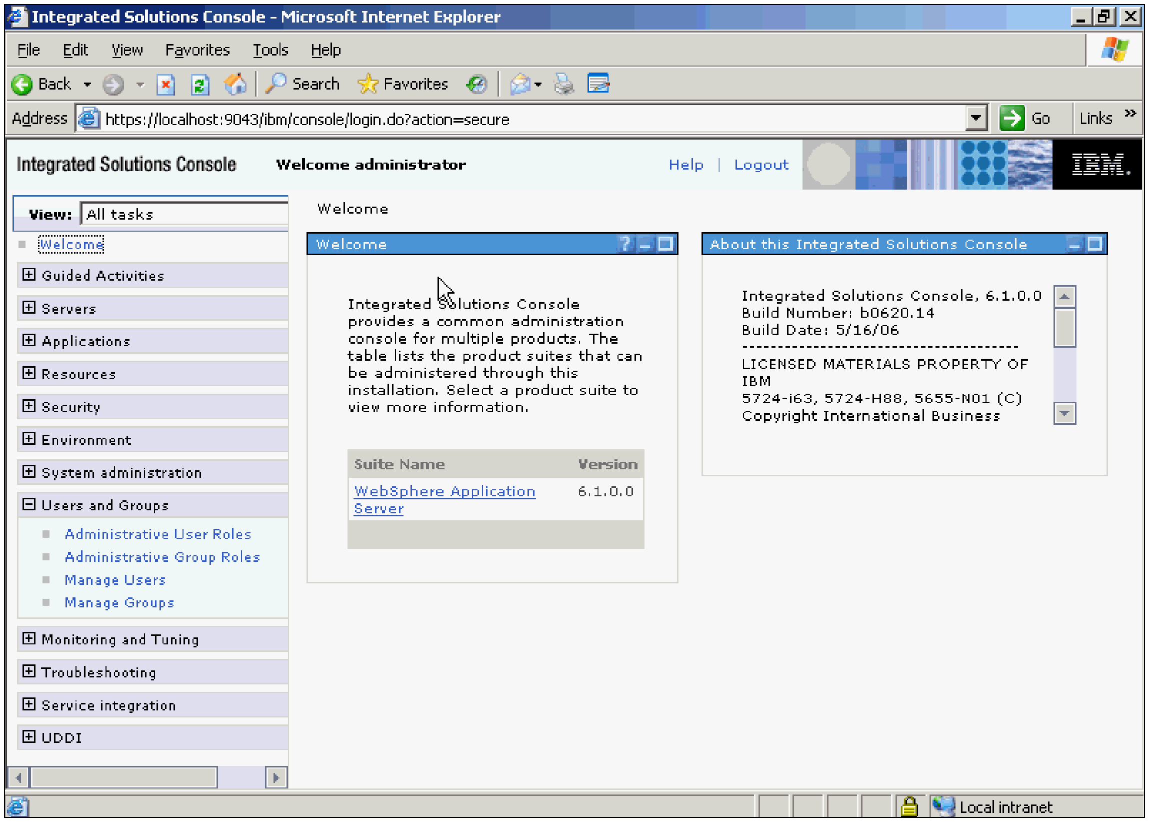
We have completed the Web services client installation. Next, we need to create a user and map them to a role that corresponds with the FNCEWS40ClientEAR application.

Setting up a FileNet CEWS SAML 2.0 Client user and mapping a role

The sample FileNet CEWS SAML 2.0 Client requires that a user logs in to access the client application. This means that the user must be created within the WebSphere Application Server directory and the user is added to the AllAuthenticated role defined by the client EAR. In this case, we add all authenticated users to the AllAuthenticated role.

Let us set up a user called jbloggs@qisa.com.au with the password filenet. This user matches the user expected in the Federated Identity Manager mapping file outlined in 6.3.5, “E-mail to user name and password mapping module” on page 190. To do this task, perform the following steps:

1. Start the WebSphere Application Server administrative console and log in, if necessary.
2. Select **Users and Groups** ∅ **Manage Users**, as shown in Figure A-27.



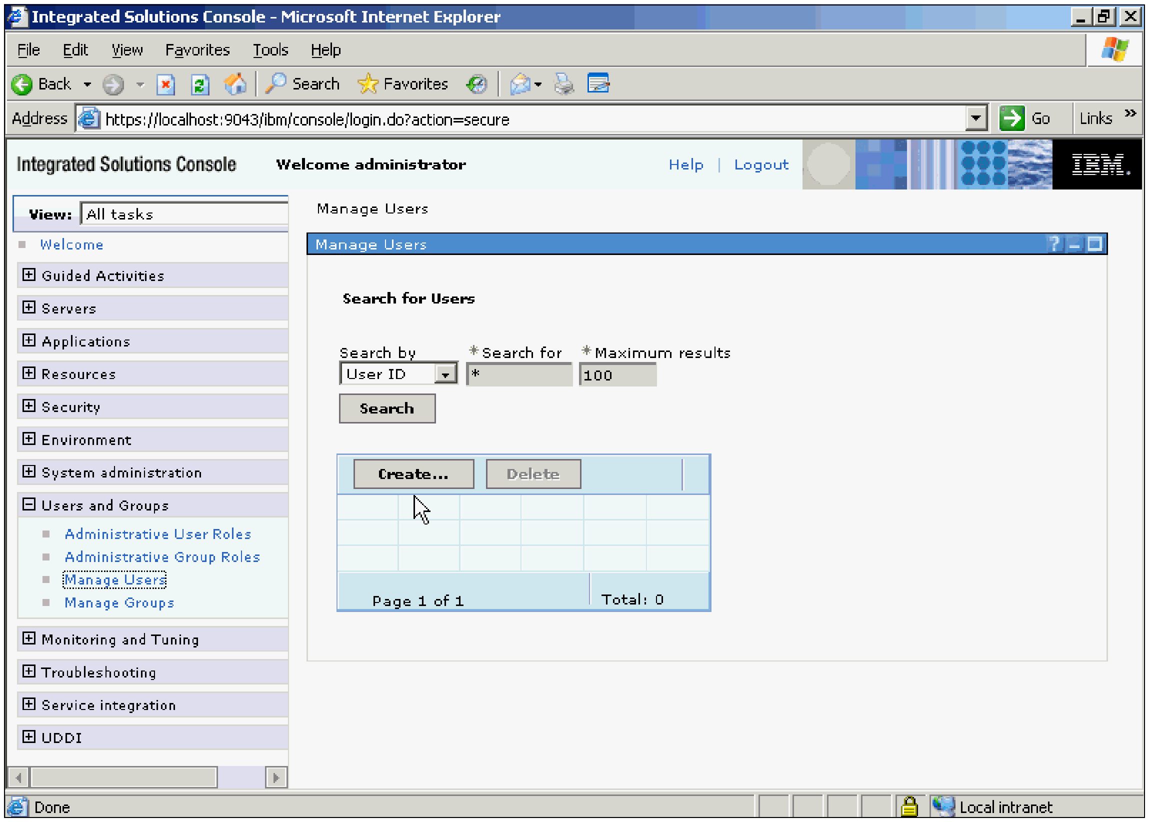
Integrated Solutions Console provides a common administration console for multiple products. The table lists the product suites that can be administered through this installation. Select a product suite to view more information.

Suite Name	Version
WebSphere Application Server	6.1.0.0

Integrated Solutions Console, 6.1.0.0
Build Number: b0620.14
Build Date: 5/16/06
LICENSED MATERIALS PROPERTY OF IBM
5724-i63, 5724-H88, 5655-N01 (C) Copyright International Business

Figure A-27 Manage users menu

3. Click **Create...**, as shown in Figure A-28.



Manage Users

Manage Users

Search for Users

Search by: User ID
* Search for: *
* Maximum results: 100

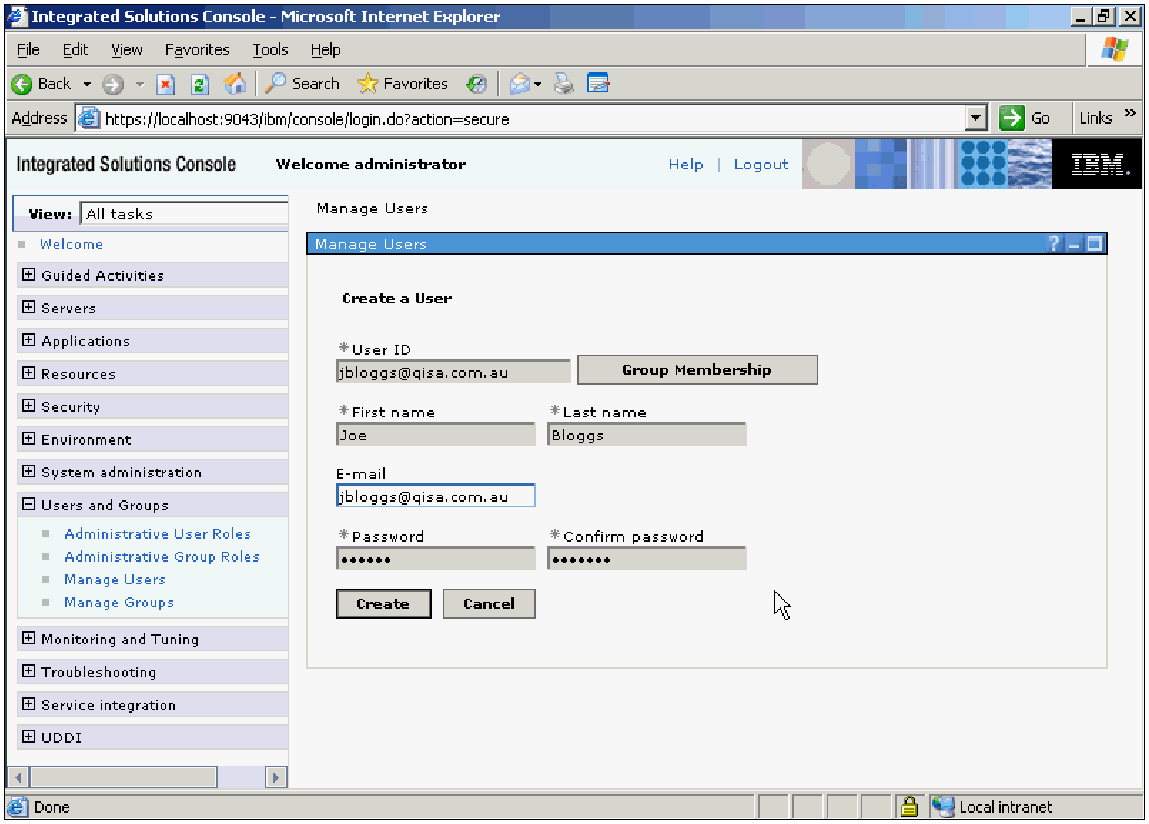
Search

Create... Delete

Page 1 of 1 Total: 0

Figure A-28 Search for Users window

4. Enter the following information to define the new user, as shown in Figure A-29:
   a. In the User ID field, enter jbloggs@qisa.com.au.
   b. In the First name field, enter Joe.
   c. In the Last name field, enter Bloggs.
   d. In the E-mail field, enter jbloggs@qisa.com.au.
   e. In the Password field, enter filenet.
   f. In the Confirm password field, enter filenet.
   g. Click **Create**.



Create a User

* User ID: jbloggs@qisa.com.au

* First name: Joe

* Last name: Bloggs

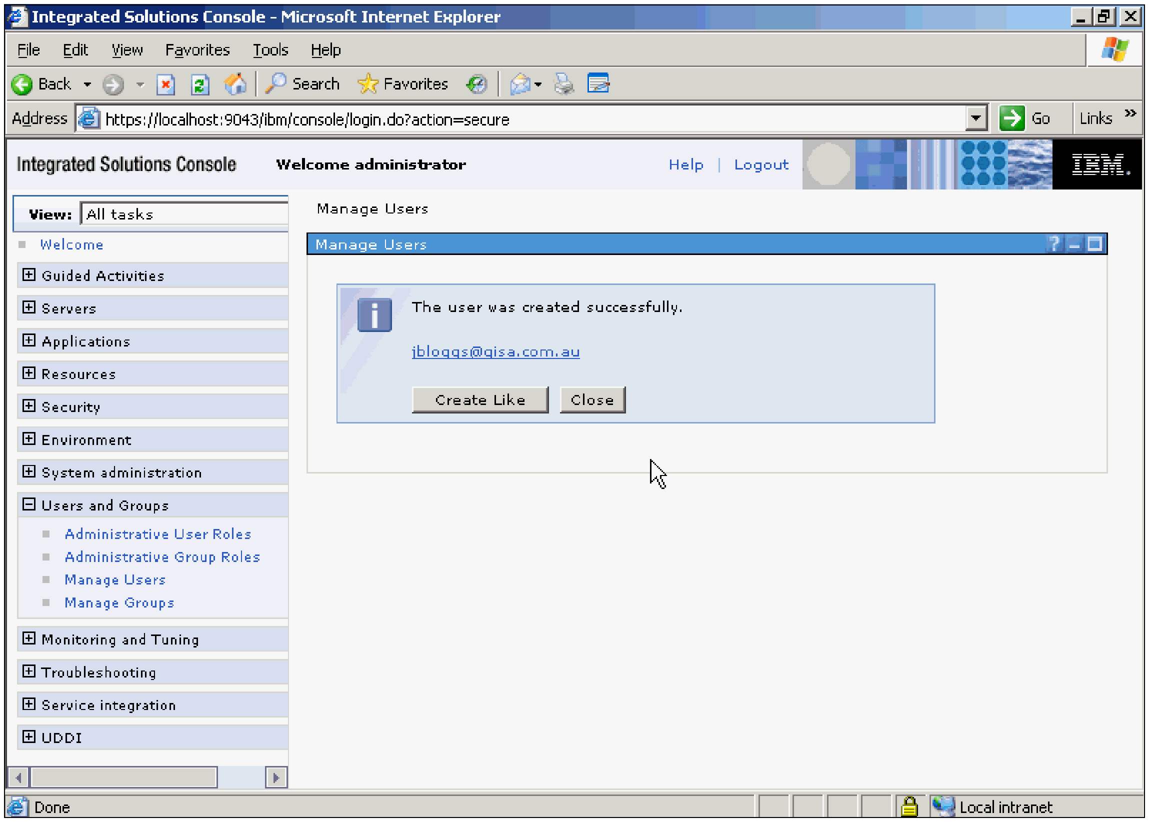
E-mail: jbloggs@qisa.com.au

* Password: ••••••

* Confirm password: •••••••

Figure A-29 Create a User window

5. Click **Close**, as shown in Figure A-30.

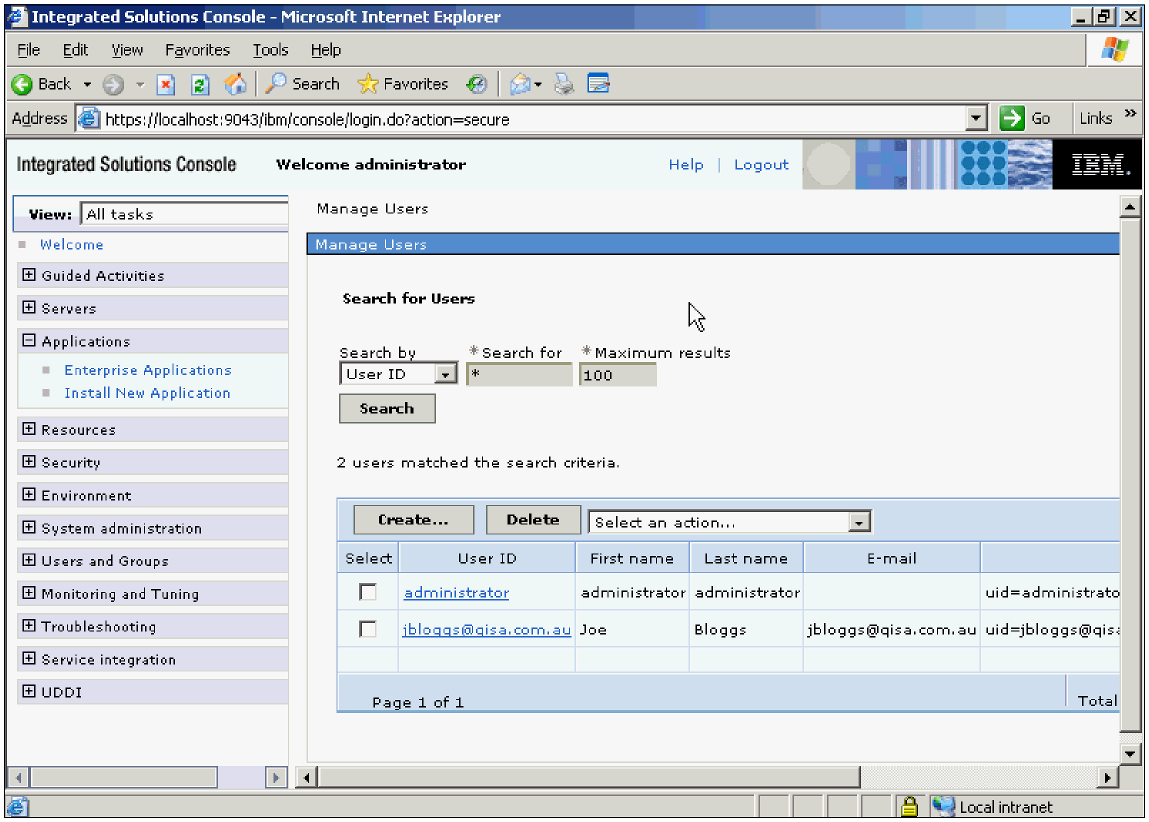


The user was created successfully.

jbloggs@qisa.com.au

Figure A-30 User created successfully window

6. Select **Applications ∅ Enterprise Applications**, as shown in Figure A-31.

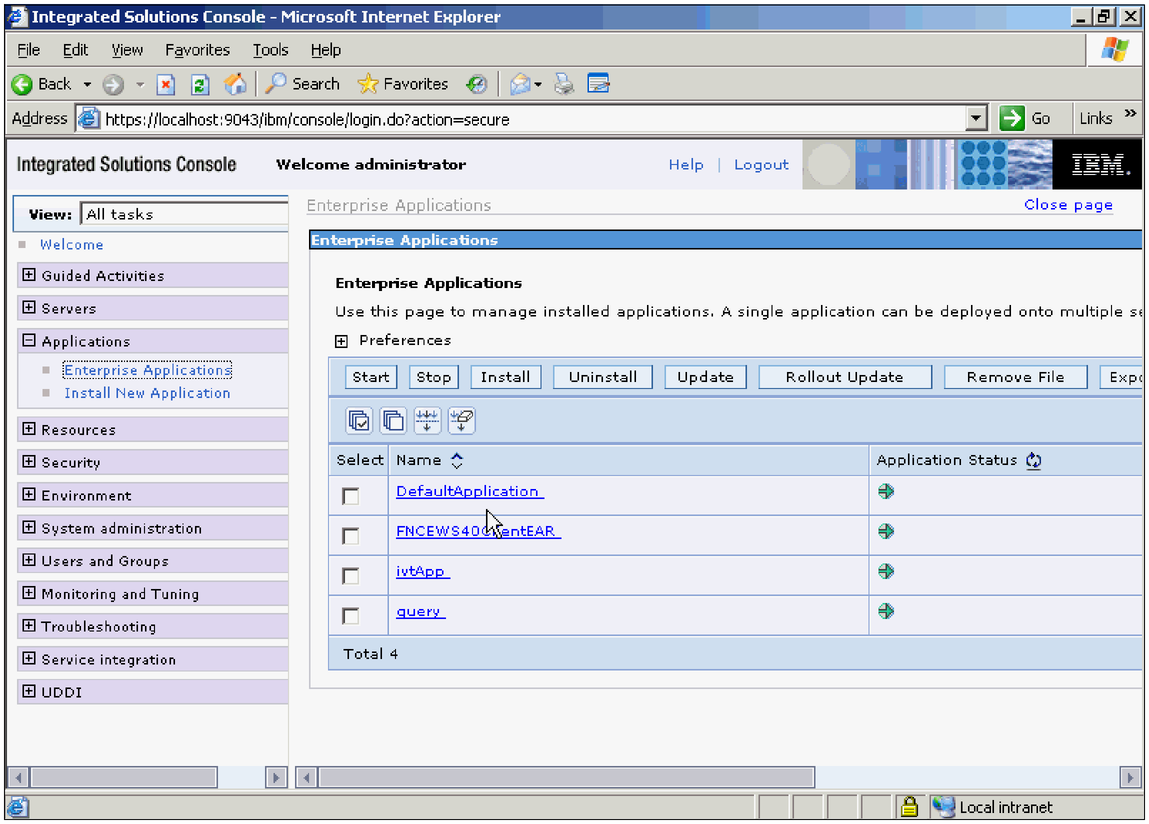


2 users matched the search criteria.

Select	User ID	First name	Last name	E-mail	
☐	administrator	administrator	administrator		uid=administrato
☐	jbloggs@qisa.com.au	Joe	Bloggs	jbloggs@qisa.com.au	uid=jbloggs@qisa

Figure A-31 Applications menu

7. Click **FNCEWS40ClientEAR**, as shown in Figure A-32.

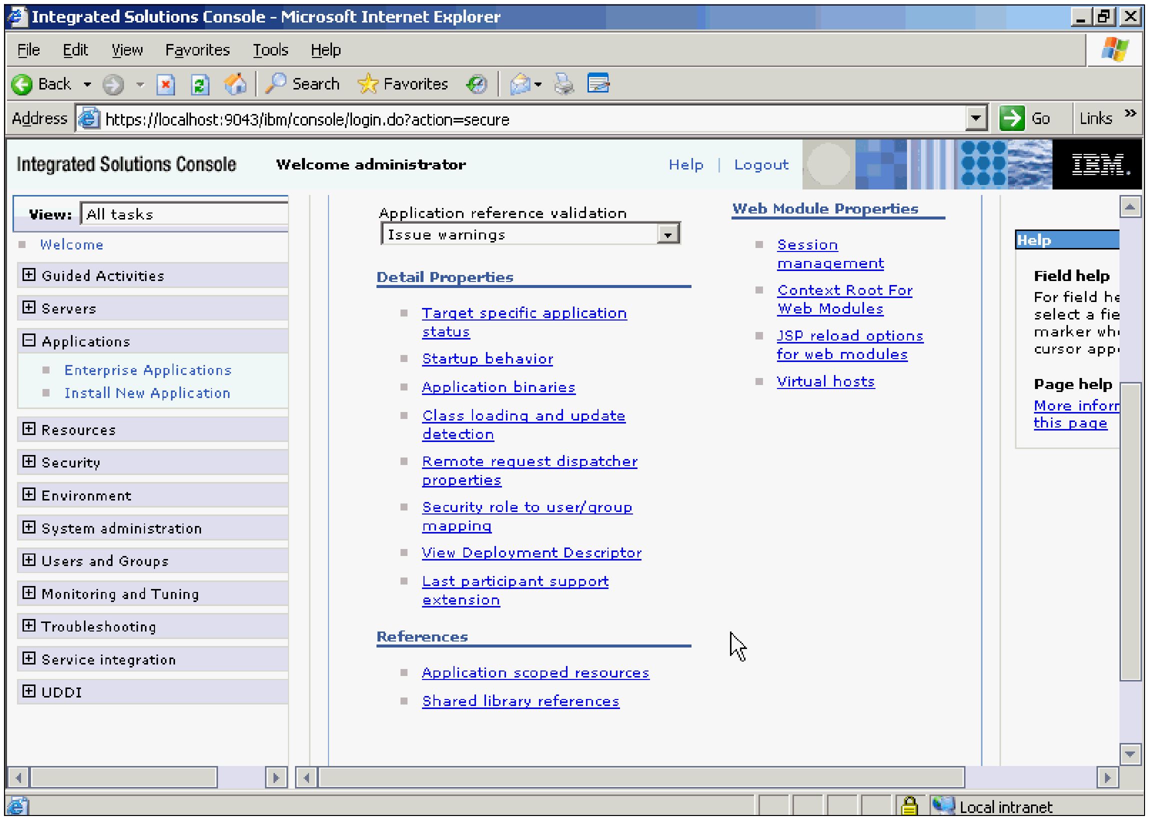

Enterprise Applications

Use this page to manage installed applications. A single application can be deployed onto multiple se

Select	Name	Application Status
☐	DefaultApplication	●
☐	FNCEWS40ClientEAR	●
☐	ivtApp	●
☐	query	●
Total 4		

Figure A-32 Enterprise Applications window

8. In the Detail Properties pane, click **Security role to user/group mapping**, as shown in Figure A-33.



Integrated Solutions Console - Microsoft Internet Explorer

Address https://localhost:9043/ibm/console/login.do?action=secure

Integrated Solutions Console Welcome administrator Help | Logout

View: All tasks

- Welcome
- Guided Activities
- Servers
- Applications
  - Enterprise Applications
  - Install New Application
- Resources
- Security
- Environment
- System administration
- Users and Groups
- Monitoring and Tuning
- Troubleshooting
- Service integration
- UDDI

Application reference validation: Issue warnings

Detail Properties

- Target specific application status
- Startup behavior
- Application binaries
- Class loading and update detection
- Remote request dispatcher properties
- Security role to user/group mapping
- View Deployment Descriptor
- Last participant support extension

References

- Application scoped resources
- Shared library references

Web Module Properties

- Session management
- Context Root For Web Modules
- JSP reload options for web modules
- Virtual hosts

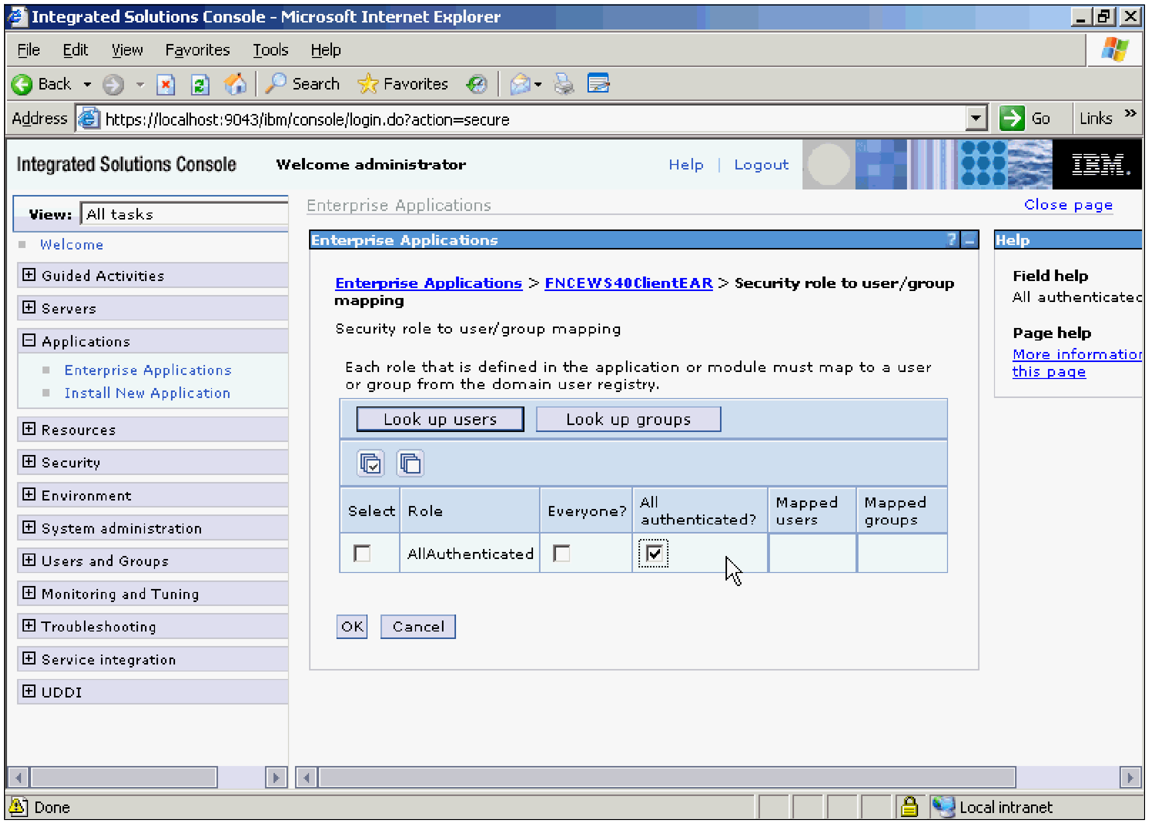
Help

Field help
For field he
select a fie
marker wh
cursor app

Page help
More infor
this page

Figure A-33 Application settings window

9. In the AllAuthenticated role, check **All authenticated?** and then click **OK**, as shown in Figure A-34.



Enterprise Applications

Enterprise Applications > FNCEWS40ClientEAR > Security role to user/group mapping

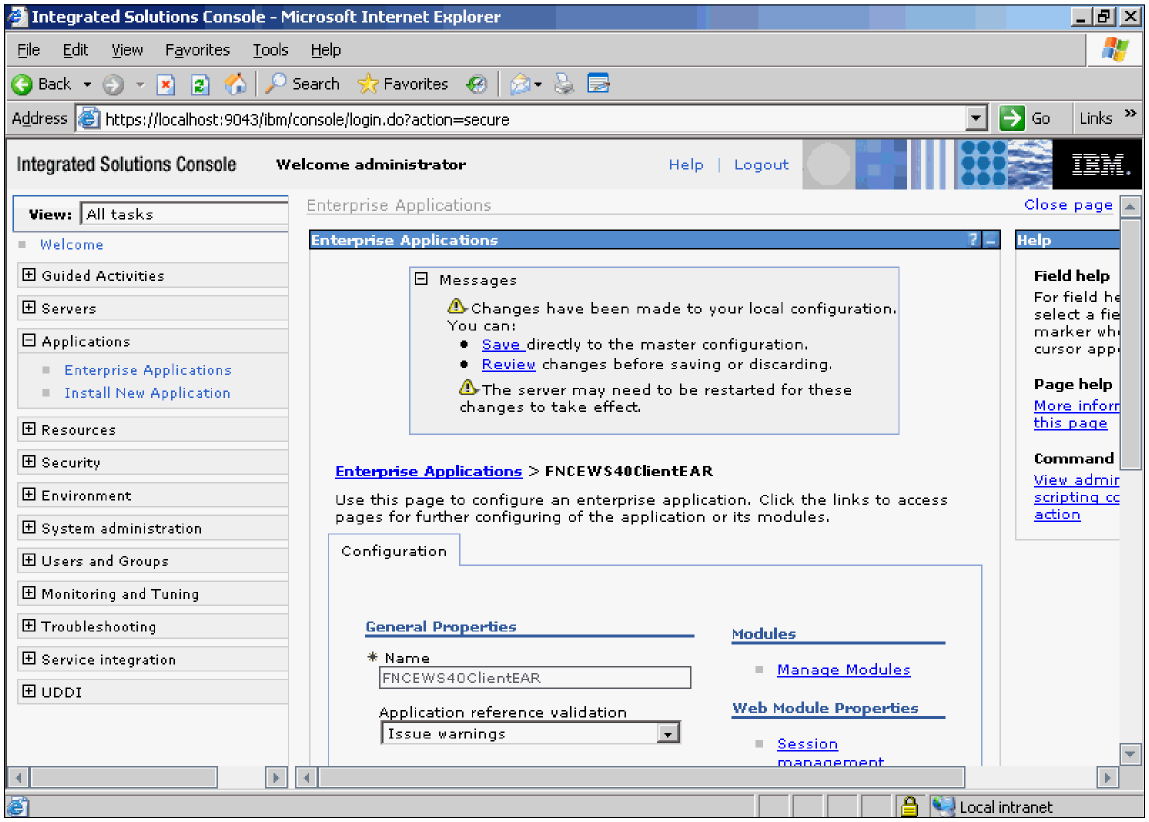
Security role to user/group mapping

Each role that is defined in the application or module must map to a user or group from the domain user registry.

Select	Role	Everyone?	All authenticated?	Mapped users	Mapped groups
☐	AllAuthenticated	☐	☑		

Figure A-34 Security role to user/group mapping window

10. In the Messages pane at the top of the Application Servers window, click **Save** to commit your changes, as shown in Figure A-35.



Messages

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take effect.

Enterprise Applications > FNCEWS40ClientEAR

Use this page to configure an enterprise application. Click the links to access pages for further configuring of the application or its modules.

Configuration

General Properties

* Name: FNCEWS40ClientEAR

Application reference validation: Issue warnings

Modules

- Manage Modules

Web Module Properties

- Session management

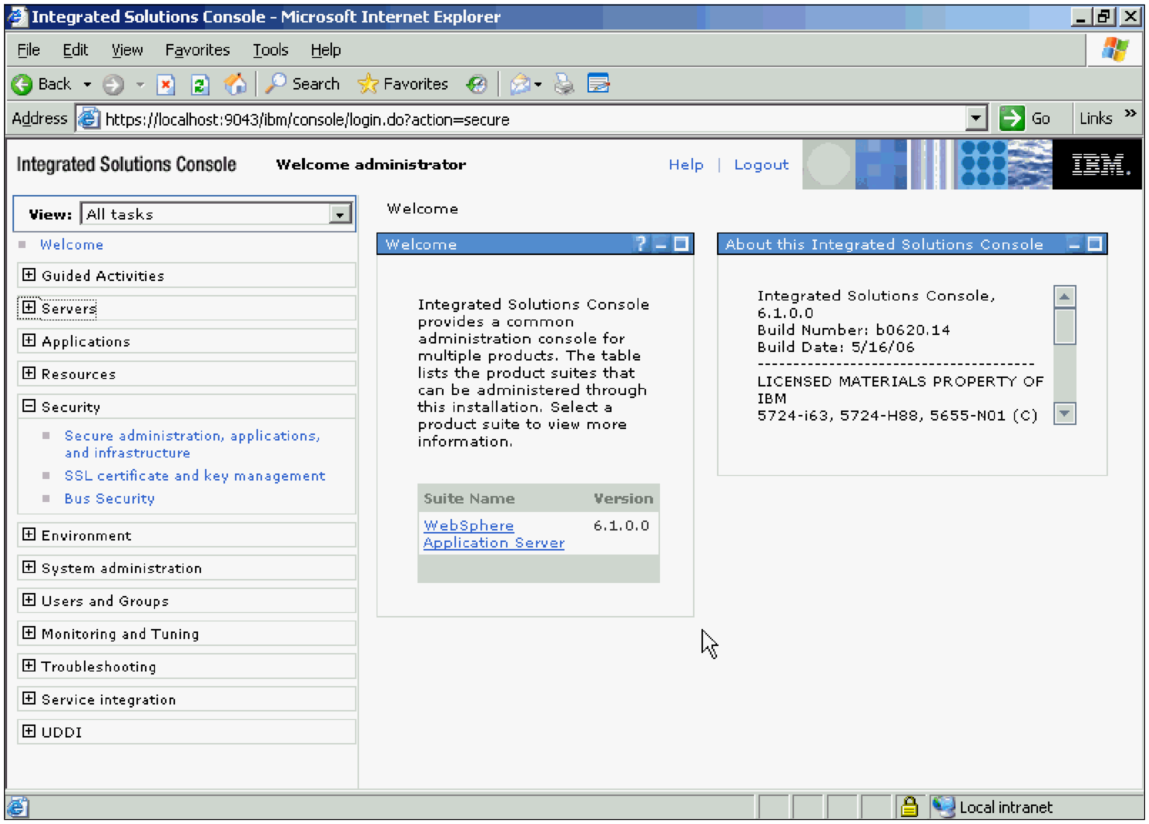
Field help
For field he
select a fie
marker wh
cursor app

Page help
More infor
this page

Command
View admir
scripting co
action

Figure A-35 Save messages pane

11. Select **Security ∅ Secure administration, applications, and infrastructure**, as shown in Figure A-36.


Integrated Solutions Console - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://localhost:9043/ibm/console/login.do?action=secure
Go Links
Integrated Solutions Console
Welcome administrator
Help | Logout
IBM.
View: All tasks
Welcome
Guided Activities
Servers
Applications
Resources
Security
Secure administration, applications, and infrastructure
SSL certificate and key management
Bus Security
Environment
System administration
Users and Groups
Monitoring and Tuning
Troubleshooting
Service integration
UDDI
Welcome
Welcome
About this Integrated Solutions Console
Local intranet

Integrated Solutions Console provides a common administration console for multiple products. The table lists the product suites that can be administered through this installation. Select a product suite to view more information.

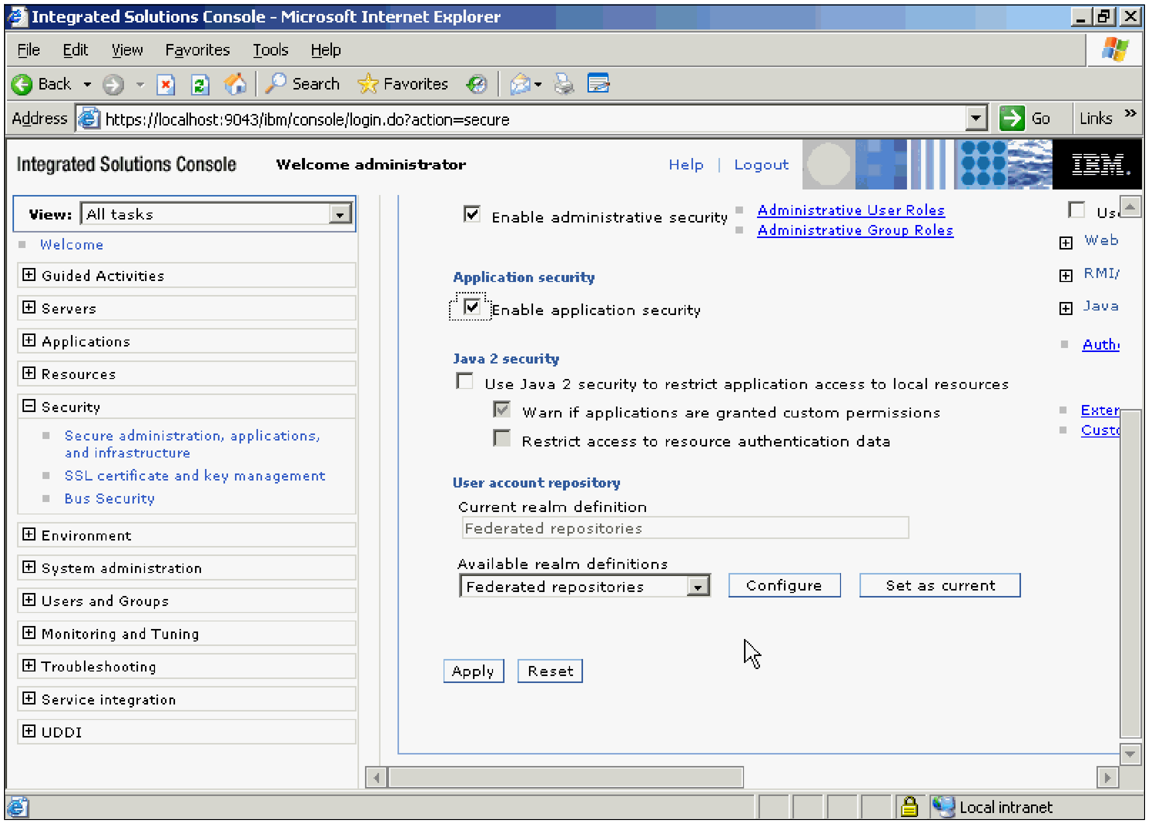
Suite Name	Version
WebSphere Application Server	6.1.0.0

Integrated Solutions Console, 6.1.0.0
Build Number: b0620.14
Build Date: 5/16/06

LICENSED MATERIALS PROPERTY OF IBM
5724-i63, 5724-H88, 5655-N01 (C)

Figure A-36 Secure administration, applications, and infrastructure menu

12. In the Application security pane, check **Enable application security** and click **Apply**, as shown in Figure A-37.


Integrated Solutions Console - Microsoft Internet Explorer

Address https://localhost:9043/ibm/console/login.do?action=secure

Integrated Solutions Console Welcome administrator Help | Logout

View: All tasks
- Welcome
- ⊞ Guided Activities
- ⊞ Servers
- ⊞ Applications
- ⊞ Resources
- ⊟ Security
  - Secure administration, applications, and infrastructure
  - SSL certificate and key management
  - Bus Security
- ⊞ Environment
- ⊞ System administration
- ⊞ Users and Groups
- ⊞ Monitoring and Tuning
- ⊞ Troubleshooting
- ⊞ Service integration
- ⊞ UDDI

☑ Enable administrative security
- Administrative User Roles
- Administrative Group Roles

Application security

☑ Enable application security

Java 2 security

☐ Use Java 2 security to restrict application access to local resources

☑ Warn if applications are granted custom permissions

☐ Restrict access to resource authentication data

User account repository

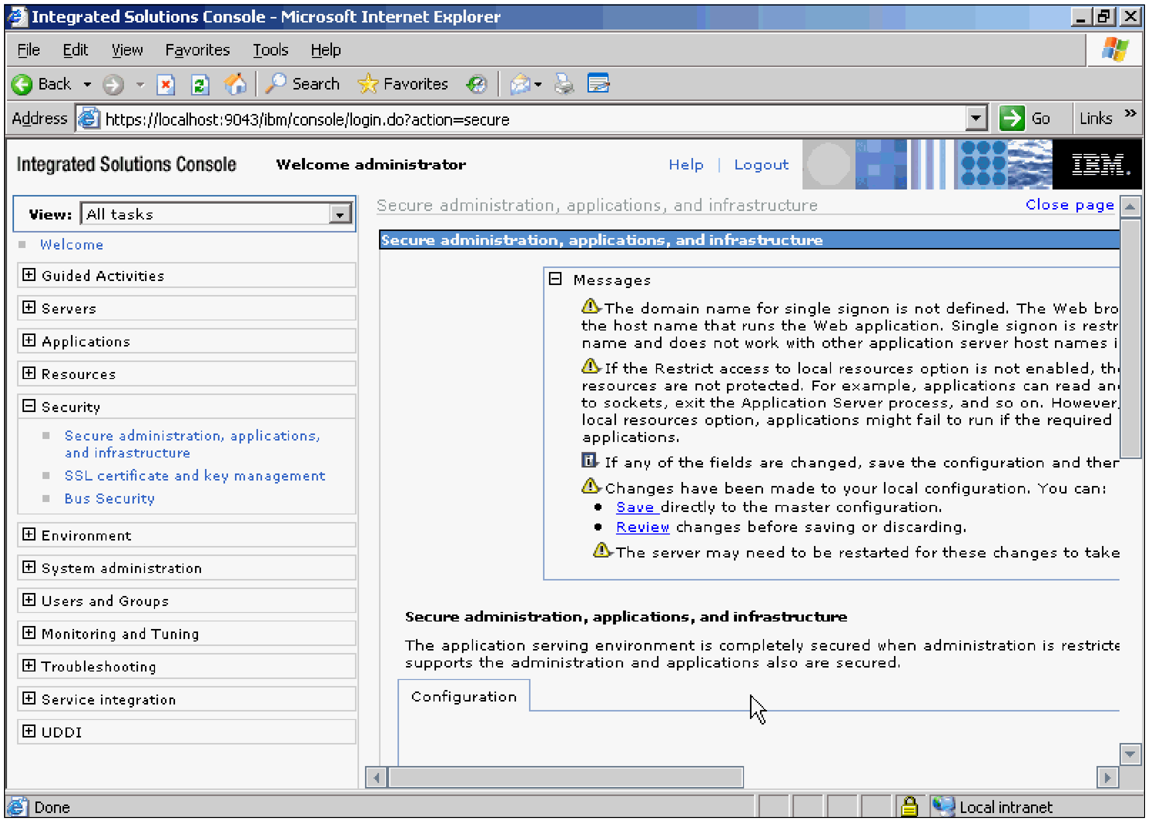
Current realm definition: Federated repositories

Available realm definitions: Federated repositories [Configure] [Set as current]

[Apply] [Reset]

Figure A-37 Application security window

13. In the Messages pane at the top of the Application Servers window, click **Save** to commit your changes, as shown in Figure A-38.


Integrated Solutions Console - Microsoft Internet Explorer

Address https://localhost:9043/ibm/console/login.do?action=secure

Integrated Solutions Console Welcome administrator Help | Logout

View: All tasks

Secure administration, applications, and infrastructure

Messages

The domain name for single signon is not defined. The Web bro the host name that runs the Web application. Single signon is restr name and does not work with other application server host names i

If the Restrict access to local resources option is not enabled, th resources are not protected. For example, applications can read an to sockets, exit the Application Server process, and so on. However, local resources option, applications might fail to run if the required applications.

If any of the fields are changed, save the configuration and ther

Changes have been made to your local configuration. You can:

- Save directly to the master configuration.
- Review changes before saving or discarding.

The server may need to be restarted for these changes to take

Secure administration, applications, and infrastructure

The application serving environment is completely secured when administration is restricte supports the administration and applications also are secured.

Figure A-38 Save changes window

14. Restart the WebSphere Application Server.

We have completed the Web service user setup. We now need to configure TCPMon to relay the Web service client request to the appropriate service.

Configuring TCPMon to relay FileNet CEWS SAML 2.0 Client requests

The sample FileNet CEWS SAML 2.0 Client has been designed to send the Web service request to the localhost machine on port 19080. In order to send this request to the appropriate FileNet Web service consumer, TCPMon can be used.

TCPMON is a simple Java GUI utility that allows you to look at on-the-wire Web services messages by configuring it as a listener and forwarding messages to their intended destination. There is a much information available about TCPMON on the Web. In order to configure TCPMon for our purposes, perform the following steps:

1. Start the TCPMonitor application provided with WebSphere Application Server. Refer to the “Tracing SOAP messages with tcpmon” topic in the WebSphere Application Server Information Center for further information[1].
2. In the Listen Port # field, specify the port number 19080.
3. To configure the TCPMonitor application to act as a listener, ensure that **Listener** is selected, and then enter the following information:
   a. In the Target Hostname field, enter the host name of the FileNet P8 Content Engine system.
   b. In the Target Port # field, enter the port number that the FileNet P8 Content Engine is using for SOAP requests (typically 9080).
4. Click **Add** to enable the TCPMonitor application to listen for requests. After this operation completes, the fields in the window are cleared and a new tab appears at the top of the window.
5. Click the **Port 19080** tab at the top of the window, where 19080 is the port number that was added in the previous step.

Now that this configuration has been completed, you can perform an operation using the FileNet CEWS SAML 2.0 Client application. The TCPMonitor window shows the request from the FileNet CEWS SAML 2.0 Client in the middle pane and the response from the FileNet P8 Content Engine in the bottom pane.

Conclusion

We showed how to configure and install the sample FileNet CEWS SAML 2.0 Client, which uses the Tivoli Federated Identity Manager Web services security management component within WebSphere Application Server. We also showed how to use TCPMon to not only view SOAP requests being sent by the client, but to also redirect the Web service client request to the appropriate Web service client consumer.

[1] You can find this Information Center at `http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp`.

B

Additional material

This book refers to additional material that can be downloaded from the Internet, as described below.

Locating the Web material

The Web material associated with this book is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser at:

`ftp://www.redbooks.ibm.com/redbooks/SG247675/SG247675.zip`

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG24-7675.

Using the Web material

The additional Web material that accompanies this book includes the following files:

File name	*Description*
SG247675.zip	Zipped code and configuration files

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

The zip file contains the following files:

- `TFIMFileNetP8Interceptor.jar`: Federated Identity Manager FileNet P8 Interceptor jar
- `ws-trust.wsdl`: TFIM WS-Trust WSDL file used with the TFIMFileNetP8Interceptor
- `tfim-filenet-p8-test-map.xsl`: Sample Federated Identity Manager XSLT FileNet P8 mapping rule
- `FNCEWS40SAML20ClientEAR.ear`: Sample FileNet CEWS SAML 2.0 client

Do the following steps:

1. Copy the `TFIMFileNetP8Interceptor.jar` and the `ws-trust.wsdl` files to a temporary directory on the FileNet P8 Content Engine machine.
2. Copy the `tfim-filenet-p8-test-map.xsl` file to a temporary directory on the Tivoli Federated Identity Manager machine.
3. Copy the `FNCEWS40SAML20ClientEAR.ear` file to a temporary directory on to a machine running WebSphere Application Server to be used as the Web service client.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 272. Note that some of the documents referenced here may be available in softcopy only.

- *Enterprise Security Architecture Using IBM Tivoli Security Solutions*, SG24-6014
- *IBM FileNet Content Manager Implementation Best Practices and Recommendations*, SG24-7547
- *Introducing IBM FileNet Business Process Manager*, SG24-7509
- *Propagating Identity in SOA with Tivoli Federated Identity Manager*, REDP-4354
- *Robust Data Synchronization with IBM Tivoli Directory Integrator*, SG24-6164
- *Understanding SOA Security Design and Implementation*, SG24-7310

Other publications

These publications are also relevant as further information sources:

- *IBM Tivoli Access Manager for e-business Administration Guide Version 6.1*, SC23-6504
- *IBM Tivoli Access Manager for e-business Installation Guide Version 6.1*, GC23-6502
- *IBM Tivoli Access Manager for e-business WebSEAL Administration Guide Version 6.1*, SC23-6505
- *IBM Tivoli Federated Identity Manager Version 6.2 Administration Guide*, SC23-6191
- *IBM Tivoli Federated Identity Manager Version 6.2 Installation and Configuration Guide*, SC23-6190

Online resources

These Web sites are also relevant as further information sources:

- IBM Tivoli product documentation, manuals, and information centers can be found at this Web link:

  http://publib.boulder.ibm.com/tividd/td/tdmktlist.html

- IBM FileNet P8 product documentation, manuals, and information centers can be found at this Web link:

  http://www.ibm.com/support/docview.wss?rs=3278&uid=swg27010422

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

Single Sign-On Solutions for IBM FileNet P8 Using IBM Tivoli

(0.5” spine)
0.475”<->0.875”
250 <-> 459 pages

Single Sign-On Solutions for IBM FileNet P8

Using IBM Tivoli and WebSphere Security Technology

Business context discussion on SSO in an Enterprise Content Management solution

Overview of SSO architecture and deployment models

Complete hands-on SSO configurations for P8 V4.0

Authentication is the act of verifying a user's identity based on the credentials that they have presented. Establishing each user's identity is a critical first step in any client/server based system. In this IBM Redbooks publication, we present an overview of the set of authentication options in the IBM FileNet P8 V4.0 release.

The two standards at the core of the authentication process in IBM FileNet P8 V4.0 are the Java Authentication and Authorization Service (JAAS) standard and the Web Services Security standard (WS-Security). The JAAS standard forms the framework for security interoperability in the J2EE world, while the WS-Security standard forms the framework for security interoperability in the heterogeneous world of clients and servers that communicate through Web services interfaces. IBM FileNet customers rely on a variety of authentication technologies to secure their corporate intranets. By implementing and adhering to these two standards, IBM FileNet P8 V4.0 enables a wide range of authentication integrations.

In this IBM Redbooks publication we discuss and demonstrate the IBM FileNet P8 integration with IBM Tivoli Access Manager for e-business, IBM Tivoli Federated Identity Manager, and the SPNEGO mechanism provided in IBM WebSphere Application Server.

This book is a valuable resource for security officers, access management administrators, and architects who wish to better understand single sign-on options for the IBM FileNet P8 V4.0 solution.

SG24-7675-00 ISBN 0738432911