WebSphere. software

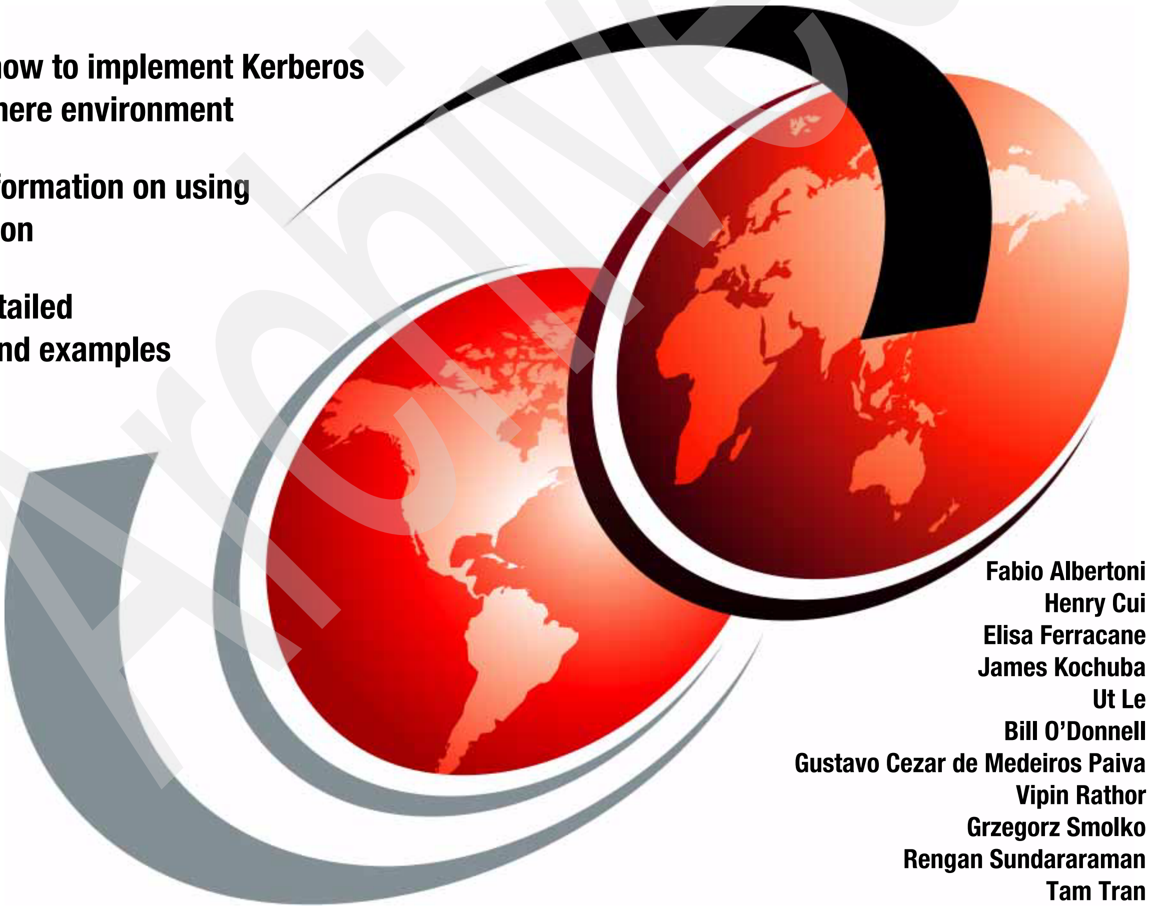
IBM

Implementing Kerberos in a WebSphere Application Server Environment

Discusses how to implement Kerberos in a WebSphere environment

Provides information on using single sign-on

Includes detailed scenarios and examples



Fabio Albertoni
Henry Cui
Elisa Ferracane
James Kochuba
Ut Le
Bill O'Donnell
Gustavo Cezar de Medeiros Paiva
Vipin Rathor
Grzegorz Smolko
Rengan Sundararaman
Tam Tran

Redbooks

ibm.com/redbooks

International Technical Support Organization

Implementing Kerberos in a WebSphere Application Server Environment

October 2009

SG24-7771-00

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

First Edition (October 2009)

This edition applies to IBM WebSphere Application Server V7.

Contact an IBM Software Services Sales Specialist

Start SMALL, Start BIG, ... **JUST START**

architectural knowledge, skills, research and development . . .

that's IBM Software Services for WebSphere.

Our highly skilled consultants make it easy for you to design, build, test and deploy solutions, helping you build a smarter and more efficient business. **Our worldwide network of services specialists wants you to have it all!** Implementation, migration, architecture and design services: IBM Software Services has the right fit for you. We also deliver just-in-time, customized workshops and education tailored for your business needs. You have the knowledge, now reach out to the experts who can help you extend and realize the value.

For a WebSphere services solution that fits your needs, contact an IBM Software Services Sales Specialist:
ibm.com/developerworks/websphere/services/contacts.html

Contents

Notices

Trademarks

Preface

This IBM® Redbooks® publication discusses Kerberos technology with IBM WebSphere® Application Server V7.0.0.5 on distributed platforms. IBM WebSphere Application Server V7.0.0.5 Kerberos Authentication and single sign-on (SSO) features enable interoperability and identity propagation with other applications (such as .NET, DB2®, and others) that support the Kerberos authentication mechanism. With this feature, a user can log in once and then can access other applications that support Kerberos Authentication without having to log in a second time. It provides an SSO, end-to-end interoperability solution and preserves the original requester identity.

This book provides a set of common examples and scenarios that demonstrate how to use the Kerberos with WebSphere Application Server. The scenarios include configuration information for WebSphere Application Server V7 when using a KDC from Microsoft®, AIX®, and z/OS® as well as considerations when using these products. The intended audience for this book is system administrators and developers who use IBM WebSphere Application Server V7 on distributed platforms.

The team who wrote this book

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

Fabio Albertoni is a Senior IT Specialist working in Integrated Technology Delivery SSO, on Hortolandia, Brazil. He has 12 years of experience in the IT and banking industries. He has spent the last 8 years developing and implementing integrated solutions using WebSphere Application Server and MQ-Series. He holds a degree in Data Processing from FATEC University of Ourinhos and a Masters degree in Computer Engineering from Instituto de Pesquisas Tecnologicas of Sao Paulo, Brazil.

Henry Cui is a Software Developer working at the IBM Toronto lab. Henry has been in the IBM Rational® Application Developer service and support team for 6 years. He has helped many customers resolve design, development, and migration issues with Web services development. He is the subject matter expert (SME) for Web services on his team. His areas of expertise include developing Java™ EE applications that use Rational tools, configuring WebSphere Application Servers, EJBs, application security, Web services, and SOA. Henry is a frequent contributor of developerWorks® articles. He also co-authored three IBM Redbooks publications that are related to Web services. Henry holds a degree in Computer Science from York University.

Elisa Ferracane is a software developer working with the IBM WebSphere Security Development team in Austin, Texas. Her areas of expertise are WebSphere Security for z/OS, Kerberos on z/OS, and CSIv2. She has also worked as a system tester for WebSphere Application Server for z/OS. She has been with IBM for over 7 years. Elisa received a Bachelor's degree in Computer Engineering from the University of Puerto Rico.

James Kochuba is the WebSphere Application Server Security Test Architect in IBM Research Triangle Park, North Carolina. He has 7 years of experience in WebSphere field. He holds a Master of Science degree in Electrical Engineering from University of Virginia. His areas of expertise include WebSphere Application Server support and development. He has written extensively on WebSphere Application Server usage and presented at IBM Impact conference.

Ut Le is an Advisory Software Engineer at IBM Austin, Texas. He has 20 years of experience in the software industry. He holds a Bachelor of Engineering degree from Marist College, New York. He is currently the SPNEGO and Kerberos developer lead for IBM WebSphere security development team. He was the development and release lead for IBM Network Authentication Service (IBM Kerberos). His areas of expertise are WebSphere security for SPNEGO, Kerberos, SSO, trust association interceptor (TAI), and authentication.

Bill O'Donnell is the Lead WebSphere Application Server Security Architect responsible for the Security Design and Architectural aspect for the WebSphere Application Server. Bill has over 20 years

Gustavo Cezar de Medeiros Paiva is a Certified IT Specialist working on Integrated Technology Delivery SSO, in Hortolandia, Brazil. He has 9 years of experience in IT. He is an expert in IBM WebSphere Application Server and certified in WebSphere Application Server and Portal. His areas of expertise include WebLogic Application Server, Linux®, AIX, and Windows® platforms. Gustavo holds a degree in computer science.

Vipin Rathor is a System Software Engineer in IBM India Software Lab. He has 3 years of experience in Network Security (especially Kerberos) field. He holds a master degree in Computer Science from University of Pune. His areas of expertise include IBM NAS (IBM Kerberos) and authentication protocols. He has written extensively on using IBM NAS with other Kerberized applications on AIX and other UNIX® platforms.

Grzegorz Smolko is a Certified IT Specialist with IBM Poland in Warsaw. Grzegorz has been working for IBM for 6 years in IBM Software Services for WebSphere. Prior to joining IBM, he worked for software house companies in Poland as Java developer and Java EE architect. His areas of expertise include Java, Java EE and WebSphere. He holds certifications from Sun and IBM in Java and WebSphere technologies. He has a Master's degree in Computer Science from the Warsaw University of Technology, Poland.

Rengan Sundararaman is a Advisory Engineer in IBM Research Triangle Park, North Carolina. He has 17 years of experience in the software industry. He holds Masters of Engineering degree from NC State University. He is currently working on the plusOne project, an IBM Strategy initiative to provide scalable ways to solve major consumability issues. He was the development lead for creating Situation Based Package (SBP) for enabling WS-SecurityKerberos for JAX-RPC applications on IBM WebSphere Application Server V6.0.2 and V6.1. Prior to joining the plusOne project, he worked in IBM WebSphere Application Server development, Network Dispatcher, Host on Demand, and PCOMM.

Tam Tran is an Advisory Software Engineer in IBM Software Group in Austin, Texas. He has 10 years of experience in IBM Software Group with 3 years of experience in WebSphere Application Server. He holds a bachelor's degree in Mechanical Engineering from The University of Texas at Austin. His area of expertise includes functional verification testing for WebSphere Application Server Security, including SPNEGO and Kerberos.

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

Margaret Ticknor, Project leader
ITSO, Raleigh Center, North Carolina, U. S.

Carla Sadtler
ITSO, Raleigh Center, North Carolina, U. S.

Tamikia Barrow
ITSO, Raleigh Center, North Carolina, U. S.

Rich Conway
ITSO, Poughkeepsie Center, New York, U. S.

Derek Ho
IBM Software Group, Application and Integration Middleware Software
Austin, Texas, U. S.

Salim Zeitouni
IBM Software Group, Application and Integration Middleware Software
Web Services Interop Team

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 publications 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

1

Introduction

IBM WebSphere Application Server V7.0.0.5 Kerberos Authentication and single sign-on (SSO) features enable interoperability and identity propagation with other applications (such as .NET, DB2, and others) that support the Kerberos authentication mechanism. With this feature, a user can log in once and then can access other applications that support Kerberos Authentication without having to log in a second time. It provides an SSO, end-to-end interoperability solution and preserves the original requester identity.

Kerberos technology can be quite daunting. In this book, we focus on a set of common scenarios that demonstrate how to use the Kerberos technology with WebSphere Application Server. This book can be useful in understanding and enabling Kerberos in your environment. This chapter focuses on the high-level aspects of the Kerberos technology. It includes the following topics:

- What is Kerberos
- What is SPNEGO
- Benefits of using Kerberos technology
- Kerberos terminology
- SSO with SPNEGO and Kerberos
- Kerberos with other authentication protocols
- Web services security using Kerberos
- High-level look at the scenarios

In the chapters that follow, we provide common scenarios that provide examples of using the Kerberos technology.

1.1 What is Kerberos

Kerberos has withstood the test of time and today enjoys wide-spread platform support (for example Windows, Linux, Solaris, AIX, and z/OS) partly because Kerberos source code is freely downloadable from Massachusetts Institute of Technology (MIT) where it was originally created. The Kerberos architecture was designed and developed in the 1980s by MIT as part of the Athena Project. Its name derives from *Cerberus*, the legendary three-headed dog of Greek mythology, which guarded the gates of the underworld. He made sure that only the souls of the dead could enter Hades and that no souls could escape.

Kerberos is a standard network authentication protocol used in providing a proof of identity between a client and server or between a server and a server. Kerberos takes advantage of cryptography as a way to secure the identity during the identity exchange. Kerberos offers SSO interoperability with other applications that support Kerberos authentication.

Using the Kerberos technology allows a user to log in once and then have access to other applications that support Kerberos Authentication without logging in a second time.

1.2 What is SPNEGO

Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO) is a standard protocol that is used to negotiate the authentication protocol used when a client application wants to authenticate to a remote server. SPNEGO is a standard specification defined in IETF RFC 2478. SPNEGO is used in a Web SSO. It is responsible for authenticating access to a secured paged, such as a WebSphere Application Server resource that is identified in an HTTP request. Microsoft also uses SPNEGO for its browser-based SSO solutions.

SPNEGO is most commonly used in Microsoft's HTTP Negotiate authentication extension, implemented in Microsoft's Internet Explorer. Microsoft first introduced this extension in Internet Explorer 5.0.1 and in IIS 5.0 to provide Microsoft customers SSO capability, which is marketed as Integrated Windows Authentication. The HTTP Negotiated extension was later implemented with similar support in Mozilla 1.7 beta, Mozilla Firefox 0.9, and Konqueror 3.3.1.

By taking advantage of the Microsoft Kerberos mechanism and the Microsoft SPNEGO implementation, Microsoft offers an SSO solution for Web applications.

1.3 Benefits of using Kerberos technology

There are many benefits in using Kerberos as the authentication mechanism for your applications running in WebSphere Application Server. The Kerberos protocol is standard, enabling the ability to interoperate with other applications (such as .NET, DB2, and others) that support the Kerberos authentication mechanism. Basically, the Kerberos technology provides an SSO solution, making interoperability possible.

When using Kerberos authentication, the user's text password never leaves the user's personal computer. After the user logs in to the system, the user is issued an encrypted Kerberos ticket that allows the user to gain access to other applications. After a user logs in, the user can gain access to J2EE, Web services, .NET, Web browser clients, and more without logging in a second time, using the Kerberos and the SPNEGO technology.

1.4 Kerberos terminology

As with any technology, Kerberos comes with a set of terminologies that are important to understand in order to use the solution effectively.

Key distribution center

A *key distribution center* (KDC) is an integral part of using Kerberos. Kerberos uses symmetric key cryptography and requires a trusted third party, in this case the KDC. The KDC has three logical components:

- An authentication server

  The *authentication server* handles requests from a client that wants to obtain a Kerberos ticket representing proof of identity. The authentication server first authenticates the client (for example, with a user ID and password verification). If the authentication is successful, the authentication server returns a Kerberos ticket called the *ticket-granting ticket* (TGT) that represents proof of identity.

- A ticket-granting server

  The *ticket-granting server* (TGS) handles requests for a *service ticket*, which the client uses to access a TGT application or service. The TGS validates the client's TGT and returns a service ticket.

- A user repository

  The *user registry* (sometimes refer to as the *user database*) holds Kerberos user information, such as the user ID, password, and the shared secret information.

Kerberos realm and principal

A *Kerberos realm* is often referred to as an *administrative domain*. A realm consists of members, which can be users, servers, services, or network resources, that are registered within a KDC database. Each of these members has a unique identifier called a *principal*. The Kerberos realm is made up of the KDC and all of its principals.

The principal is a unique identifier to which the KDC can assign tickets. A principal name includes the following components, as shown in the following example:

```
primary/instancename@REALM
```

- The `primary` name can be the user's name, the host, or the name of the service. An example of a principal in the KKDC.TEST.COM realm is `CUI@KKDC.TEST.COM`.
- The `instancename` is optional. It is used to further define the primary name, for example `CUI/admin@KKDC.TEST.COM`. Note that the principals `CUI` and `CUI/admin` are two completely separate principals with different passwords and possibly a different set of authorities.

  The `instancename` component is also used to specify a host or a service principal. In this case, it can be the fully qualified domain name of the host, such as `telnet/kcgg1d6.kkdc.test.com@KKDC.TEST.COM`.
- The `REALM` name is the name of the Kerberos realm, which is usually the domain name in uppercase letters.

Kerberos ticket

The word *ticket* is used to describe how authentication data is transmitted in the Kerberos environment. Tickets are essentially an encrypted data structure that uses shared keys that are issued by the KDC to communicate in a secure fashion. The purpose of the ticket depends on where it was created.

When a client requests an initial authentication, the authentication server authenticates the client. If the authentication is successful, the authentication server returns to the client a TGT that is used to request tickets for other services in the network. When the client wants to use a service in the network, it sends a request including its TGT to the TGS. The TGS responds by issuing and sending a service ticket. When the client uses a service in the network, it sends a request that includes its service ticket to the server that hosts the service. The server accepts the service ticket and executes the service.

Kerberos authentication protocol

Figure 1-1 demonstrates the end-to-end Kerberos authentication technology flow and shows how each of the various Kerberos components work together.

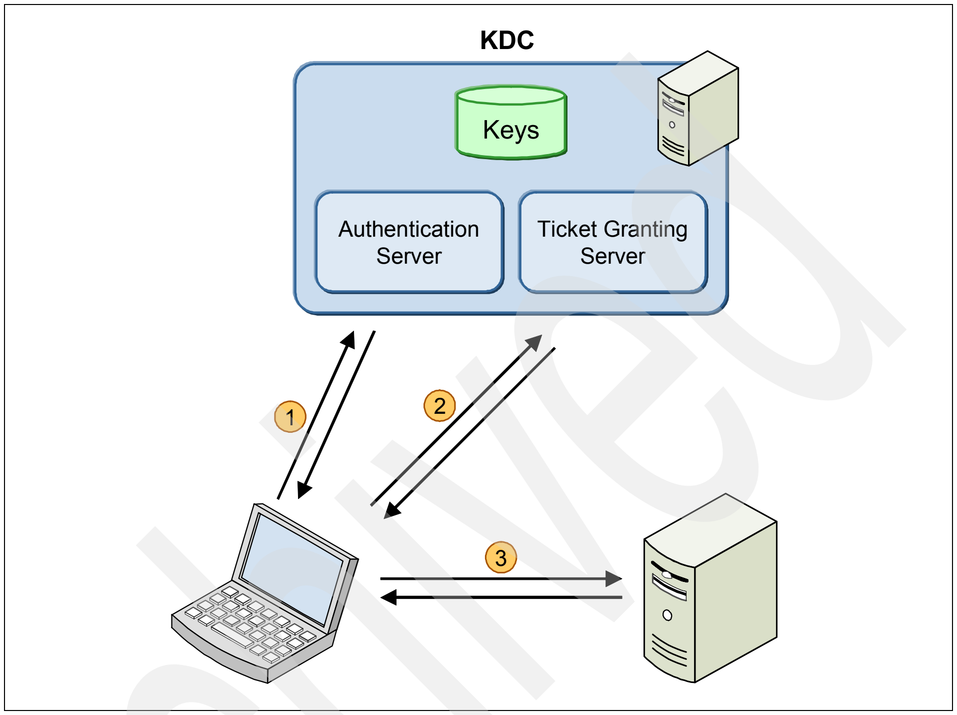


Figure 1-1 Kerberos technology flow

The Kerberos protocol flow, as shown in the figure, is as follows:

1. The first exchange takes place between a client and the authentication server. The authentication server authenticates the user (for example, by validating the user ID and password).

   After a successful login, the authentication server obtains the user’s secret keys and returns a TGT that is used to get the credentials to grant access.

2. Upon receiving the TGT, the client sends a request (containing the TGT) for a service ticket to the TGS. The TGS authenticates the TGT and then returns a service ticket to the client.

3. The client now has a service ticket that allows the client to communicate with the server that is providing a service that the client wants to use. The application server can verify the service ticket without contacting the KDC.

Kerberos token

A *Kerberos token*, referred to as the *Kerberos authentication token KRBAuthnToken*, is created when the client authenticates with WebSphere. If a client sends the delegate Kerberos credential as part of the authentication request, then the KRBAuthnToken includes the client delegate Kerberos credential. Otherwise, the KRBAuthnToken includes the Kerberos principal and the realm name that the client is using to authenticate.

Kerberos and Microsoft Windows

Kerberos is implemented in Microsoft Windows Server 2000 and later. The implementation of Kerberos on a Windows server is composed the Kerberos service and Active Directory, which is Microsoft’s implementation of the LDAP protocol.

When a user logs in to the Windows domain, the log on program captures the information that the user enters and transmits that information to the computer’s local security authority. The local security authority is a Windows component that authenticates users to the local system. This local security authority then communicates with the network’s KDC to receive TGTs and service tickets so that this user can access Kerberos services in the Windows domain.

Kerberos on Windows Server platforms use Active Directory for all information regarding principals on the Kerberos network. The encryption key used in communicating with principals is stored in the Active Directory database in the user’s profile. Active Directory plays the role of an LDAP server on a Windows server and also is used as the KDC database.

1.5 SSO with SPNEGO and Kerberos

SPNEGO is used when a client application wants to authenticate to a remote server, but neither the server nor the client end is sure which authentication protocol the other supports. Similar to Microsoft, WebSphere offers SPNEGO support, which allows SSO between Microsoft and WebSphere Web-based applications. Many customers use SPNEGO as an SSO solution between the Microsoft Desk Top and WebSphere.

Implementing a Kerberos and SPNEGO solution requires the following environment:

- A Windows domain controller with the KDC enabled, and one client system connected on a domain.
- In a UNIX or Linux environment, a login through Kerberos using the PAM modules, which fetches the TGT.

- A client application supported by Kerberos, such as Firefox, that use service tickets so that the user is not prompted to re-authenticate.
- In the multiple domains scenario, you need to use Kerberos to connect to all domains.

Using SSO with SPNEGO and Kerberos provides the following advantages:

- Establishes an integrated SSO environment with Windows Server 2000 or 2003 using an Active Directory domain.
- Reduces the cost of administering a large number of user IDs and passwords.
- Provides a secure and mutually authenticated transmission of security credentials from the Web browser or Microsoft .NET clients to WebSphere.
- Achieves interoperability with Web services and Microsoft .NET, or Web service applications that use SPNEGO authentication at the transport level.
- Provides an end-to-end SPNEGO to Kerberos solution and preserves the Kerberos credentials from the client using Kerberos authentication support and SPNEGO Web authentication.

When considering this solution, however, keep in mind the following potential disadvantages:

- SSO can introduce a *single point of failure*. It requires continuous availability of a central server. Thus, if the Kerberos server is down, no one can log in. You can mitigate this potential single point of failure by using multiple Kerberos replica servers and fallback authentication mechanisms.
- Kerberos requires that the clocks of the involved hosts are synchronized. The tickets have a time availability period and, if the host clock is not synchronized with the Kerberos server clock, the authentication will fail. The default configuration requires that clock times are no more than 5 minutes apart. In practice, Network Time Protocol daemons can keep the host clocks synchronized.
- The administration protocol is not standardized and differs between Kerberos implementations.
- Because the secret keys for all users are stored on the central server, a compromise of that server compromises all users' secret keys. The KDC system needs to be secured and protected.
- A compromised client can allow unauthorized access to applications.

WebSphere Application Server V7 includes the following features, which provide enhanced use of SPNEGO for SSO:

- SPNEGO Web authentication and filters can be configured using the administrative console.
- Dynamic reload of SPNEGO is provided without the need to stop and restart the application server.
- Fallback to an application login method is provided if the SPNEGO Web authentication fails.
- SPNEGO can be customized at the WebSphere security domain level.

1.6 Kerberos with other authentication protocols

WebSphere Application Server V7 supports Lightweight Third Party Authentication (LTPA) and Kerberos. WebSphere is designed to support both of these mechanisms simultaneously. Applications that use LTPA and applications that use Kerberos or SPNEGO can run together in WebSphere.

You can continue to use LTPA to provide an SSO solution. For applications that exist outside of WebSphere and that support the Kerberos authentication mechanism, you can use Kerberos to achieve SSO. In many respects, Kerberos complements the WebSphere LTPA solution. The LTPA solution has been offered since WebSphere Application Server V4.0.

1.7 Web services security using Kerberos

Web services security (WSS) provides a general-purpose mechanism to associate *security tokens* with messages for message authentication and protection. A security token represents a set of claims made by a client that might include a name, password, identity, key, certificate, group, privilege, and so on. A security token is embedded in the SOAP message within the SOAP header. Then, the security token within the SOAP header is propagated from the message sender to the intended message receiver. On the receiving side, the WSS handler authenticates the security token and sets up the caller identity on the running thread.

WSS does not require a specific type of security token. It is designed to be extensible and supports multiple security token formats to accommodate a variety of authentication mechanisms. For example, a client might provide proof of identity and proof of a particular business certification.

The security token usage for WSS is defined in separate profiles, such as the *Username* token profile and the *X.509* token profile. Similar functions can be performed with the Kerberos token.

1.7.1 Kerberos Token Profile 1.1 specification

The OASIS Web Services Security (WSS) Kerberos Token Profile 1.1 specification describes the use of Kerberos tokens with respect to the WSS: SOAP Message Security specification. This specification defines how to encode Kerberos tickets and attach them to SOAP messages. It also specifies how to add signatures and encryption to the SOAP message, in accordance with the WSS: SOAP Message Security specification, which uses and references the Kerberos tokens.

For interoperability concerns and for some security concerns, the specification is limited to using the AP-REQ packet (service ticket and authenticator) that is defined by Kerberos as the Kerberos token. This token allows a service to authenticate the ticket and interoperate with existing Kerberos implementations.

Note: Explaining how the AP-REQ is obtained is out of scope of this book, as are scenarios that involve other ticket types and user-to-user interactions.

The WSS Kerberos Token Profile 1.1 specification is available at:

http://www.oasis-open.org/committees/download.php/16788/wss-v1.1-spec-os-KerberosTokenProfile.pdf

The Kerberos token includes the following security for Web services messages:

- Provides a secure third-party authentication mechanism that Web services applications can use to send identity and protect messages.
- Allows interoperability between Web services vendors because Web services Kerberos support is standardized in the WSS Kerberos Token Profile 1.1 specification.
- Includes rich security features such as authentication, message integrity and confidentiality, and delegation.

1.7.2 Support in WebSphere Application Server V7

WebSphere Application Server V7 supports the WSS Kerberos Token Profile 1.1 specification. The following Kerberos-related functions are also supported by Web services in WebSphere Application Server:

- Client programming models for JAX-WS applications with WSS APIs
- Interoperability with Web Services Enhancements (WSE) Version 3.5 and Windows Communication Foundation (WCF) Version 3.5 for Microsoft.NET
- Recovery of Web services message security tokens for JAX-WS applications
- Kerberos token profile enablement
- Integration with the base security for the application server
- Kerberos token generation for the client and service
- Kerberos consumption at the service
- Clustering and high-availability for JAX-WS applications
- Kerberos token profile configuration of authentication and message protection for JAX-WS applications
- Integration in a single realm with either a Microsoft, AIX or z/OS operating system KDC
- Kerberos token profile configuration of authentication for JAX-RPC applications

1.7.3 Kerberos configuration models for Web services

The WebSphere Application Server V7 configuration model for the WS-SecurityKerberos token takes advantage of existing frameworks. You can configure Web services Kerberos using any of the following methods:

- Use policy sets and bindings to enable the Kerberos token profile for JAX-WS applications.
- Use deployment descriptors and bindings to enable the Kerberos token profile for JAX-RPC applications.
- The use of WSS APIs for JAX-WS applications.

JAX-WS configuration model

For JAX-WS applications, the WebSphere Application Server client configuration model uses the policy set and takes advantage of a custom policy set for the Kerberos token. The WSS policy is the security policy that is used to secure the application messages.

Using the WebSphere administrative tools, you can specify the Kerberos token type, message signing, and message encryption by using an existing custom policy set. Kerberos token generation and consumption includes the Kerberos token generation for unmanaged JAX-WS clients.

The JAX-WS programming model also provides capabilities to enable the Kerberos token profile and identity assertion by configuring the Kerberos token using policy sets, WSS APIs, and administrative command scripts.

JAX-RPC configuration model

JAX-RPC applications are configured using a deployment model. The deployment descriptor specifies the custom token to use for the Kerberos token. A JAX-RPC client can generate the specified Kerberos token. A JAX-RPC Web service can successfully authenticate the Kerberos token using a custom or default Kerberos identity mapping login module.

API configuration model

A set of application programming interfaces (APIs) is provided by WebSphere Application Server. To successfully use these APIs, application developers must have knowledge about the OASIS WSS Kerberos Token Profile 1.0 and 1.1 specifications. When you use these APIs, the application server assumes that a policy set is not attached to the client resources; however, a warning is still issued when the application server detects any policy set information.

For JAX-WS client applications, the APIs include and enforce WSS policy for the Kerberos token, which is based on the OASIS WSS Kerberos Token Profile. To enable the OASIS WSS Kerberos Token Profile with the policy set, you must first configure the WSS policy and the binding files with the custom token.

For JAX-RPC applications, APIs for WSS are not provided. You must use the deployment descriptor to specify the custom token to use the Kerberos token. You can use the custom token panels within an assembly tool, such as Rational Application Developer, to configure the deployment information.

1.8 High-level look at the scenarios

The remaining chapters in this book focus on a set of common scenarios to demonstrate how to use the Kerberos technology with WebSphere Application Server. These scenarios include solutions from other vendors, such as Microsoft Active Directory and Microsoft KDC.

We successfully tested each of these scenarios as part of writing this book, based on certain patch levels and version of these products; however, IBM does not intend to become the authoritative source on how to configure these solutions from other vendors. Our goal is to demonstrate the possibilities and to show how the scenarios worked for us in our testing.

This book outlines what we did to get the scenarios working, based on the version and patch level that we used. The scenarios cover how to configure WebSphere when using a KDC from Microsoft, AIX, and z/OS and cover any special considerations when using these products.

Directory conventions: In the scenarios, directories are referred to using the following conventions:

- *app_server_root*

  Default installation directories for WebSphere Application Server.
- *profile_root*

  Default profile directories.
- *dmgr_profile_root*

  The *profile_root* directory for the deployment manager.

2

Setting up a KDC on a z/OS system

This chapter explains how to configure a key distribution center (KDC) on a z/OS system that is specific to the System Authorization Facility (SAF) registry. It includes the steps that you run on the KDC as part of the Kerberos setup and management.

This chapter includes the following topics:

- Introduction to the KDC on a z/OS system
- Creating the KDC on a z/OS system
- KDC administrative management tasks
- Validating the setup
- Enabling KDC trace

2.1 Introduction to the KDC on a z/OS system

Kerberos V5 is implemented in z/OS security capabilities and includes the Kerberos service and the registry database (either SAF or NDBM, New database manager). For this scenario, we use the SAF registry with Resource Access Control Facility (RACF®).

In this chapter, we explain how to set up and administer a new KDC on a z/OS system using the RACF registry as the Kerberos registry. The setup includes basic RACF commands to get the KDC running, as well as helpful modification commands to better work with WebSphere Application Server or other products.

To set up a KDC on the z/OS system, you need an understanding of z/OS administration. The setup requires the following environment:

- A z/OS v1r9 system that supports AES128 and AES256 encryption types

  For AES encryption types, you must download and install unrestricted IBM software development kit (SDK) policy files. For instructions about how to download and install the new policy files, see:

  `http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tsec_egs.html`

- WebSphere Application Server for z/OS v7.0.0.5 installed on the z/OS system

  You use the Java commands that are installed with WebSphere to create and verify the Kerberos keytab file.

2.2 Creating the KDC on a z/OS system

This section describes how to create and start a basic KDC on a z/OS system with RACF. Make sure that the z/OS user ID has SPECIAL authority to issue RACF commands and superuser authority to the file system. Be sure to review the commands carefully and understand information that is specific to the individual system (paths, users, and so forth) and information that is general.

2.2.1 Creating the KDC on the z/OS system using RACF commands

You use a TSO (Time Sharing Option) session to submit commands to create the KDC on the z/OS system. You can include these commands in a batch job to make sure that everything is correct before you execute the job.

To create the KDC on the z/OS system, follow these steps.

1. Use the Interactive System Productivity Facility (ISPF) command shell to run RACF commands:
   a. Start a TSO session to the z/OS system and log in to the ISPF.
   b. Select the option for the "ISPF Command Shell."
2. Create the SKRBKDC user that will own the KDC STARTED task, using the following command:

```
ADDUSER SKRBKDC DFLTGRP(SYS1) NOPASSWORD OMVS(UID(0)
PROGRAM('/bin/sh') HOME('/etc/skrb/home/kdc'))
```

3. Activate the APPL class:

```
SETROPTS CLASSACT(APPL) RACLIST(APPL)
```

4. Define the SKRBKDC application in the APPL class:

```
RDEFINE APPL SKRBKDC UACC(READ)
```

5. Refresh the APPL class:

```
SETROPTS RACLIST(APPL) REFRESH
```

6. Activate the PTKTDATA class:

```
SETROPTS CLASSACT(PTKTDATA) RACLIST(PTKTDATA)
```

7. Define a key to mask values in RACF for the SKRBKDC application:

```
RDEFINE PTKTDATA SKRBKDC UACC(NONE)
SSIGNON(KEYMASKED(3734343237343131))
```

8. Refresh the PTKTDATA class:

```
SETROPTS RACLIST(PTKTDATA) REFRESH
```

9. Define the IRR.RUSERMAP profile in the FACILITY class with READ access and refresh the FACILITY class:

```
RDEFINE FACILITY IRR.RUSERMAP UACC(READ)
SETROPTS RACLIST(FACILITY) REFRESH
```

10. Define the STARTED tasks for SKRBKDC and refresh the STARTED class:

```
RDEFINE STARTED SKRBKDC.** STDATA(USER(SKRBKDC))
RDEFINE STARTED SKRBWTR.** STDATA(USER(SKRBKDC))
SETROPTS RACLIST(STARTED) REFRESH
```

11. Define the KERBDFLT RACF REALM for the KDC. The REALM must be KERBDFLT, but you must customize the value for KERBNAME to your system's domain. Set the password and values for ticket life (in seconds).

```
RDEFINE REALM KERBDFLT KERB(KERBNAME(ITSO.IBM.COM)
PASSWORD(security) MINTKTLFE(15) DEFTKTLFE(36000)
MAXTKTLFE(86400))
```

Note: Remember the PASSWORD used for future configuration modifications and the REALM name.

12. Refresh the REALM class:

```
SETROPTS RACLIST(REALM) REFRESH
```

2.2.2 Starting the KDC on the z/OS system

To start the KDC on the z/OS system, you must have a procedure (`proc`) for starting the Kerberos SKRBKDC started task. In this section, we assume a `proc` does not exist and, thus, copy a `proc` to the system's PROCLIB. Also, copy the `envar` and `krb5.conf` files to the correct location. You can find examples of these commands in the `/usr/lpp/skrb/examples` directory.

To start the KDC on the z/OS system, follow these steps:

1. Use the ISPF 3.4 option to copy the sample `proc` in EUFV.SEUVFSAM(SKRBKDC) to the system's PROCLIB:
   a. Start a TSO session to the z/OS system and log in to ISPF.
   b. Select the ISPF 3.4 option to open the Data Set List Utility.
   c. For the Dsname Level, enter `EUVF.SEUVFSAM`.
   d. When the EUVF.SEUVFSAM data set is found, type `e` to edit this data set.
   e. Find SKRBKDC in this data set and type `c` to copy.
   f. Enter the data set name to which you want to copy the `proc`. In this scenario, we copied it to the system's USER.PROCLIB.
2. From a command line, copy the `envar` and `krb5.conf` files to the following locations if they do not exist there already:

```
cp /usr/lpp/skrb/examples/skrbkdc.envar /etc/skrb/home/kdc/envar
cp /usr/lpp/skrb/examples/krb5.conf /etc/skrb/krb5.conf
```

On z/OS, the default location of the Kerberos configuration file is /etc/skrb. You can create a symbolic link so that WebSphere Application Server can find the Kerberos configuration file in the /etc/krb5 directory on z/OS:

```
ln -s /etc/skrb /etc/krb5
```

3. Modify the krb5.conf file to specify the Kerberos realm name and the KDC host name. The realm in the krb5.conf file must be the same as the realm that you specified in the RDEFINE REALM command (described in step 11 of 2.2.1, "Creating the KDC on the z/OS system using RACF commands" on page 14).

   The krb5.conf file in Example 2-1 sets the realm to ITSO.IBM.COM and the host name to wtsc60.itso.ibm.com. These updated entries are shown in bold font in the example.

Example 2-1 The krb5.conf file

```
;-------------------------------------------------------------------;
;  Sample Kerberos configuration file                               ;
;                                                                   ;
;  Copy this file to /etc/skrb/krb5.conf and then tailor it for     ;
;  your Kerberos configuration                                      ;
;                                                                   ;
;  Do not enable an encryption type unless all of the systems in the ;
;  realm have support.  To change the enabled encryption types for  ;
;  tickets, you must set the SKDC_TKT_ENCTYPES environment variable ;
;  in /etc/skrb/home/kdc/envar.                                     ;
;-------------------------------------------------------------------;

[libdefaults]

default_realm = ITSO.IBM.COM
kdc_default_options = 0x40000010
use_dns_lookup = 0

; Enable DES encryption types (AES, DES3 and DESD are disabled)
default_tkt_enctypes = des-cbc-md5,des-cbc-md4,des-cbc-crc
default_tgs_enctypes = des-cbc-md5,des-cbc-md4,des-cbc-crc
; Enable DES3 and DES encryption types (AES and DESD are disabled)
;default_tkt_enctypes = des3-cbc-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
;default_tgs_enctypes = des3-cbc-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
; Enable all encryption types
;default_tkt_enctypes =
aes256-cts-hmac-sha1-96,aes128-cts-hmac-sha1-96,des3-cbc-sha1,des-hmac-sha1
,
des-cbc-md5,des-cbc-md4,des-cbc-crc
;default_tgs_enctypes =
aes256-cts-hmac-sha1-96,aes128-cts-hmac-sha1-96,des3-cbc-sha1,des-hmac-sha1
,
des-cbc-md5,des-cbc-md4,des-cbc-crc

[realms]

ITSO.IBM.COM = {
    kdc = wtsc60.itso.ibm.com:88
    kpasswd_server = wtsc60.itso.ibm.com:464
}
[domain_realm]
.itso.ibm.com = ITSO.IBM.COM
```

4. Run the following command from System Display and Search Facility (SDSF) to start the z/OS KDC:

   ```
   START SKRBKDC
   ```

   Example 2-2 shows the console output for the START command.

   Example 2-2 START SKRBKDC output

   ```
   START SKRBKDC
   $HASP100 SKRBKDC  ON STCINRDR
   IEF695I START SKRBKDC  WITH JOBNAME SKRBKDC  IS ASSIGNED TO USER
   SKRBKDC, GROUP SYS1
   $HASP373 SKRBKDC  STARTED
   EUVF04022I Security server start command processed.
   ```

5. If you need to stop the KDC on the z/OS system, run the following command from SDSF:

   ```
   STOP SKRBKDC
   ```

2.3 KDC administrative management tasks

To configure the KDC on a z/OS system for Kerberos, you need to identify a z/OS system for the WebSphere Application Server. You also need to identify users for the WebSphere registry. For the examples that we show in this section, the host name of the z/OS system where WebSphere is installed is *wtso60.itso.ibm.com*.

We discuss the following tasks in this section:

- Creating the Kerberos service principal name
- Creating a Kerberos keytab file
- Creating users for Kerberos in RACF
- Changing Kerberos ticket life
- Setting the encryption type

2.3.1 Creating the Kerberos service principal name

Create a RACF ID with a Kerberos segment for the WebSphere Application Server. The RACF ID and Kerberos segment must be unique. The Kerberos segment KERNAME must be `<`*`component name`*`>/<`*`fully qualified host name`*`>`. Kerberos does not require a specific the component name, but SPNEGO requires the component name to be HTTP. For example, the service principal name (SPN) needs to be `HTTP/<`*`fully qualified host name`*`>`.

The TSO commands in Example 2-3 use the RACF ID `krbid` with the principal `WAS101/wtsc60.itso.ibm.com` for Kerberos and `krbid2` with the principal `HTTP/wtsc60.itso.ibm.com` for SPNEGO.

Example 2-3 TSO commands

```
ADDUSER krbid DFLTGRP(SYS1) PASSWORD(SECURITY)
ALTUSER krbid PASSWORD(SECURITY) NOEXPIRED
KERB(KERBNAME(WAS101/wtsc60.itso.ibm.com) ENCRYPT(DES NODES3 NODESD))
ADDUSER krbid2 DFLTGRP(SYS1) PASSWORD(SECURITY)
ALTUSER krbid2 PASSWORD(SECURITY) NOEXPIRED
KERB(KERBNAME(HTTP/wtsc60.itso.ibm.com) ENCRYPT(DES NODES3 NODESD))
```

Note: Remember the password that you use for these users so that you can correctly generate the keytab files and future updates to the user.

2.3.2 Creating a Kerberos keytab file

The SPN and keys are stored in a keytab file. The keytab file can contain multiple keys with different encryptions so that a service can decode the Kerberos tokens. Also, the keytab file can contain different SPN and keys to help manage the services for the WebSphere Application Server cell.

You can create the keytab file with a key and can add more keys using the Java **`ktab`** utility. Run the **`ktab -a`** command from the *`app_server_root`*`/java/bin` directory to create a Kerberos keytab file for a specific principal name. The keytab file is created in the directory set by the `default_keytab_name` setting in the `krb5.conf` file. For example, if you set the following line in `krb5.conf`, the keytab file is created in the `/etc/skrb5` directory:

```
default_keytab_name = FILE:/etc/skrb/krb5.keytab
```

The **`ktab -a`** command adds the specified SPN to a keytab file.

For more information about the **`ktab`** command, run **`ktab -help`**.

Example 2-4 shows the **ktab** command for both Kerberos (`WAS101/wtsc60.itso.ibm.com`) and SPNEGO (`HTTP/wtsc60.itso.ibm.com`) service principal names.

Example 2-4 The ktab -a command to create keytab file

```
/usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./ktab -a
WAS101/wtsc60.itso.ibm.com@ITSO.IBM.COM SECURITY
Done!
Service key for principal WAS101/wtsc60.itso.ibm.com@ITSO.IBM.COM saved

/usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./ktab -a
HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM SECURITY
Done!
Service key for principal HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM saved
```

The keytab files are created at `/etc/skrb/krb5.keytab`. You can run **ktab** to verify the Kerberos service principal name or names in the keytab, as shown in Example 2-5.

Example 2-5 Verify SPN with ktab

```
/usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./ktab
2 entries in keytab, name: /etc/skrb/krb5.keytab
        KVNO    Principal
        ----    ---------

        1       WAS101/wtsc60.itso.ibm.com@ITSO.IBM.COM

        1       HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM
```

2.3.3 Creating users for Kerberos in RACF

RACF users cannot use Kerberos by default, even though RACF is the user registry for the KDC. The users need to map a Kerberos principal name to a RACF ID so that those IDs can be used for Kerberos.

To create the RACF ID and a Kerberos segment for that ID, use the following commands:

```
ADDUSER MTADMIN DFLTGRP(SYS1) PASSWORD(pw)
ALTUSER MTADMIN PASSWORD(MTADMIN) NOEXPIRED KERB(KERBNAME(MTADMIN)
ENCRYPT(DES NODES3 NODESD))
```

Note that in this example the KERBNAME is case sensitive (unlike the RACF ID). It does not need to match the RACF ID, and it does not have the eight character length restriction. Also, if the command SETROPTS PASSWORD(MIXEDCASE) is not activated to enable mixed-case passwords, then use the ALTUSER command to convert the password to uppercase.

Note: If mixed-case passwords are not activated, the PASSWORD is converted to UPPERCASE.

You can run these TSO commands for all the users in the WebSphere user registry who you want to map and use Kerberos. You can modify existing RACF users by adding a Kerberos segment with the ALTUSER command.

If the KDC on the z/OS system is set up with cross realm trust with another KDC, then you can map a foreign principal to the RACF ID using the following command:

```
RDEFINE KERBLINK /.../foreign_realm/[foreign-principal_name]
appldata('racfid')
```

In this example, the RACF user `tamt` is mapped to the foreign principal `TAMT` in the foreign realm `KDC2008.ITSO.COM`:

```
RDEFINE KERBLINK /.../KDC2008.ITSO.COM/TAMT appldata('tamt')
```

For more information about the RDEFINE KERBLINK command and other RACF commands, refer to:

http://publibz.boulder.ibm.com/epubs/pdf/ichza780.pdf

2.3.4 Changing Kerberos ticket life

Ticket life determines how long Kerberos tickets are valid. You can change the times to match the security requirements for your system (for example, user tokens are not valid for more then one week) and the user requirement needs (for example, a user does not want to login again to a system every hour). You can also change times to help stop unwanted repeat logins from happening or to require users to log in again to validate the user ID.

The KERBDFLT REALM that you created for the KDC has the values for the ticket life. You can view the KERBDFLT REALM with the TSO command RLIST REALM KERBDFLT KERB NORACF.

Example 2-6 shows the output of this command.

Example 2-6 Output of RLIST REALM KERBDFLT KERB NORACF command

```
CLASS      NAME
-----      ----
REALM      KERBDFLT

KERB INFORMATION
----------------
KERBNAME= ITSO.IBM.COM
MINTKTLFE= 0000000015
MAXTKTLFE= 0000086400
DEFTKTLFE= 0000036000
KEY VERSION= 003
KEY ENCRYPTION TYPE= DES DES3 DESD AES128 AES256
```

The values for minimum (`MINTKTLFE`), maximum (`MAXTKTLFE`), and default (`DEFTKTLFE`) are shown in the output. To change the values for these options, run the following commands:

```
RALT REALM KERBDFLT KERB(MINTKTLFE(xxxxx))
RALT REALM KERBDFLT KERB(MAXTKTLFE(xxxxx))
RALT REALM KERBDFLT KERB(DEFTKTLFE(xxxxx))
```

In these commands, *xxxxx* is the time in seconds.

The MAXTKTLFE setting can effect a client program. For example, if the Java KerberosTicket method refresh is used, which updates the Kerberos ticket time automatically if this time has not expired and if the MAXTKTLFE has expired, then the user is required to authenticate again to generate a new Kerberos token.

2.3.5 Setting the encryption type

Choosing the encryption type used for Kerberos tokens can help increase the security of the users credentials. Review the encryptions that the KDC supports (each version can support different encryption types) and the supported service product, in this case WebSphere Application Server. Refer to the supported encryption types that are available in the information center:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tsec_kerb_create_conf.html

When changing the encryption, you might need to make multiple modification to support the new encryption. You need to ensue that:

- KDC supports the encryption
- User supports the encryption
- Keytab encryption update

KDC encryption modifications

To add or remove encryption types for the KDC on z/OS, update the `envar` file SKDC_TKT_ENCTYPES setting:

```
SKDC_TKT_ENCTYPES=aes256-cts-hmac-sha1-96,aes128-cts-hmac-sha1-96,de
s3-cbc-sha1,des-hmac-sha1,des-cbc-md5,des-cbc-md4,des-cbc-crc
```

Then, recycle the KDC to load the new encryption list.

User encryption modifications

When the Kerberos segment is created for a user, the encryption types that the ID uses are listed. You can change the encryption types using the following command:

```
ALTUSER MTADMIN PASSWORD(MTADMIN) NOEXPIRED KERB(KERBNAME(MTADMIN)
ENCRYPT(DES DES3 DESD))
```

Keytab encryption modifications

Update the service with the encryption type that is used to communicate with the KDC by updating the keytab file and the Kerberos configuration file. To set the encryption type for the keytab file, modify the following lines in the `krb5.conf` file:

```
default_tkt_enctypes = xxxxx
default_tgs_enctypes = xxxxx
```

In these lines, *xxxxx* is the encryption type. After you set the encryption type in the `krb5.conf` file, create the keytab with the **`ktab -a`** command. Note that you can update an existing keytab file with more than one encryption type for the same SPN. To verify that the correct encryption type is set, you can view it with the **`klist -k -e`** command, as shown in Example 2-7.

Example 2-7 The klist command to show encryption type

```
/usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./klist -k -e

Key table: /etc/skrb/krb5.keytab
Number of entries: 2

[1] principal: WAS101/wtsc60.itso.ibm.com@ITSO.IBM.COM
        KVNO: 1
        Encryption type: DES CBC mode with MD5

[2] principal: HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM
        KVNO: 1

        Encryption type: DES CBC mode with MD5
```

2.4 Validating the setup

To verify the KDC setup on the z/OS system, follow these steps:

1. Run the following command to list information about the SKRBKDC user that owns the STARTED task:

   ```
   LISTUSER SKRBKDC OMVS NORACF
   ```

   Example 2-8 shows the output of this command.

   Example 2-8 Lists SKRBKDC user information

   ```
   USER=SKRBKDC

   OMVS INFORMATION
   ----------------
   UID= 0000000000
   HOME= /etc/skrb/home/kdc
   PROGRAM= /bin/sh
   CPUTIMEMAX= NONE
   ASSIZEMAX= NONE
   FILEPROCMAX= NONE
   PROCUSERMAX= NONE
   THREADSMAX= NONE
   MMAPAREAMAX= NONE
   ```

2. Run the following command to show the realm name of the KDC, the values for ticket life, and the supported encryption types:

   ```
   RLIST REALM KERBDFLT KERB NORACF
   ```

   Example 2-9 shows the output of this command.

   Example 2-9 Lists REALM Kerberos information

   ```
   CLASS      NAME
   -----      ----
   REALM      KERBDFLT
   ```

```
KERB INFORMATION
----------------
KERBNAME= ITSO.IBM.COM
MINTKTLFE= 0000000015
MAXTKTLFE= 0000086400
DEFTKTLFE= 0000036000
KEY VERSION= 003
KEY ENCRYPTION TYPE= DES DES3 DESD AES128 AES256
```

3. Run the RLIST KERB command to show the RACF user that is associated with the service principal name (if you know the principal name):

   ```
   RLIST KERB WAS101/wtsc60.itso.ibm.com
   ```

   The value for the APPLICATION DATA field in the command output is the RACF user. In Example 2-10, the RACF user `KRBID` is mapped to the principal name `WAS101/wtsc60.itso.ibm.com`.

Example 2-10 Lists RACF ID mapped to principal

```
CLASS      NAME
-----      ----
KERBLINK   WAS101/wtsc60.itso.ibm.com

LEVEL  OWNER      UNIVERSAL ACCESS  YOUR ACCESS  WARNING
-----  --------   ----------------  -----------  -------
00    MTADMIN           NONE              NONE    NO

INSTALLATION DATA
-----------------
NONE

APPLICATION DATA
----------------
KRBID

AUDITING
--------
FAILURES(READ)

NOTIFY
------
NO USER TO BE NOTIFIED
```

4. Run the following command to show the Kerberos segment for a RACF user. If no Kerberos segment exists for the RACF user, no Kerberos information is returned.

   ```
   LISTUSER MTADMIN KERB NORACF
   ```

   In Example 2-11, the RACF user and Kerberos segment (`MTADMIN`) are the same.

   Example 2-11 Lists Kerberos segment for RACF user

   ```
   USER=MTADMIN

   KERB INFORMATION
   ----------------
   KERBNAME= MTADMIN
   KEY FROM= PASSWORD
   KEY VERSION= 002
   KEY ENCRYPTION TYPE= DES NODES3 NODESD AES128 AES256
   ```

5. Run the following command from the *`app_server_root`*`/java/bin` directory to verify the service principal by authenticating to the KDC and then create a Kerberos cache ticket:

   ```
   ./kinit -k WAS101/wtsc60.itso.ibm.com
   ```

   Example 2-12 shows the output for this command.

   Example 2-12 The kinit command output for SPN

   ```
   /usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./kinit -k
   WAS101/wtsc60.itso.ibm.com
   Done!
   New ticket is stored in cache file /krb5cc_MTADMIN
   ```

   You can also run the following command to verify a RACF user with a Kerberos segment:

   ```
   ./kinit MTADMIN
   ```

   You must specify the password for the RACF user.

   Example 2-13 shows the output from this command.

   Example 2-13 The kinit command output for RACF user

   ```
   /usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./kinit MTADMIN
   Password for MTADMIN@ITSO.IBM.COM:

   Done!
   New ticket is stored in cache file /krb5cc_MTADMIN
   ```

6. Run the following command from the *app_server_root*/java/bin directory to show the principal and encryption type information in the Kerberos cache:

   ```
   ./klist -e
   ```

   Example 2-14 shows the output from this command.

   Example 2-14 The klist -e command output

   ```
   /usr/lpp/zWebSphereMT/V7R0/java/J6.0/bin>./klist -e

   Credentials cache: /krb5cc_MTADMIN
   Default principal: MTADMIN@ITSO.IBM.COM
   Number of entries: 1

   [1] Service principal: krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
           Valid starting: Thursday, July 9, 2009 at 5:34:13 PM
           Expires: Friday, July 10, 2009 at 3:34:13 AM

           Encryption type: DES CBC mode with MD5
   ```

2.5 Enabling KDC trace

To troubleshoot the KDC on the z/OS system, you can enable debug trace using one of the following methods:

- Modify the envar file to write debug messages to a file, which requires you to restart the KDC for the changes to take effect
- Use the FORMAT or MODIFY commands from the SDSF panels, which enables debug trace dynamically.

2.5.1 Enabling debug with the envar file

To enable debug using the envar file to write the debug messages to a file, follow these steps:

1. Modify the /etc/skrb/home/kdc/envar file and include the following settings:

   ```
   _EUV_SVC_MSG_LOGGING=STDOUT_LOGGING
   _EUV_SVC_MSG_IDENTITY=SKRBKDC
   _EUV_SVC_MSG_FACILITY=AUTH
   _EUV_SVC_STDOUT_FILENAME=/<path>/<filename>
   _EUV_SVC_DBG_MSG_LOGGING=1
   _EUV_SVC_DBG=*.8
   ```

2. Set the path and file name for `_EUV_SVC_STDOUT_FILENAME` to set the location of the file to write debug messages. If `_EUV_SVC_STDOUT_FILENAME` is not specified, the debug messages are written to STDOUT.
3. Because these settings are processed during the initialization of the KDC, you must restart the KDC for the changes to take effect.
4. To disable debug, set `_EUV_SVC_DBG_MSG_LOGGING=0`, and restart the KDC.

You can customize the KDC debug settings, such as the level of debug for each subcomponent and the message logging type. For more information about setting environment variables in the `envar` file, refer to:

http://publib.boulder.ibm.com/infocenter/zos/v1r9/index.jsp?topic=/com.ibm.zos.r9.euvfa00/euvfad0021004132.htm

2.5.2 Enabling debug dynamically

This section explains how to enable debug using the FORMAT or MODIFY commands to set debug dynamically. Because the parameters and usage of both commands are the same, our example shows only the FORMAT command.

The syntax for the FORMAT command is `F SKRBKDC,`*`parameters`*. You can run the following commands from ISPF option SD:

- To turn debug on, use the following command:

```
F SKRBKDC,DEBUG ON
```

- To turn debug off, use the following command:

```
F SKRBKDC,DEBUG OFF
```

- To set the debug levels, use the following command:

```
F SKRBKDC,DEBUG subcomponent.level[,subcomponent.level, ...]
```

  To include all subcomponents, specify an asterisk (*). The level must be an integer from 0-9, where 0 is no debug messages and 9 has the most debug messages.

For more information about the FORMAT or MODIFY commands, refer to:

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/download/euvb3a50.pdf

3

Configuring IBM Network Authentication Service KDC on AIX

This chapter provides information about how to configure Kerberos KDC on AIX using IBM Network Authentication Service (IBM NAS), which is the IBM offering of Kerberos Version 5 implementation. The chapter also covers briefly the steps to perform Kerberos-related tasks (such as generating keytab files, creating principals, and so forth).

This chapter explains how to configure IBM NAS KDC on AIX with IBM Tivoli® Directory Server. It includes the following topics:

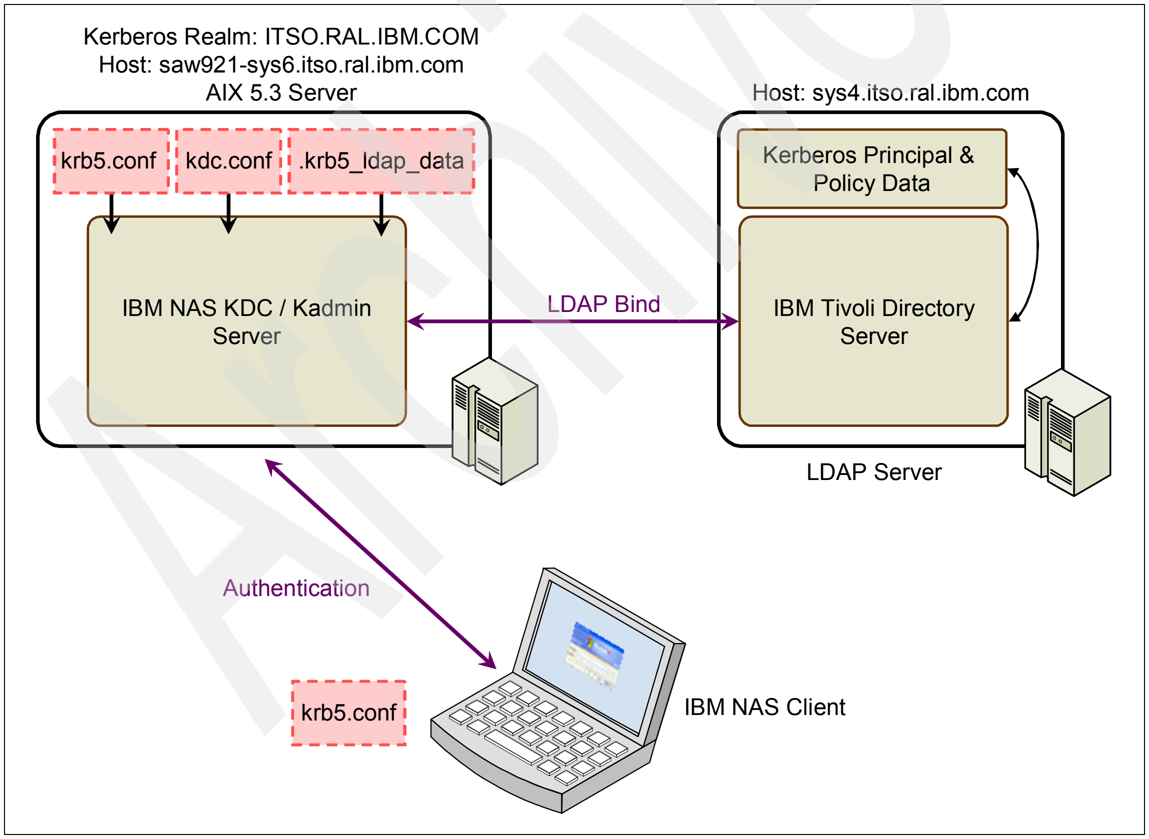
- Introduction to this chapter
- Configuring the IBM NAS KDC on AIX
- Additional Kerberos tasks

3.1 Introduction to this chapter

The IBM Network Authentication Service (IBM NAS) is the IBM offering of Kerberos. It is based on IETF RFC 1510 for Kerberos V5 network authentication protocol. It supports storing Kerberos data in an LDAP directory server as well as the local file system (an existing database).

This chapter covers the steps to set up a KDC on AIX using an LDAP directory to store Kerberos data. It also describes how to perform additional administration tasks (for example, creating principal, generating keytab file, and so forth), to help administrators in configuring various applications (for example WebSphere Application Server) with this KDC.

Figure 3-1 illustrates the IBM NAS KDC configuration on AIX for the scenarios that we describe in this book. The KDC and LDAP directory are placed on two different computers to prevent having a single point of failure.



Kerberos Realm: ITSO.RAL.IBM.COM
Host: saw921-sys6.itso.ral.ibm.com
AIX 5.3 Server
krb5.conf
kdc.conf
.krb5_ldap_data
IBM NAS KDC / Kadmin Server
LDAP Bind
Host: sys4.itso.ral.ibm.com
Kerberos Principal & Policy Data
IBM Tivoli Directory Server
LDAP Server
Authentication
krb5.conf
IBM NAS Client

Figure 3-1 High-level view of the IBM NAS KDC configuration on AIX

We use the following setup and the configuration in this chapter:

- Kerberos Realm: `ITSO.RAL.IBM.COM`
  - Kerberos Administrator: `admin/admin`
  - Kerberos Administration password: `<krb_passwd>`
- IBM NAS 1.4 KDC
  - Host name: `saw921-sys6.itso.ral.ibm.com Port: 88`
  - OS: AIX 5.3
- IBM NAS 1.4 Administration Server
  - Host name: `saw921-sys6.itso.ral.ibm.com Port: 749`
  - OS: AIX 5.3
- IBM Tivoli Directory Server 6.1 server
  - Instance Name: `idsinst`
  - Root DN: `cn=root`
  - Password: `<dn_passwd>`
  - Host name: `sys4.itso.ral.ibm.com Port: 389`
  - OS: Windows 2003 Server
- IBM Tivoli Directory Server 6.1 client
  - Host name: `saw921-sys6.itso.ral.ibm.com`
  - OS: AIX 5.3

For the scenarios that we describe in this book, the Tivoli Directory Server 6.1 server instance on Windows is assumed to be configured and running. The Tivoli Directory Server 6.1 client on AIX is also installed.

The IBM NAS package for AIX is distributed with the IBM AIX expansion pack CDs. You can also download it from the AIX Web pack download page:

`https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?source=aixbp`

The package includes client, server, and toolkit file sets. For a Kerberos KDC, both the server and client file sets are required. (The client file set is prerequisite for the server file set.)

As of publication of this book, the latest version is IBM NAS Version 1.4.0.10. We use this version throughout this book for Kerberos KDC on AIX.

3.2 Configuring the IBM NAS KDC on AIX

This section describes the steps to configure the IBM NAS KDC on AIX using the LDAP directory database.

Note: When the IBM NAS KDC is configured on a system, the IBM NAS client is configured automatically.

Before you begin, make sure that the IBM Tivoli Directory Server client on the AIX system can communicate with the IBM Tivoli Directory Server using the **`ldapsearch`** command. If there is no communication between IBM Tivoli Directory Server and the client, you need to correct the problem before proceeding.

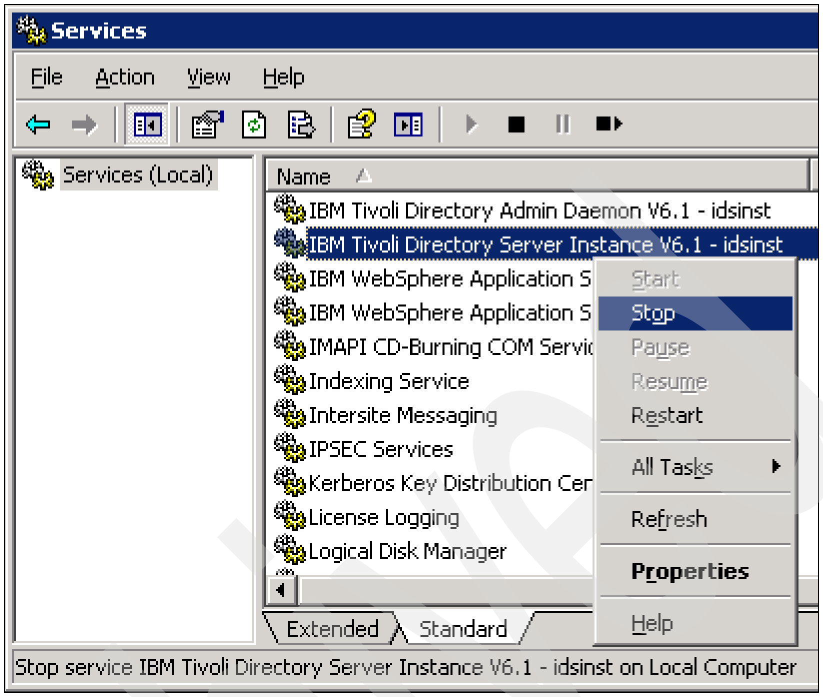
To configure the IBM NAS KDC on AIX, you need to complete the following basic steps:

1. Add the directory suffix in IBM Tivoli Directory Server.
2. Load the IBM NAS Kerberos schema in IBM Tivoli Directory Server.
3. Add Kerberos realm information in IBM Tivoli Directory Server.
4. Configure the IBM NAS KDC on the AIX system.

3.2.1 Add the directory suffix in IBM Tivoli Directory Server

To store Kerberos data in an IBM Tivoli Directory Server directory, you need to add a directory suffix. The *suffix* is a branch in the directory tree under which all the Kerberos principal and policy information is stored. To add a suffix, follow these steps:

1. Stop the IBM Tivoli Directory Server instance:
   a. Select **Start** → **Run** and enter the following command:
      ```
      services.msc /s
      ```
   b. In the Services window, right-click **IBM Tivoli Directory Server Instance V6.1 - idsinst** and select **Stop**, as shown in Figure 3-2.


Services
File
Action
View
Help
Services (Local)
Name
IBM Tivoli Directory Admin Daemon V6.1 - idsinst
IBM Tivoli Directory Server Instance V6.1 - idsinst
IBM WebSphere Application S
IBM WebSphere Application S
IMAPI CD-Burning COM Servic
Indexing Service
Intersite Messaging
IPSEC Services
Kerberos Key Distribution Cer
License Logging
Logical Disk Manager
Start
Stop
Pause
Resume
Restart
All Tasks
Refresh
Properties
Help
Extended
Standard

Stop service IBM Tivoli Directory Server Instance V6.1 - idsinst on Local Computer

Figure 3-2 Stopping the IBM Tivoli Directory Server instance

2. After stopping the IBM Tivoli Directory Server, add the suffix using the following command at a command prompt:

```
idscfgsuf -I idsinst -s "ou=raleigh,o=ibm,c=us" -n
```

This command provides the details of what is added, as shown in Example 3-1.

Example 3-1 Adding the suffix in IBM Tivoli Directory Server

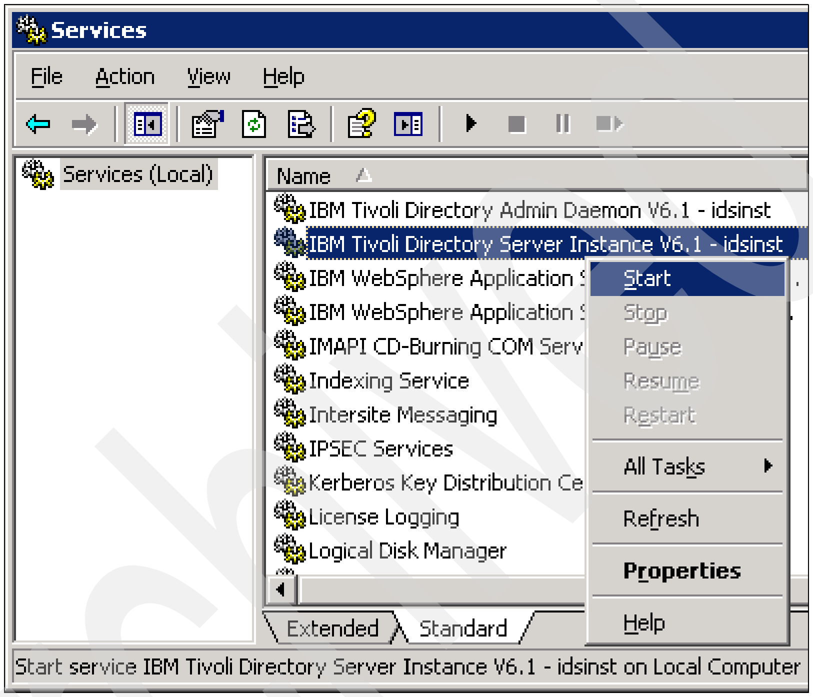
```
c:\> c:\LDAP\V6.1\sbin\idscfgsuf -I idsinst -s
"ou=raleigh,o=ibm,c=us" -n
You have chosen to perform the following actions:
GLPCSF007I Suffix 'ou=raleigh,o=ibm,c=us' will be added to the
configuration file of the directory server instance 'idsinst'.
GLPCSF004I Adding suffix: 'ou=raleigh,o=ibm,c=us'.
GLPCSF005I Added suffix: 'ou=raleigh,o=ibm,c=us'.
```

3. Now, restart the IBM Tivoli Directory Server instance:

   a. Select **Start** → **Run** and enter the following command:

```
services.msc /s
```

   b. In the Services window, right-click **IBM Tivoli Directory Server Instance V6.1 - idsinst** and select **Start**, as shown in Figure 3-3.



IBM Tivoli Directory Admin Daemon V6.1 - idsinst
IBM Tivoli Directory Server Instance V6.1 - idsinst
IBM WebSphere Application S
IBM WebSphere Application S
IMAPI CD-Burning COM Serv
Indexing Service
Intersite Messaging
IPSEC Services
Kerberos Key Distribution Ce
License Logging
Logical Disk Manager

Start service IBM Tivoli Directory Server Instance V6.1 - idsinst on Local Computer

Figure 3-3 Starting the IBM Tivoli Directory Server instance

4. To confirm that the suffix is added properly, issue the following command from the AIX system:

```
ldapsearch -h sys4.itso.ral.ibm.com -b "" -s base "objectclass=*"
namingcontexts
```

This command displays all the suffixes that are available on the specified IBM Tivoli Directory Server instance, including the newly added suffix, as shown in Example 3-2.

Example 3-2 Viewing the suffix after addition

```
[saw921-sys6][/]# ldapsearch -h sys4.itso.ral.ibm.com -b "" -s base
"objectclass=*" namingcontexts

namingcontexts=CN=SCHEMA
namingcontexts=CN=LOCALHOST
namingcontexts=CN=IBMPOLICIES
namingcontexts=O=SAMPLE
namingcontexts=OU=RALEIGH,O=IBM,C=US
```

3.2.2 Load the IBM NAS Kerberos schema in IBM Tivoli Directory Server

To store Kerberos data in the IBM Tivoli Directory Server, you need to load the Kerberos schema first. This schema specifies how to store the Kerberos data.

The Kerberos schema comes with the IBM NAS file sets as LDAP Data Interchange Format (LDIF) file. For the IBM Tivoli Directory Server, the schema file is installed as `/usr/krb5/ldif/IBM.KRB.schema.ldif`.

You can use the **`ldapmodify`** command to load the schema as follows:

```
ldapmodify -h sys4.itso.ral.ibm.com -D cn=root -w <dn_passwd> -f
/usr/krb5/ldif/IBM.KRB.schema.ldif -v -c
```

This command displays the output of the schema entries that are added in the IBM Tivoli Directory Server. This command might produce warnings that some entries already exist in IBM Tivoli Directory Server. You can safely ignore these warnings.

3.2.3 Add Kerberos realm information in IBM Tivoli Directory Server

After loading the Kerberos schema, you next need to add the Kerberos realm-specific information into IBM Tivoli Directory Server, including the name of the realm and where in the directory this realm is to be stored.

IBM NAS provides a sample LDIF file, `/usr/krb/ldif/realm_add.ldif`, for realm information that you can modify as needed. You can edit the information shown in bold highlighting in Example 3-3.

Note: Notice that the first `dn` entry in Example 3-3 is the name of the suffix that you added earlier. You need to use this same suffix throughout the LDIF file. Also, you need to decide the realm name before you edit the file.

Example 3-3 Sample realm_add.ldif for realm ITSO.RAL.IBM.COM

```
dn: ou=raleigh,o=ibm,c=us
ou: Raleigh
objectclass: organizationalUnit

dn: krbrealmName-V2=ITSO.RAL.IBM.COM, ou=raleigh,o=ibm,c=us
objectclass: KrbRealm-V2
objectclass: KrbRealmExt
krbrealmName-V2: ITSO.RAL.IBM.COM
krbprincSubtree: krbrealmName-V2=ITSO.RAL.IBM.COM,
ou=raleigh,o=ibm,c=us
krbDeleteType: 3
krbMaxFailAuth: 0
krbDisableTimeInterval: 0

dn: cn=principal, krbrealmName-V2=ITSO.RAL.IBM.COM,
ou=raleigh,o=ibm,c=us
objectclass: container
cn: principal

dn: cn=policy, krbrealmName-V2=ITSO.RAL.IBM.COM, ou=raleigh,o=ibm,c=us
objectclass: container
cn: policy
```

After you edit the file, save it, and add the realm information to IBM Tivoli Directory Server using the following **`ldapadd`** command:

```
ldapadd -h sys4.itso.ral.ibm.com -D cn=root -w <dn_passwd>-f
/usr/krb/ldif/realm_add.ldif.raleigh -v
```

This command adds three entries in the IBM Tivoli Directory Server instance, as shown in Example 3-4.

Example 3-4   Adding realm information in IBM Tivoli Directory Server

```
[saw921-sys6][/]# ldapadd -h sys4.itso.ral.ibm.com -D cn=root -w
<dn_passwd> -f /usr/krb/ldif/realm_add.ldif.raleigh -v
ldap_init(sys4.itso.ral.ibm.com, 389)
add ou:
        BINARY (7 bytes) Raleigh
add objectclass:
        BINARY (18 bytes) organizationalUnit
Operation 0 adding new entry ou=raleigh,o=ibm,c=us

add objectclass:
        BINARY (11 bytes) KrbRealm-V2
        BINARY (11 bytes) KrbRealmExt
add krbrealmName-V2:
        BINARY (16 bytes) ITSO.RAL.IBM.COM
add krbprincSubtree:
        BINARY (55 bytes) krbrealmName-V2=ITSO.RAL.IBM.COM,
ou=raleigh,o=ibm,c=us
add krbDeleteType:
        BINARY (1 bytes) 3
add krbMaxFailAuth:
        BINARY (1 bytes) 0
add krbDisableTimeInterval:
        BINARY (1 bytes) 0
Operation 1 adding new entry krbrealmName-V2=ITSO.RAL.IBM.COM,
ou=raleigh,o=ibm,c=us

add objectclass:
        BINARY (9 bytes) container
add cn:
        BINARY (9 bytes) principal
Operation 2 adding new entry cn=principal,
krbrealmName-V2=ITSO.RAL.IBM.COM, ou=raleigh,o=ibm,c=us

add objectclass:
        BINARY (9 bytes) container
add cn:
        BINARY (6 bytes) policy
Operation 3 adding new entry cn=policy,
krbrealmName-V2=ITSO.RAL.IBM.COM, ou=raleigh,o=ibm,c=us
```

3.2.4 Configure the IBM NAS KDC on the AIX system

To configure an IBM NAS KDC, use the following IBM NAS command:

```
/usr/krb5/sbin/config.krb5
```

This comprehensive command performs the complete Kerberos configuration. It creates the Kerberos configuration files, the Kerberos database, and the Kerberos admin and service principal names. It also starts the Kerberos daemons.

The syntax of the command is as follows:

```
/usr/krb5/sbin/config.krb5 -S -r <realm_name> -d <domain_name> -a
<admin_name> -l <fully_qualified_ldap_server_hostname> -u <root_DN>
-p <dn_passwd>
```

The **config.krb5** command prompts you for the Kerberos database master password (two times) and then the Kerberos administrator password (two times). If no errors are encountered, then the Kerberos daemons are started when the command completes.

For our testing, we used the following information:

Realm name:	`ITSO.RAL.IBM.COM`
Kerberos administrator name:	`admin/admin` (default)
Kerberos administrator password:	*<krb_passwd>*
Kerberos database master password:	*<master_passwd>*
Domain name:	`itso.ral.ibm.com`
LDAP server host name:	`sys4.itso.ral.ibm.com`
LDAP server root DN:	`cn=root`
LDAP server root DN password:	*<dn_passwd>*

Thus, we use the following command:

```
/usr/krb5/sbin/config.krb5 -S -r ITSO.RAL.IBM.COM -d
itso.ral.ibm.com -a admin/admin -l sys4.itso.ral.ibm.com -u cn=root
-p <dn_passwd>
```

Example 3-5 shows the complete command output.

Example 3-5 Configuring the IBM NAS KDC

```
[saw921-sys6][/]# /usr/krb5/sbin/config.krb5 -S -r ITSO.RAL.IBM.COM -d
itso.ral.ibm.com -a admin/admin -l sys4.itso.ral.ibm.com -u cn=root -p
<dn_passwd>
Initializing configuration...
Creating /etc/krb5/krb5_cfg_type...
Creating /etc/krb5/krb5.conf...
Creating /var/krb5/krb5kdc/kdc.conf...
Creating database files...
Initializing database 'LDAP' for realm 'ITSO.RAL.IBM.COM'
master key name 'K/M@ITSO.RAL.IBM.COM'
Attempting to bind to one or more LDAP servers. This may take a
while...
You are prompted for the database Master Password.
It is important that you DO NOT FORGET this password.
Enter database Master Password: <---- ENTER <master_passwd> here
Re-enter database Master Password to verify: <-- RE-ENTER
Attempting to bind to one or more LDAP servers. This may take a
while...
WARNING: no policy specified for admin/admin@ITSO.RAL.IBM.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Enter password for principal "admin/admin@ITSO.RAL.IBM.COM": <----
ENTER <krb_passwd> here
Re-enter password for principal "admin/admin@ITSO.RAL.IBM.COM": <--
RE-ENTER
Principal "admin/admin@ITSO.RAL.IBM.COM" created.
Creating keytable...
Attempting to bind to one or more LDAP servers. This may take a
while...
Creating /var/krb5/krb5kdc/kadm5.acl...
Starting krb5kdc...
Attempting to bind to one or more LDAP servers. This may take a
while...
krb5kdc was started successfully.
Starting kadmind...
Attempting to bind to one or more LDAP servers. This may take a
while...
kadmind was started successfully.
The command completed successfully.
```

This command creates all the required Kerberos configuration files automatically. The most important Kerberos configuration files are as follows:

- `/etc/krb5/krb5.conf`

  The main Kerberos configuration file that is used by both the client and server.

- `/var/krb5/krb5kdc/kdc.conf`

  The KDC configuration file that is used only by IBM NAS daemons.

- `/var/krb5/krb5kdc/.kdc_ldap_data`

  The LDAP configuration file for IBM NAS that stores LDAP-related information.

Note: If these files are changed, you will need to restart the IBM NAS daemons. For information about how to start and stop IBM NAS daemons, refer to 3.3.6, "Starting and stopping IBM NAS Kerberos daemons" on page 48.

To confirm that the Kerberos daemons are running, use the **ps** command as shown in Example 3-6. The `krb5kdc` daemon is the IBM NAS KDC. The `kadmind` daemon is the IBM NAS administration server.

Example 3-6 Verifying the Kerberos daemons

```
[saw921-sys6][/]# ps -ef | grep krb
    root 110714      1   0 01:18:39      -  0:00 /usr/krb5/sbin/krb5kdc
    root 270496      1   0 01:18:39      -  0:00 /usr/krb5/sbin/kadmind
```

To test the new KDC configuration, get an initial Kerberos ticket-granting ticket (TGT) using the following IBM NAS command.

```
/usr/krb5/bin/kinit <principal_name>
```

This command contacts the KDC server and gets a TGT after verifying the Kerberos principal's password (as shown in Example 3-7). It stores the TGT in a credential cache in the local file system.

Example 3-7 Getting a Kerberos TGT using the IBM NAS kinit command

```
[saw921-sys6][/]# /usr/krb5/bin/kinit admin/admin
Password for admin/admin@ITSO.RAL.IBM.COM: <-- ENTER <krb_passwd>
```

To view the TGT that was acquired, use the following IBM NAS command:

```
/usr/krb5/bin/klist
```

This command, shown in Example 3-8, reads the credential cache and displays the TGT information.

Example 3-8 Examining the Kerberos TGT

```
[saw921-sys6][/]# /usr/krb5/bin/klist
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  admin/admin@ITSO.RAL.IBM.COM

Valid starting     Expires            Service principal
07/06/09 05:17:52  07/07/09 05:17:51
krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
```

This quick test illustrates that the Kerberos KDC on AIX using IBM NAS is running properly.

3.3 Additional Kerberos tasks

This section covers common tasks to set up any Kerberos application (for example WebSphere Application Server) with the Kerberos KDC, including the following tasks:

- Using the kadmin interface
- Creating a Kerberos principall
- Generating a keytab
- Setting the ticket lifetime
- Changing encryption types
- Starting and stopping IBM NAS Kerberos daemons
- Diagnosing Kerberos related issues

3.3.1 Using the kadmin interface

You can use the **kadmin** command-line interface to create, modify, view, or delete Kerberos principals and policies. You can also manage keytabs and principal passwords from this interface.

The **kadmin** interface communicates securely with the IBM NAS Administration Server (**kadmind**) using Kerberos authentication and can be run from any client system. To invoke the **kadmin** interface, run the following command:

```
/usr/krb5/sbin/kadmin -p admin/admin
```

This command prompts you for the password of the Kerberos principal `admin/admin`. Enter the password to get the command-line interface, as shown in Example 3-9.

Example 3-9 Invoking the kadmin interface

```
[saw921-sys6][/]# /usr/krb5/sbin/kadmin -p admin/admin
Authenticating as principal admin/admin with password.
Password for admin/admin@ITSO.RAL.IBM.COM: <--- ENTER <krb_passwd> here
kadmin:
```

The interface is now ready for other commands. Type a question mark (`?`) for list of available commands. Exit the **kadmin** interface by entering `q`.

3.3.2 Creating a Kerberos principal

This section describes how to create a Kerberos user principal and a Kerberos service principal name.

Creating a Kerberos user principal

The Kerberos user principal is a Kerberos entity that is created for each user in a Kerberos realm. The format of the principal is as follows:

```
<user_name>/<instance>@<REALM>
```

In this command:

- The *<user_name>* is the actual name of the user.
- The *<instance>* qualifies the user (for example, `admin`). The *<instance>* can also be `NULL`.
- The *<REALM>* is the name of the Kerberos realm (in upper case letter) to which the user belongs.

To create a user principal, use the kadmin **`addprinc`** command with the following options:

```
addprinc -pw <password> <user_principal_name>
```

This command scrutinizes the password that is given as per the Kerberos password policy (if present) and creates the principal in the Kerberos database.

Example 3-10 shows the output of this command.

Example 3-10 Creating Kerberos user principal

```
kadmin:  addprinc -pw <password> john
WARNING: no policy specified for john@ITSO.RAL.IBM.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "john@ITSO.RAL.IBM.COM" created.
kadmin:
```

Creating a Kerberos service principal name

The Kerberos service principal name (SPN) is a Kerberos principal that is generally used by the Kerberized services. The format of the principal is as follows:

```
<service>/<fully qualified hostname>
```

In our testing environment, the default service name for WebSphere Application Server is *WAS*, and the default service name for SPNEGO is *HTTP*. For example, if WebSphere is running on a system (`some.host.com`), then the Kerberos SPN is `WAS/some.host.com`.

To create an SPN, use the same **`addprinc`** command with a different option:

```
addprinc -randkey <service_principal_name>
```

This command creates an SPN in the Kerberos database and randomizes its password so that nobody has knowledge of the password. Example 3-11 shows how to use this command to create a Kerberos SPN for WebSphere Application Server.

Note: Always make sure that the host name portion for any Kerberos service principal name is a fully qualified domain name (FQDN), for example `some.host.com`.

Example 3-11 Creating the Kerberos SPN

```
kadmin:  addprinc -randkey WAS/some.host.com@ITSO.RAL.IBM.COM
WARNING: no policy specified for WAS/some.host.com@ITSO.RAL.IBM.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "WAS/some.host.com@ITSO.RAL.IBM.COM" created.
kadmin:
```

To see all the available options for the **`addprinc`** command, enter the **`addprinc`** command with no options. The complete syntax displays as shown in Example 3-12.

Example 3-12 The addprinc command syntax

```
kadmin:  addprinc
usage: add_principal [options] principal
        options are:
                [-expire expdate] [-pwexpire pwexpdate] [-maxlife
maxtixlife]
               [-kvno kvno] [-policy policy] [-clearpolicy] [-randkey]
[-pw password]
                [-maxrenewlife maxrenewlife]
                [-e keysaltlist]
                [{+|-}attribute]
        attributes are:         allow_postdated allow_forwardable
allow_tgs_req allow_renewable
```

```
                allow_proxiable allow_dup_skey allow_tix
requires_preauth
                requires_hwauth needchange allow_svr
password_changing_service
                support_desmd5 delegate_ok
```

3.3.3 Generating a keytab

A Kerberos *keytab* is a file that stores the keys for a particular Kerberos service principal name. This keytab file is generally used by the Kerberos application to read the keys of any Kerberos service.

Consider the keytab file as a table that contains keys of various principals. One keytab file can hold the keys for many principals. To generate a keytab file, first you need to create the Kerberos SPN for which a keytab entry is generated (as discussed in "Creating a Kerberos service principal name" on page 42).

After you create the principal, generate a keytab entry using the **ktadd** command of the **kadmin** interface:

```
ktadd [-k[eytab] keytab] <principal_name>
```

This command adds the keytab entries in the `/etc/krb5/krb5.keytab` file by default. To override this default, specify a fully-qualified file name with the **-k** option in the command. By default, this command adds all the available keys of the specified principal to the keytab file. Before adding the keys to the keytab, it randomizes the keys.

Example 3-13 shows the command to add the Kerberos SPN for WebSphere Application Server in a keytab file called `/tmp/was.keytab`.

Example 3-13 Generating the keytab file

```
kadmin:  ktadd -k /tmp/was.keytab -e arcfour-hmac:normal
WAS/some.host.com@ITSO.RAL.IBM.COM
Entry for principal WAS/some.host.com@ITSO.RAL.IBM.COM with kvno 3,
encryption type ArcFour with HMAC/md5 added to keytab
WRFILE:/tmp/was.keytab.
kadmin:
```

To see the new keytab entries, exit the **kadmin** interface and use the **/usr/krb5/bin/klist** command. As shown in Example 3-14, this command shows the key version number (KVNO), the timestamp when the entry was made, the SPN, the encryption type of the key, and the key.

Example 3-14 Viewing the keytab file

```
[saw921-sys6][/]# /usr/krb5/bin/klist -Ketk /tmp/was.keytab
Keytab name:  FILE:/tmp/was.keytab
KVNO Timestamp         Principal
---- ----------------- --------------------------------------------------------
   3 07/08/09 04:19:47 WAS/some.host.com@ITSO.RAL.IBM.COM (ArcFour with HMAC/md5)
(0xcc73b08c1235a564b1dbe9954ed6f873)
[saw921-sys6][/]#
```

To remove stale entries from a keytab file, use the **ktremove** command of the **kadmin** interface.

3.3.4 Setting the ticket lifetime

Every Kerberos ticket has an elapsed time, referred to as a *lifetime*, after which the ticket expires and becomes unusable. The lifetime is based upon application-specific requirements. A Kerberos ticket can have two types of lifetimes:

- A ticket expiration lifetime, which is the normal time after which the current instance of a ticket is expired.
- A renewable lifetime, which is the maximum time limit to renew a ticket. With a renewable lifetime, a user needs to renew the ticket before the first expiration if the ticket is needed longer. When the renewable lifetime expires, the ticket is expired and is unusable.

By default, IBM NAS sets the ticket expiration lifetime to 1 day and the renewable lifetime to 7 days for all the principals.

The ticket expiration lifetime of any Kerberos ticket is calculated as the minimum of the following values:

- The lifetime specified by the **kinit** command (using the **-l** flag) *or* the `ticket_lifetime` relation in the configuration file (`/etc/krb5/krb5.conf` file). [client-side control]
- The `max_lifetime` value specified in the `/var/krb5/krb5kdc/kdc.conf` file. [KDC-side control]

- The Maximum ticket life value of the user principal. [per-principal control]
- The Maximum ticket life value of the service principal (`krbtgt/REALM@REALM` in the case of an initial TGT and the actual service principal in the case of a service ticket). [per-principal control]

For the ticket renewable lifetime, the formula is similar to the ticket expiration lifetime with the following changes:

- Renew lifetime as specified by **`kinit`** command (by using **`-r`** flag) *or* the `renew_lifetime` relation in the `/etc/krb5/krb5.conf` file. [client-side control]
- The `max_renewable_lifetime` value specified in the `/var/krb5/krb5kdc/kdc.conf` file. [KDC-side control]
- The Maximum renewable life value of the user principal. [per-principal control]
- The Maximum renewable life value of the service principal (`krbtgt/REALM@REALM` in the case of an initial TGT and the actual service principal in the case of a service ticket). [per-principal control]

To set the ticket lifetime at the client-side, you can change the `/etc/krb5/krb5.conf` file for the entire client system. Edit the configuration file and add the `ticket_lifetime` relation (or the `renew_lifetime` relation for a renewable lifetime) in the `[libdefaults]` section of configuration file.

Note that the time units are specified as *d* for day, *h* for hour, *m* for minute, and *s* for seconds. For example. if you want to set a ticket expiration time of 2 hours and 10 minutes, define it as shown in Example 3-15.

Example 3-15 Sample krb5.conf relation for setting ticket expiration time

```
[libdefaults]
.....
.....
ticket_lifetime = 2h 10m
```

To modify the lifetime values of the principal, use the **`modprinc`** command in the **`kadmin`** interface. For example, to modify ticket expiration time to 5 hours for principal `WAS/some.host.com`, use the command as shown in Example 3-16.

Example 3-16 Changing the ticket expiration time of a service principal

```
kadmin:  modprinc -maxlife "5 hours" WAS/some.host.com
Principal "WAS/some.host.com@ITSO.RAL.IBM.COM" modified.
```

3.3.5 Changing encryption types

The Kerberos encryption types specifies the type of encryption algorithm that the client and server use to communicate to other parties. For Kerberos clients, you can change the encryption type by editing the `/etc/krb5/krb5.conf` configuration file.

The following relations govern the encryption type preference for Kerberos clients:

- `default_tkt_enctypes`

  Identifies the list of encryption types used while making the initial ticket request to the KDC.
- `default_tgs_enctypes`

  Identifies the list of encryption types used while requesting a service ticket.

The IBM NAS configuration script creates the encryption types in then `/etc/krb5/krb5.conf` file as shown in Example 3-17.

Example 3-17 Default encryption type configuration in krb5.conf

```
[libdefaults]
...
...
        default_tkt_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts
des-cbc-md5 des-cbc-crc
        default_tgs_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts
des-cbc-md5 des-cbc-crc
...
...
```

IBM NAS supports the following encryption types:

- `aes128–cts`
- `aes256–cts`
- `des–cbc–crc`
- `des–cbc–md4`
- `des–cbc–md5`
- `des3–cbc–sha1`
- `arcfour–hmac`
- `arcfour–hmac–exp`

For example, if you do not need to change the encryption type to ArcFour (RC4), which is generally required while contacting with Microsoft KDC, then on the AIX KDC side, edit the `/etc/krb5/krb5.conf` file. Add `arcfour-hmac` at the beginning of the `default_tkt_enctypes` and `default_tgs_enctypes` relations, as shown in Example 3-18. For compatibility reasons in our testing, we added other encryption types after it.

Example 3-18 Changing the encryption type configuration in krb5.conf

```
default_tkt_enctypes = arcfour-hmac des3-cbc-sha1 aes256-cts
des-cbc-md5 des-cbc-crc
default_tgs_enctypes = arcfour-hmac des3-cbc-sha1 aes256-cts
des-cbc-md5 des-cbc-crc
```

3.3.6 Starting and stopping IBM NAS Kerberos daemons

If you change the KDC configuration file, you need to restart the IBM NAS Kerberos daemons for the changes to take effect using the following command:

```
/usr/krb5/sbin/start.krb5 [krb5kdc | kadmind]
```

If you omit the parameters, **`start.krb5`** starts both the daemons, as shown in Example 3-19.

Example 3-19 Starting the IBM NAS daemons

```
[saw921-sys6][/]# /usr/krb5/sbin/start.krb5
Starting krb5kdc...
Attempting to bind to one or more LDAP servers. This may take a
while...
krb5kdc was started successfully.
Starting kadmind...
Attempting to bind to one or more LDAP servers. This may take a
while...
kadmind was started successfully.
The command completed successfully.
[saw921-sys6][/]#
```

Similarly, to stop IBM NAS daemons, use the following command:

```
/usr/krb5/sbin/stop.krb5 [krb5kdc | kadmind]
```

When executed with no parameters, this command stops both the daemons, as shown in Example 3-20.

Example 3-20 Stopping the IBM NAS daemons

```
[saw921-sys6][/]# /usr/krb5/sbin/stop.krb5
Stopping /usr/krb5/sbin/krb5kdc...
/usr/krb5/sbin/krb5kdc was stopped successfully.
Stopping /usr/krb5/sbin/kadmind...
/usr/krb5/sbin/kadmind was stopped successfully.
The command completed successfully.
[saw921-sys6][/]#
```

3.3.7 Diagnosing Kerberos related issues

If a Kerberos application is not working properly or if clients cannot get the tickets, you first need to check the IBM NAS daemon log files. If you cannot determine the cause of the problem from the log files, you can turn on serviceability to obtain additional debugging information.

IBM NAS daemon log files

The IBM NAS daemons record all the activities and error messages to the log files. All requests (both success and failure) made to any of the daemons are logged. These log files are the primary source of debugging information. You can resolve many configuration-related issues by observing these files.

By default, the IBM NAS KDC daemon (`krb5kdc`) writes to the `/var/krb5/log/krb5kdc.log` file. The administration daemon (`kadmind`) writes to the `/var/krb5/log/kadmind.log` file. You can change the log file settings in the `/etc/krb5/krb5.conf` file on the server system.

Serviceability in IBM NAS

If the log files are not helpful and you need more debugging information, then IBM NAS provides a facility to turn on serviceability. With serviceability turned on, the IBM NAS commands and libraries output extra internal debugging information that can be helpful in diagnosing the actual error.

Note: The internal debugging information is mainly for the use of the IBM NAS support team. If you are experiencing errors, collect the logs, and contact the IBM NAS support team.

To turn on serviceability, you need to set the following environment variables:

- `_KRB5_SVC_DBG`

  Describes the subcomponents and the levels of debugging messages. There are various subcomponents. For each subcomponent, you can set the level of description that you need. Most of the time, this variable is set to output all messages for all the sub-components as follows:

  ```
  export _KRB5_SVC_DBG=*.9
  ```

- `_KRB5_SVC_DBG_MSG_LOGGING`

  Identifies whether to suppress debugging messages. The possible values are `0` (show nothing) and `1` (display the messages).

  ```
  export _KRB5_SVC_DBG_MSG_LOGGING=1
  ```

- `_KRB5_SVC_DBG_FILENAME`

  Specifies which file to use to store the messages. If it is not specified, then `stdout` is used.

  ```
  export _KRB5_SVC_DBG_FILENAME=/tmp/krb5_kinit.log
  ```

For a complete list of environment variables, refer the *IBM NAS 1.4 Administrator's and User's Guide for AIX, Linux and Solaris*, which is available only with the product. It is not available online.

4

Setting up Microsoft Active Directory and Kerberos KDC

This chapter provides information about how to configure the Microsoft Active Directory environment that is used in some scenarios for this book. It explains how to create the Microsoft Active Directory domains and how to establish the trust relationship between these Kerberos realms.

This chapter includes the following topics:

- Introduction to Microsoft Kerberos KDC
- Configuring a domain name system server
- Creating the Microsoft Kerberos KDC
- Establishing a trust relationship between Microsoft Active Directory realms
- Validating the trust relationship
- Adding a Kerberos service principal name

4.1 Introduction to Microsoft Kerberos KDC

Kerberos is implemented in Microsoft Windows Server 2000 and later. The implementation of Kerberos on a Windows server is composed of the Key Distribution Center (KDC) as a domain service. The KDC uses the domain's Active Directory as its user registry. The KDC provides two services:

- Authentication service
- Ticket-granting service (TGS)

These services are started automatically and run in the domain controller for a Microsoft Active Directory architecture.

When a user logs on to the Windows domain, the information that the user enters is captured by the logon program and is transmitted to the computer's local security authority (LSA). The LSA is a Windows component that authenticates users to the local system. This LSA then communicates with the network's KDC in order to receive ticket-granting tickets and service tickets so that the user can access Kerberized services on the Windows domain.

Kerberos on Windows server platforms uses Active Directory for all information about Kerberos principals on the Kerberos network. The encryption key that is used to communicate with Kerberos principals is stored in the Active Directory database in the user's profile.

Active Directory plays the role of an LDAP server on a Windows server and is also used as the KDC database. This dual role can be a great benefit to an organization in terms of maintaining one central store but can be an issue in the event of system failure.

4.2 Configuring a domain name system server

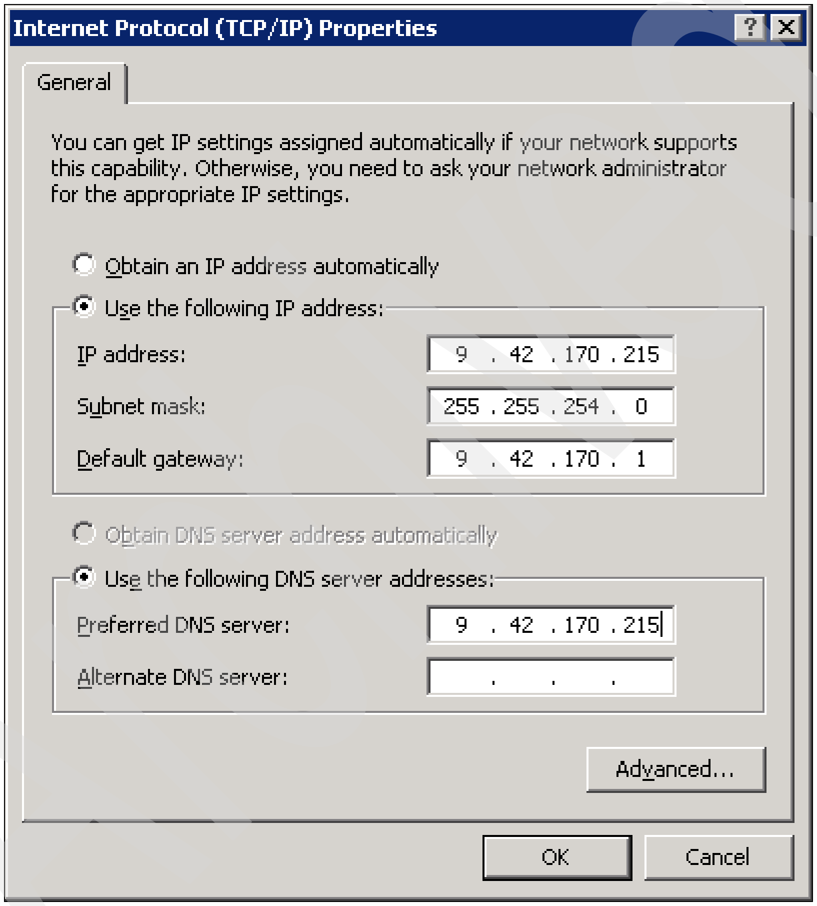
A domain name system (DNS) service is required for Active Directory to work properly. In this section, we explain how to create the DNS configuration that the Active Directory domain uses. To complete the steps in this section, we assume that you have a basic understanding of DNS and how it works in your environment. For further information about DNS refer to RFC1035, which is available at:

`http://www.ietf.org/rfc/rfc1035.txt`

When creating a realm (domain), the wizard verifies that the DNS is running. If the DNS is not running, the wizard provides the option to create the DNS service in the same machine where you create the realm.

Note: The DNS service does not need to run in the same machine where you create the KDC realm. You can execute these services in different machines. However, for the scenarios in this book, the DNS server runs in the same machine where we created the realm.

To configure the DNS, you first need to update the IP address for the “Preferred DNS server” field to be the IP address of the DNS server, as shown in Figure 4-1.



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

( ) Obtain an IP address automatically
[✓ Use the following IP address:]

IP address:	9 . 42 . 170 . 215
Subnet mask:	255 . 255 . 254 . 0
Default gateway:	9 . 42 . 170 . 1

( ) Obtain DNS server address automatically
[✓ Use the following DNS server addresses:]

Preferred DNS server:	9 . 42 . 170 . 215
Alternate DNS server:	. . .

Figure 4-1 TCP/IP configuration for DNS

If a DNS server is not installed in your system, use the following information to install the service:

- For Windows 2003 Server, use the instructions at:

  `http://technet.microsoft.com/en-us/library/cc779205(WS.10).aspx`

- For Windows Server 2008, select **Start** → **Server Manager**. In the Server Manager console, select **Roles** in the left Menu to display the current roles that the server is playing. Click **Add Roles**, and add the DNS server as a service that is provided by your system.

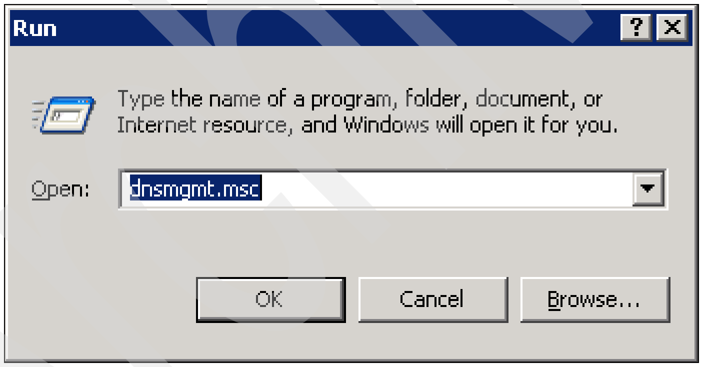
After you install the DNS server, you need to configure it. The following steps describe how to configure the DNS service for a domain with the following suffix:

```
kdc.itso.com
```

Note: Windows 2000, 2003, and 2008 servers have similar procedures for DNS configuration, although you will find some variations in the screens.

To configure the DNS server, follow these steps:

1. Start the DNS console by clicking **Start** → **Run**. Then, enter `dnsmgmt.msc` and click **OK**, as shown in Figure 4-2.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

dnsmgmt.msc

Figure 4-2 Running the dnsmgmt.msc command

2. The DNS window opens. Right-click **Forward Lookup Zones** and select **New Zone**, as shown in Figure 4-3.

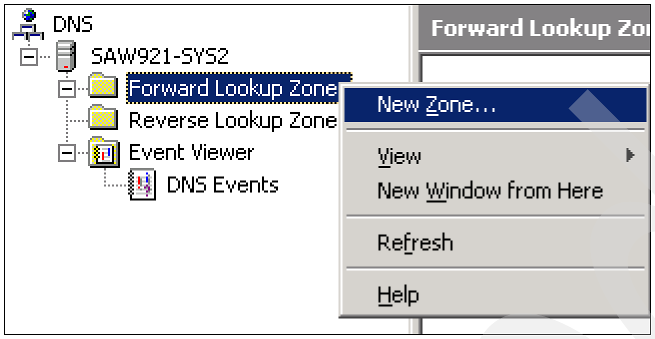
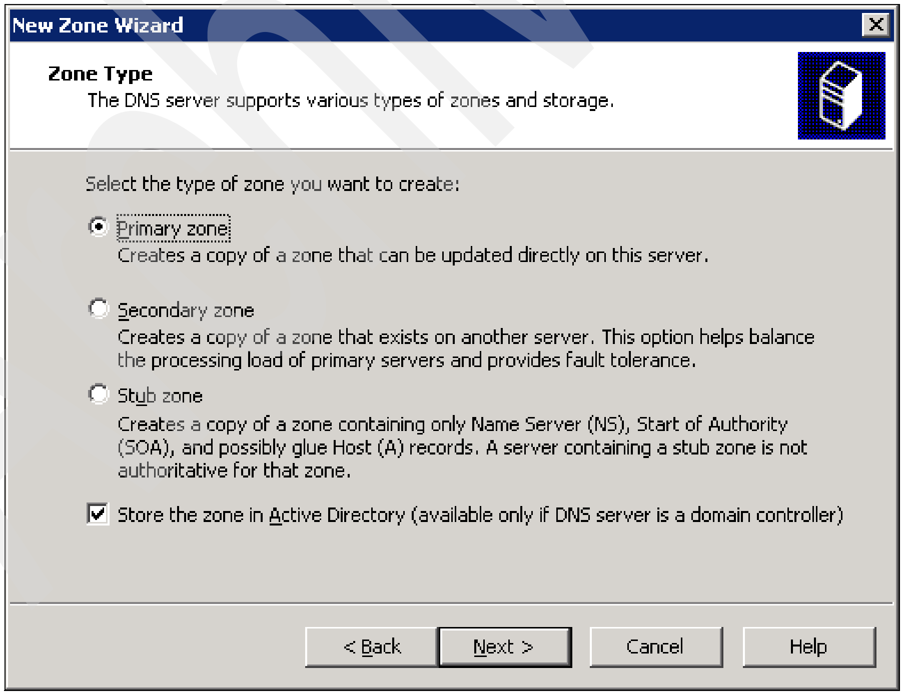


Figure 4-3 Creating a new zone

3. When the Wizard opens, click **Next**.
4. In the Zone Type window, select **Primary Zone**, and enable the “Store the zone in Active Directory” option, as shown in Figure 4-4, because the DNS server and the KDC will run in the same machine. Click **Next**.



New Zone Wizard

Zone Type
The DNS server supports various types of zones and storage.

Select the type of zone you want to create:

[● Primary zone]
Creates a copy of a zone that can be updated directly on this server.

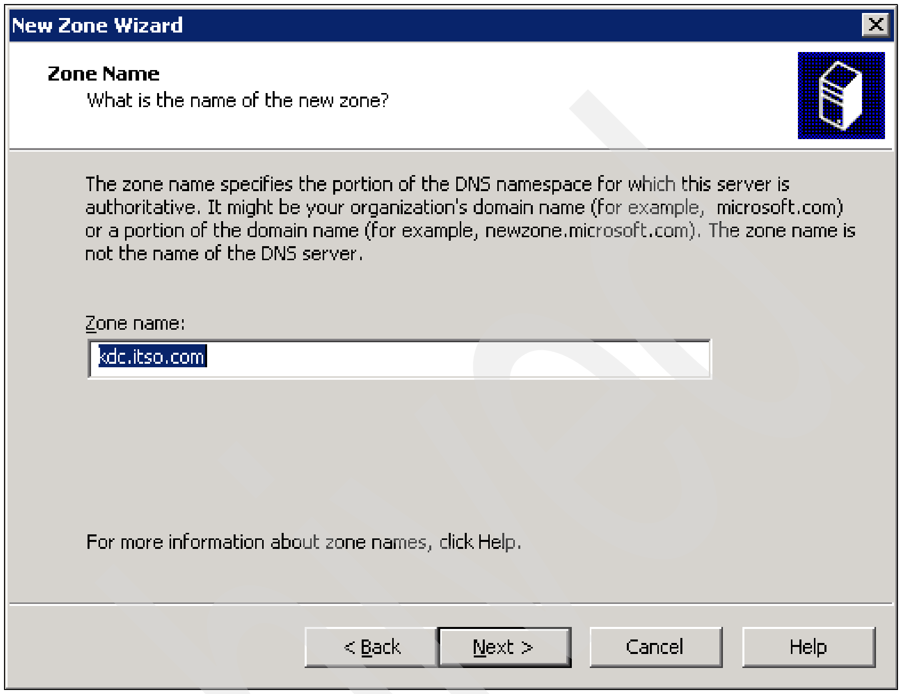
○ Secondary zone
Creates a copy of a zone that exists on another server. This option helps balance the processing load of primary servers and provides fault tolerance.

○ Stub zone
Creates a copy of a zone containing only Name Server (NS), Start of Authority (SOA), and possibly glue Host (A) records. A server containing a stub zone is not authoritative for that zone.

[✓] Store the zone in Active Directory (available only if DNS server is a domain controller)

Figure 4-4 Selecting the type of DNS zone

5. Next, you need to provide the zone name, which in this case is `kdc.itso.com` as shown in Figure 4-5. Click **Next**.


New Zone Wizard

Zone Name
What is the name of the new zone?

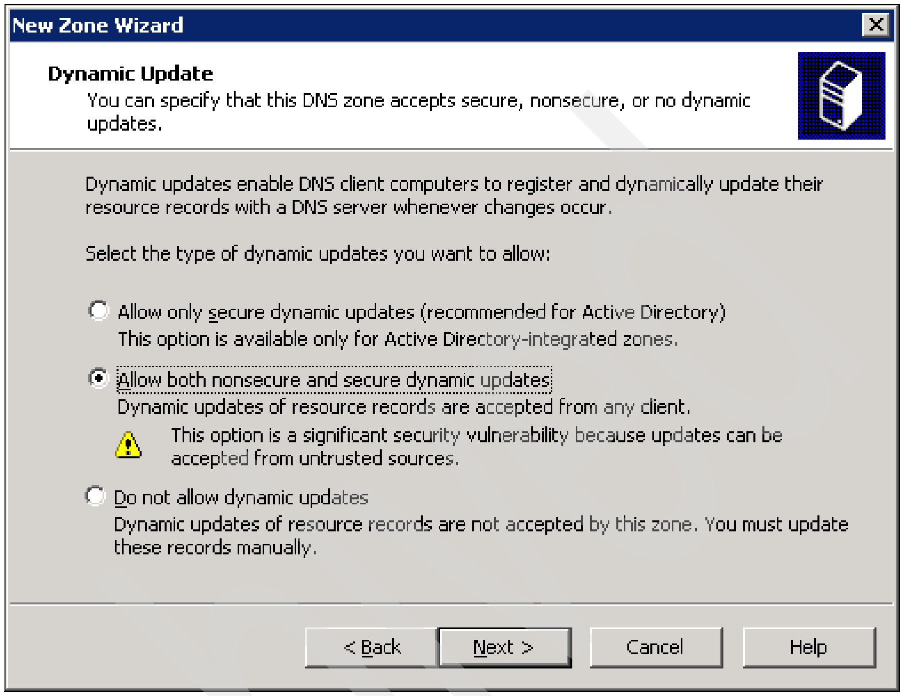
The zone name specifies the portion of the DNS namespace for which this server is authoritative. It might be your organization's domain name (for example, microsoft.com) or a portion of the domain name (for example, newzone.microsoft.com). The zone name is not the name of the DNS server.

Zone name: kdc.itso.com

For more information about zone names, click Help.

Figure 4-5 Providing the zone name

6. In the Dynamic Update window, select the “Allow both nonsecure and secure dynamic updates” option, as shown in Figure 4-6, and click **Next**.



You can specify that this DNS zone accepts secure, nonsecure, or no dynamic updates.

Dynamic updates enable DNS client computers to register and dynamically update their resource records with a DNS server whenever changes occur.

Select the type of dynamic updates you want to allow:

( ) Allow only secure dynamic updates (recommended for Active Directory)
This option is available only for Active Directory-integrated zones.

(●) Allow both nonsecure and secure dynamic updates
Dynamic updates of resource records are accepted from any client.
This option is a significant security vulnerability because updates can be accepted from untrusted sources.

( ) Do not allow dynamic updates
Dynamic updates of resource records are not accepted by this zone. You must update these records manually.

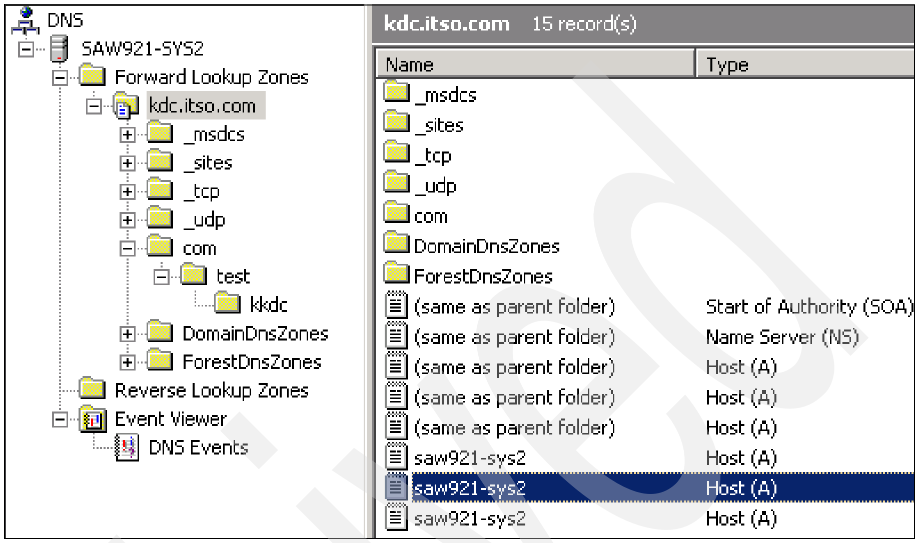
Figure 4-6 Allowing dynamic updates

7. A summary displays. Click **Finish** to create the DNS Forward Zone.
8. To test the new DNS configuration, open a command prompt window and execute the command **`ipconfig /registerdns`**. This command displays a message as shown in Example 4-1.

Example 4-1 Registering on DNS server

```
c:\>ipconfig /registerdns

Windows Ip Configuration

Registration of the DNS resource records for all adapters of this
computer has been initiated. Any errors will be reported in the
Event Viewer in 15 minutes.
```

9. Check the DNS console to make sure that the host was registered. An entry displays with the host name and the IP address of your host, as shown in Figure 4-7.



```
DNS
  SAW921-SYS2
    Forward Lookup Zones
      kdc.itso.com
        _msdcs
        _sites
        _tcp
        _udp
        com
          test
            kkdc
        DomainDnsZones
        ForestDnsZones
    Reverse Lookup Zones
    Event Viewer
      DNS Events
```

kdc.itso.com 15 record(s)

Name	Type
_msdcs	
_sites	
_tcp	
_udp	
com	
DomainDnsZones	
ForestDnsZones	
(same as parent folder)	Start of Authority (SOA)
(same as parent folder)	Name Server (NS)
(same as parent folder)	Host (A)
(same as parent folder)	Host (A)
(same as parent folder)	Host (A)
saw921-sys2	Host (A)
saw921-sys2	Host (A)
saw921-sys2	Host (A)

Figure 4-7 Checking whether the host entry was created on the DNS server

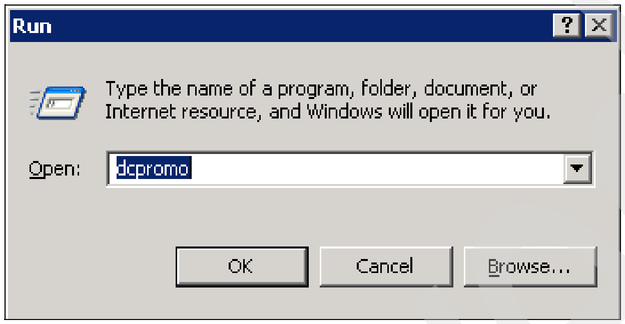
4.3 Creating the Microsoft Kerberos KDC

This section describes how to create and configure the Active Directory domain that we used for the scenarios in this book. To create this realm, the following requirements must be met:

- A workstation running a suitable operating system for Active Directory is available.
- A user ID in the administrative group on the system.
- A DNS server is active and has the DNS zone that is used in the domain. (Alternatively, you can create the DNS zone at the same time as the Microsoft Kerberos KDC.)

The KDC is installed when the domain controller is created. To create the Kerberos KDC, follow these steps:

1. Issue the **`dcpromo`** command on the server that will be acting as the KDC server. Click **Start** → **Run**, enter `dcpromo`, and click **OK**, as shown in Figure 4-8.



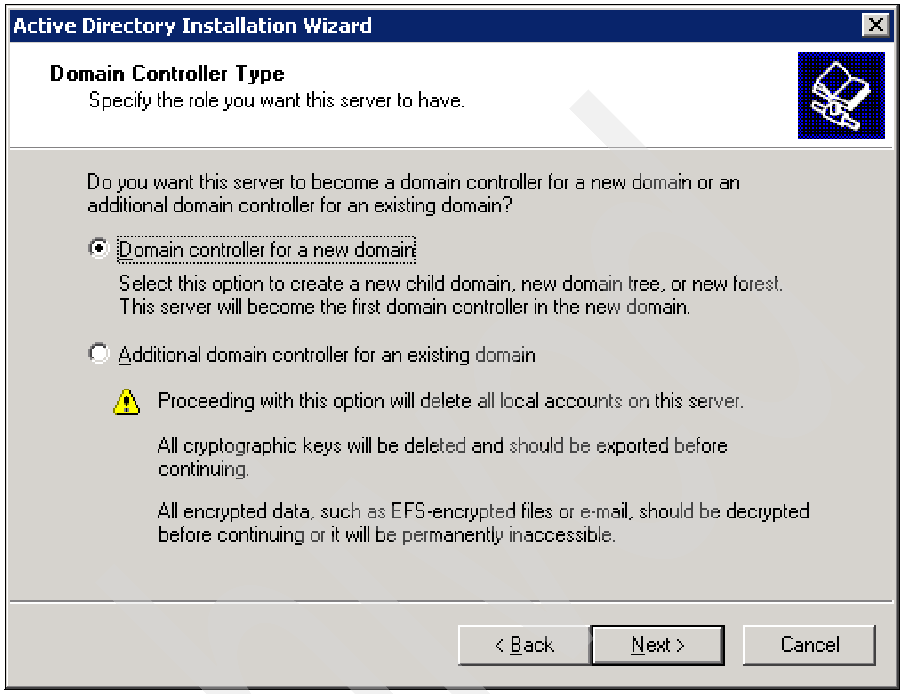
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: dcpromo

Figure 4-8 Running the dcpromo command

2. In the Welcome window, click **Next** to proceed.
3. Then, in the Operating System Compatibility window, click **Next**.

4. In the Domain Controller Type window, select the “Domain Controller for a new domain” as shown in Figure 4-9. Then, click **Next**.



Active Directory Installation Wizard

Domain Controller Type

Specify the role you want this server to have.

Do you want this server to become a domain controller for a new domain or an additional domain controller for an existing domain?

[✓ Domain controller for a new domain]

Select this option to create a new child domain, new domain tree, or new forest. This server will become the first domain controller in the new domain.

Additional domain controller for an existing domain

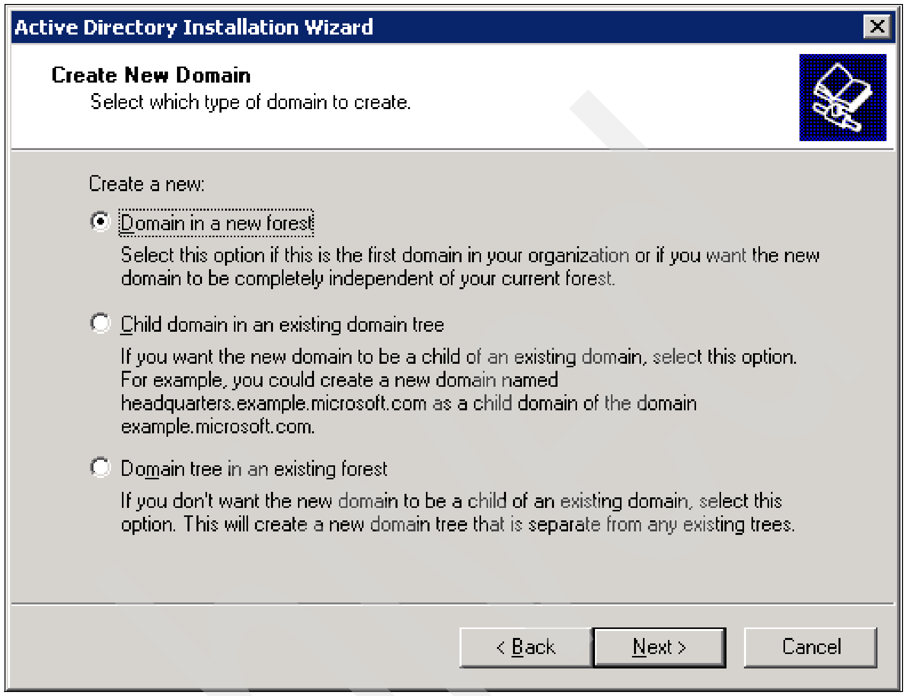
Proceeding with this option will delete all local accounts on this server.

All cryptographic keys will be deleted and should be exported before continuing.

All encrypted data, such as EFS-encrypted files or e-mail, should be decrypted before continuing or it will be permanently inaccessible.

Figure 4-9 Selecting Domain Controller options

5. Because this is a new KDC realm, in the Create New Domain window, select the “Domain in a new forest” option, as shown in Figure 4-10. Click **Next**.



Select which type of domain to create.

Create a new:

[✓ Domain in a new forest]

Select this option if this is the first domain in your organization or if you want the new domain to be completely independent of your current forest.

Child domain in an existing domain tree

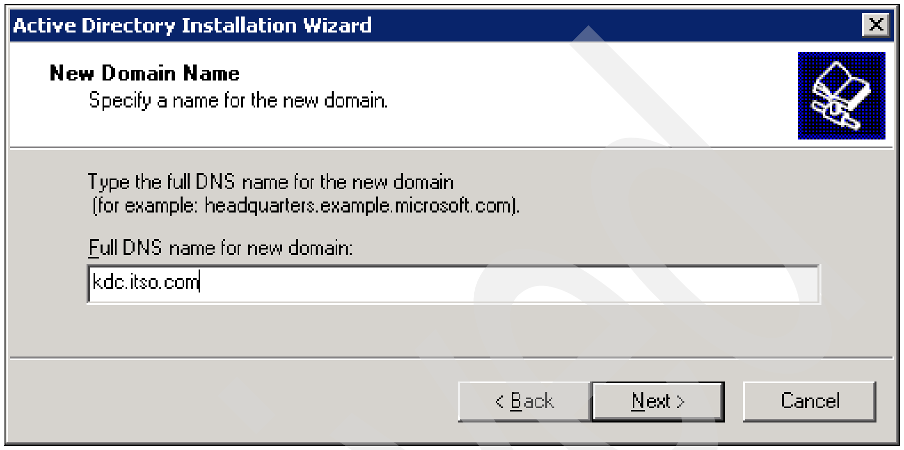
If you want the new domain to be a child of an existing domain, select this option. For example, you could create a new domain named headquarters.example.microsoft.com as a child domain of the domain example.microsoft.com.

Domain tree in an existing forest

If you don't want the new domain to be a child of an existing domain, select this option. This will create a new domain tree that is separate from any existing trees.

Figure 4-10 Creating a new domain in a forest

6. In the New Domain Name window, enter the full DNS name of the domain as shown in Figure 4-11. This name needs to match the name that you set up in the DNS, in this case (`kdc.itso.com`), as shown in Figure 4-5 on page 56. Click **Next**.

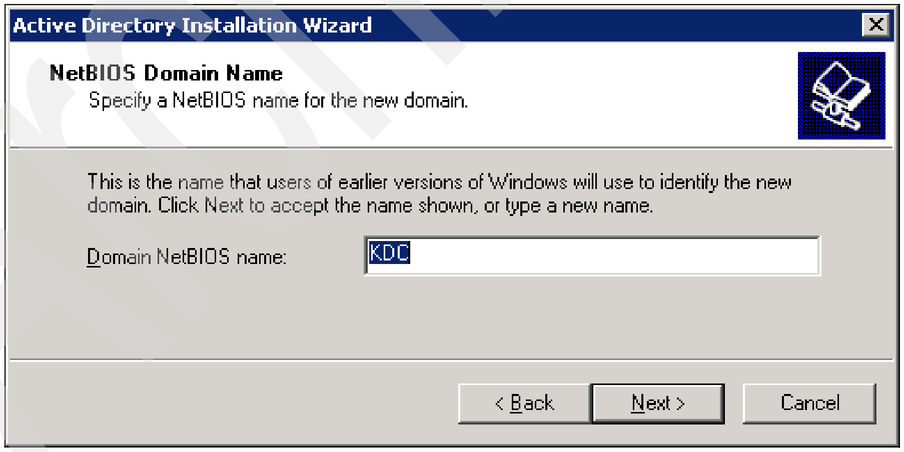

Active Directory Installation Wizard
New Domain Name
Specify a name for the new domain.
< Back
Next >
Cancel

Type the full DNS name for the new domain (for example: headquarters.example.microsoft.com).

Full DNS name for new domain: kdc.itso.com

Figure 4-11 DNS name configuration

7. If the Windows Server is 2003, you need to provide a NetBIOS name for the new domain. In this scenario, the domain is called *KDC*, as shown in Figure 4-12. Click **Next**.


Active Directory Installation Wizard
NetBIOS Domain Name
Specify a NetBIOS name for the new domain.
< Back
Next >
Cancel

This is the name that users of earlier versions of Windows will use to identify the new domain. Click Next to accept the name shown, or type a new name.

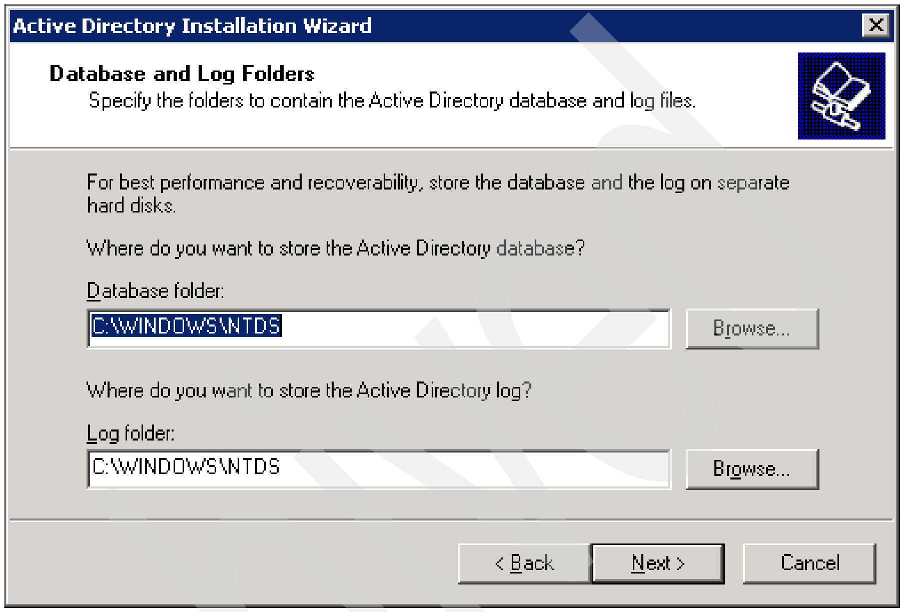
Domain NetBIOS name: KDC

Figure 4-12 Providing the NetBIOS name

8. If your Windows Server is 2008, you need to provide information about the functional level for the forest and the domain. Select the option that is applicable to your environment. For example, if you have Windows 2000

servers in the network, select the Windows 2000 option for both the Forest functional level and Domain functional level settings.

9. If your Windows Server is 2003, you are prompted to provide a location for the database and the log file as shown in Figure 4-13. Click **Next**.



Active Directory Installation Wizard

Database and Log Folders
Specify the folders to contain the Active Directory database and log files.

For best performance and recoverability, store the database and the log on separate hard disks.

Where do you want to store the Active Directory database?

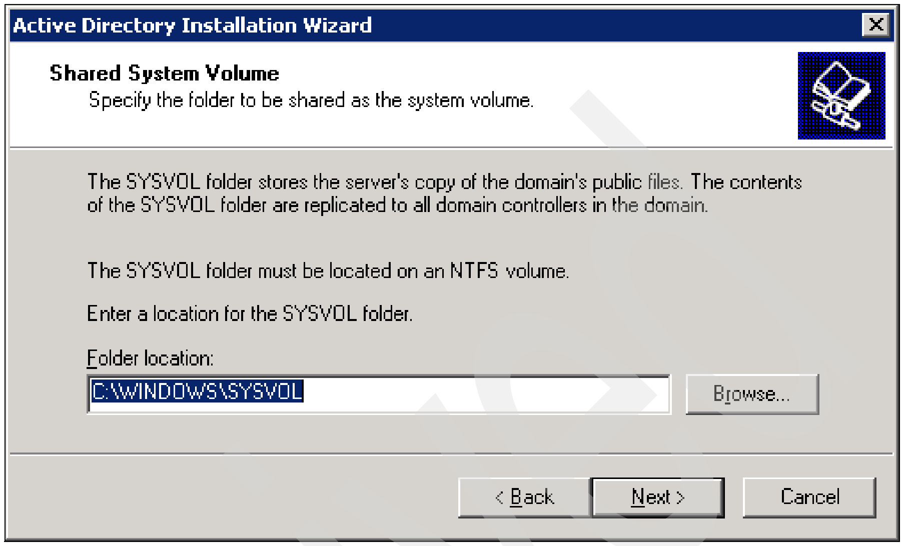
Database folder: C:\WINDOWS\NTDS

Where do you want to store the Active Directory log?

Log folder: C:\WINDOWS\NTDS

Figure 4-13 Selecting files location for database and log files

10. Accept the default location for SYSVOL folder as shown in Figure 4-14. Click **Next**.


Active Directory Installation Wizard

Shared System Volume

Specify the folder to be shared as the system volume.

The SYSVOL folder stores the server's copy of the domain's public files. The contents of the SYSVOL folder are replicated to all domain controllers in the domain.

The SYSVOL folder must be located on an NTFS volume.

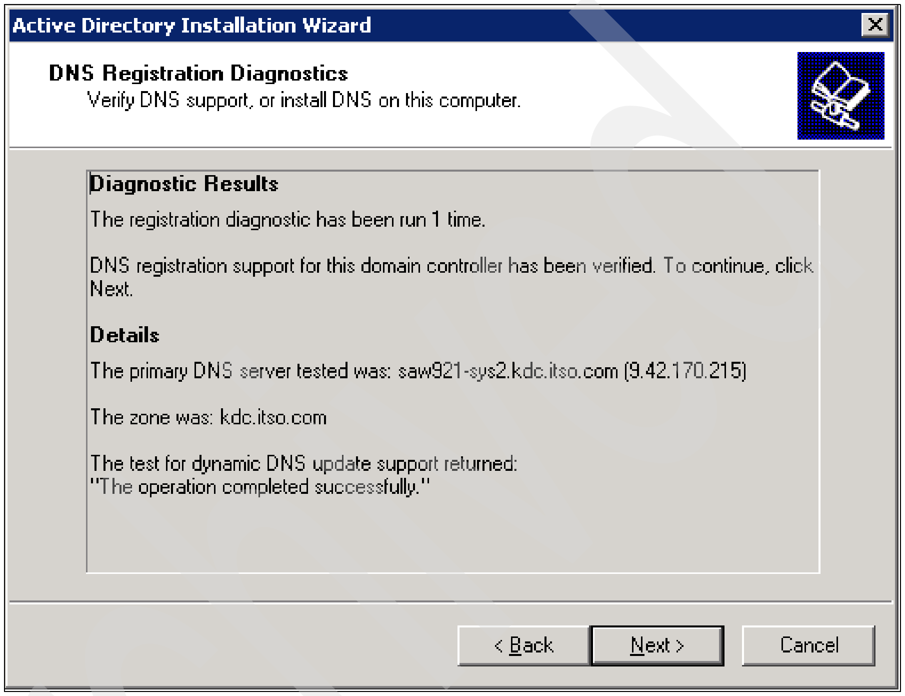
Enter a location for the SYSVOL folder.

Folder location:

C:\WINDOWS\SYSVOL

Figure 4-14 Selecting the location for SYSVOL folder

11. If your Windows Server is 2003 and the proper DNS configuration is in place, a window displays information about the configuration, and you can move forward. Otherwise, you are prompted to re-create the DNS configuration. Check the DNS configuration and machine name if you have problems. Figure 4-15 displays a report with the information about the DNS. Click **Next**.


Active Directory Installation Wizard
DNS Registration Diagnostics
Verify DNS support, or install DNS on this computer.
< Back
Next >
Cancel

Diagnostic Results

The registration diagnostic has been run 1 time.

DNS registration support for this domain controller has been verified. To continue, click Next.

Details

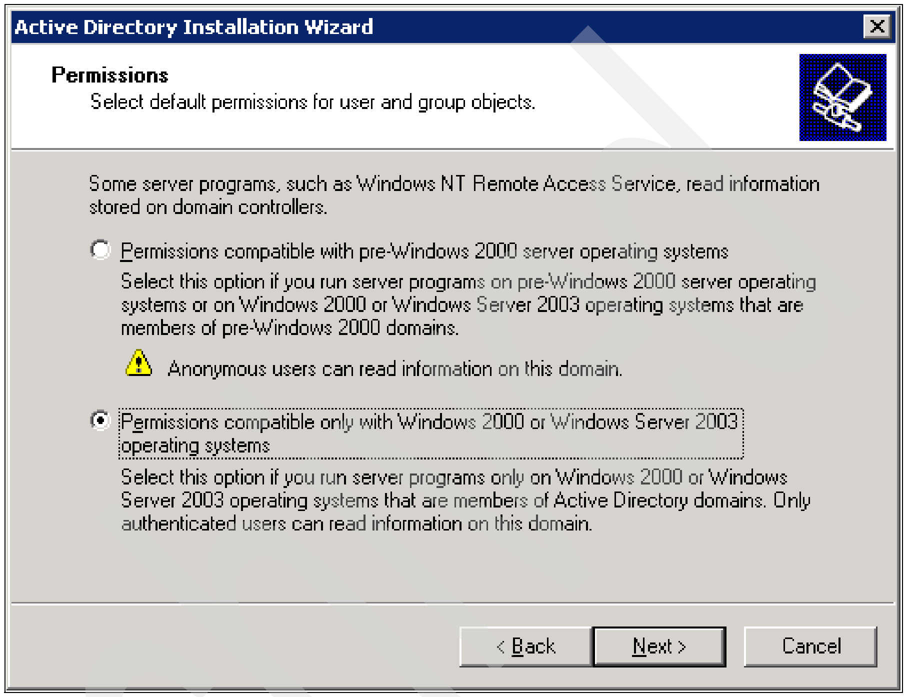
The primary DNS server tested was: saw921-sys2.kdc.itso.com (9.42.170.215)

The zone was: kdc.itso.com

The test for dynamic DNS update support returned:
"The operation completed successfully."

Figure 4-15 DNS diagnostic results

12. This step is applicable for Windows Server 2000 and 2003 only. If there are no pre-Windows 2000 servers, select the “Permissions compatible only with Windows 2000 or Windows Server 2003 operating systems” option as shown in Figure 4-16. Click **Next**.


Active Directory Installation Wizard

Permissions

Select default permissions for user and group objects.

Some server programs, such as Windows NT Remote Access Service, read information stored on domain controllers.

( ) Permissions compatible with pre-Windows 2000 server operating systems

Select this option if you run server programs on pre-Windows 2000 server operating systems or on Windows 2000 or Windows Server 2003 operating systems that are members of pre-Windows 2000 domains.

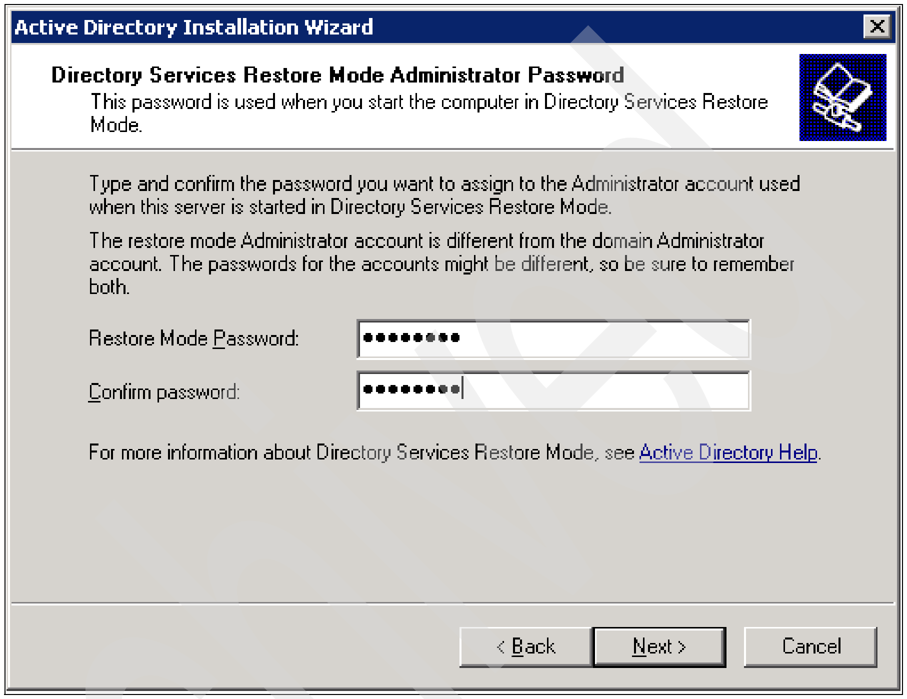
Anonymous users can read information on this domain.

[✓ Permissions compatible only with Windows 2000 or Windows Server 2003 operating systems]

Select this option if you run server programs only on Windows 2000 or Windows Server 2003 operating systems that are members of Active Directory domains. Only authenticated users can read information on this domain.

Figure 4-16 Selecting compatibility with Windows 2000 and 2003 servers only

13. Provide a password for the administrator account to use in the event that this domain controller is started in restore modem as shown in Figure 4-17. This password must be different from the current administrator account. Click **Next**.



Active Directory Installation Wizard

Directory Services Restore Mode Administrator Password

This password is used when you start the computer in Directory Services Restore Mode.

Type and confirm the password you want to assign to the Administrator account used when this server is started in Directory Services Restore Mode.

The restore mode Administrator account is different from the domain Administrator account. The passwords for the accounts might be different, so be sure to remember both.

Restore Mode Password: ●●●●●●●●●

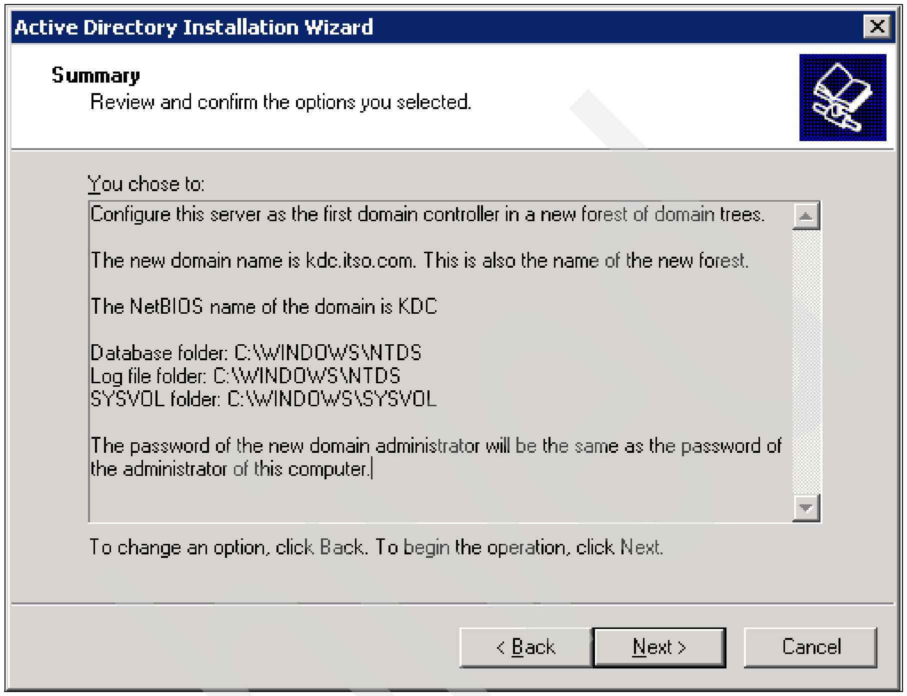
Confirm password: ●●●●●●●●●

For more information about Directory Services Restore Mode, see Active Directory Help.

Figure 4-17 Providing the password for restore mode

Note: If you are using a Windows 2003 or 2008 Server, you can update this password later using the NTDSUTIL command.

14. Review the summary of the installation, as shown in Figure 4-18. Click **Next** to create the KDC.



Active Directory Installation Wizard

Summary
Review and confirm the options you selected.

You chose to:

Configure this server as the first domain controller in a new forest of domain trees.

The new domain name is kdc.itso.com. This is also the name of the new forest.

The NetBIOS name of the domain is KDC

Database folder: C:\WINDOWS\NTDS
Log file folder: C:\WINDOWS\NTDS
SYSVOL folder: C:\WINDOWS\SYSVOL

The password of the new domain administrator will be the same as the password of the administrator of this computer.

To change an option, click Back. To begin the operation, click Next.

Figure 4-18 Summary of the installation

The KDC creation will take a few minutes.

If you determine that you need to restart the wizard with different information, do not cancel the wizard. Wait until the installation has completed, and then remove the realm and reinstall it.

15. If no errors are encountered, the KDC is created successfully. Click **Finish**.

16. When the installation has completed, select **Restart Now** to reboot the system.

Note: This book contains several scenarios that use the Microsoft Active Directory and Kerberos KDC. We created an additional Microsoft Active Directory domain (`kkdc.test.com`) and Kerberos realm (`KKDC.TEST.COM`) has using this same process for use in these scenarios.

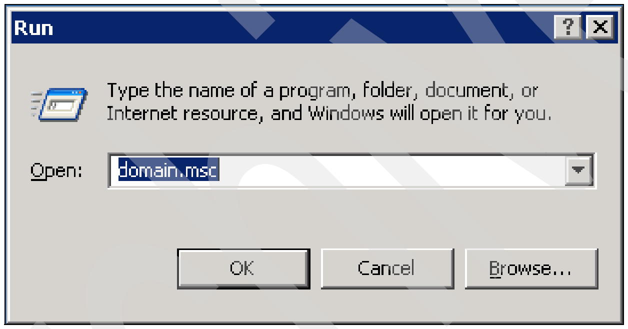
4.4 Establishing a trust relationship between Microsoft Active Directory realms

This section describes how to create the trust relationship between two Active Directory domains. In this scenario, a two-way trust relationship is created between the `kdc.itso.com` and `kkdc.test.com` realms. This trust enables users in the reciprocal realms to access resources in either domain.

The process that we describe in this section assumes that the two distinct Active Directory domains are configured and working properly. Also, to create the trust relationship, you must have administrative credentials for each of realm.

To begin, select one of the KDC servers (`kdc.itso.com` in this case) and perform the following steps:

1. Click **Start** → **Run**. Type `domain.msc`, as shown in Figure 4-19, and click **OK**.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

domain.msc

Figure 4-19 Executing domain.msc command

2. In the Active Directory Domain and Trusts console, right-click the domain, and select **Properties**, as shown in Figure 4-20.

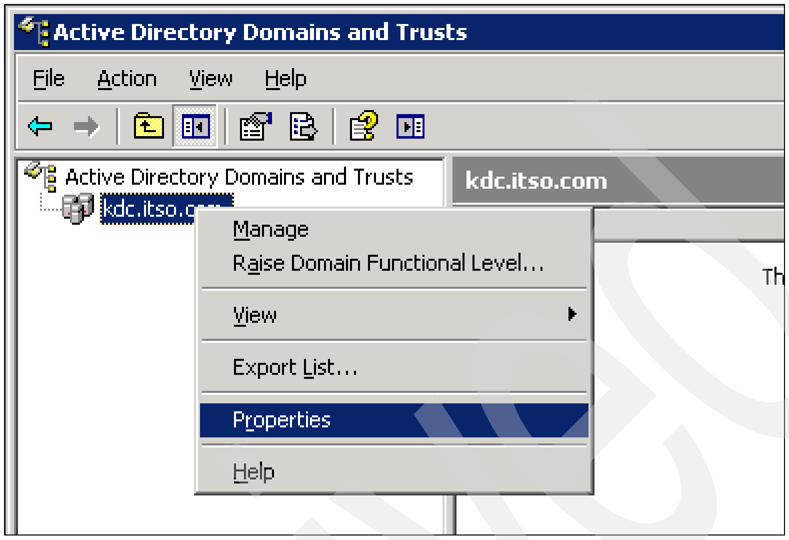

Active Directory Domains and Trusts
File
Action
View
Help
Active Directory Domains and Trusts
kdc.itso.com
kdc.itso.com
Manage
Raise Domain Functional Level...
View
Export List...
Properties
Help

Figure 4-20 Updating the realm configuration

3. Go to the Trusts tab, and click **New Trust**, as shown in Figure 4-21.

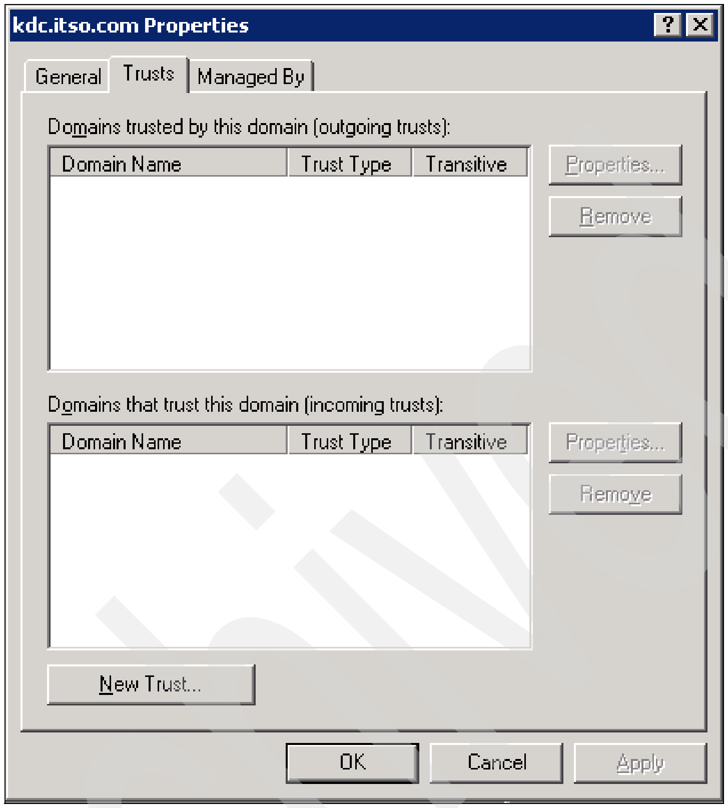
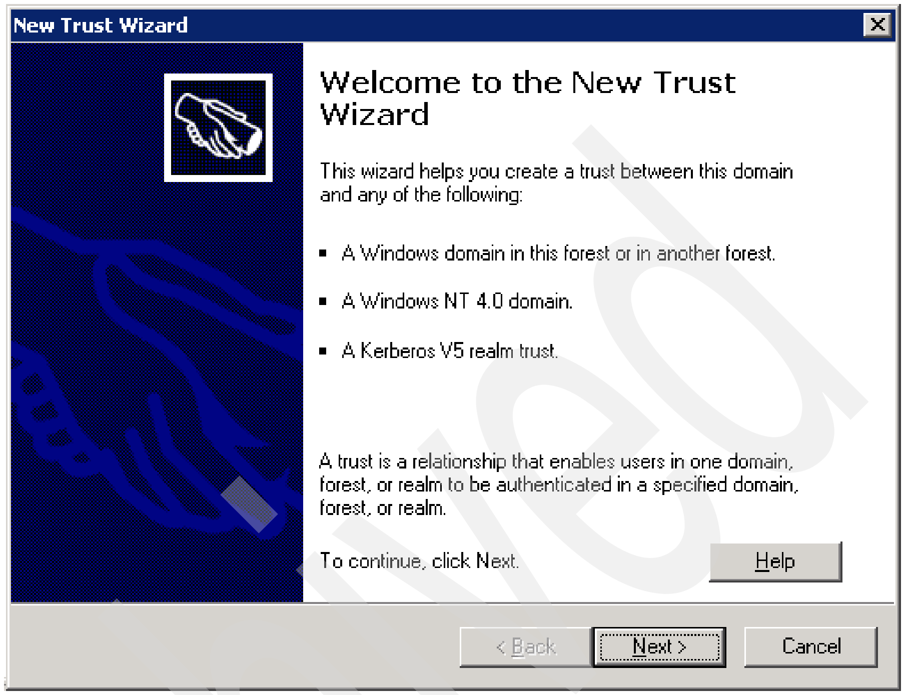


Figure 4-21 Creating a new trust relationship

4. In the Welcome window, shown in Figure 4-22, click **Next** to proceed.



New Trust Wizard

Welcome to the New Trust Wizard

This wizard helps you create a trust between this domain and any of the following:

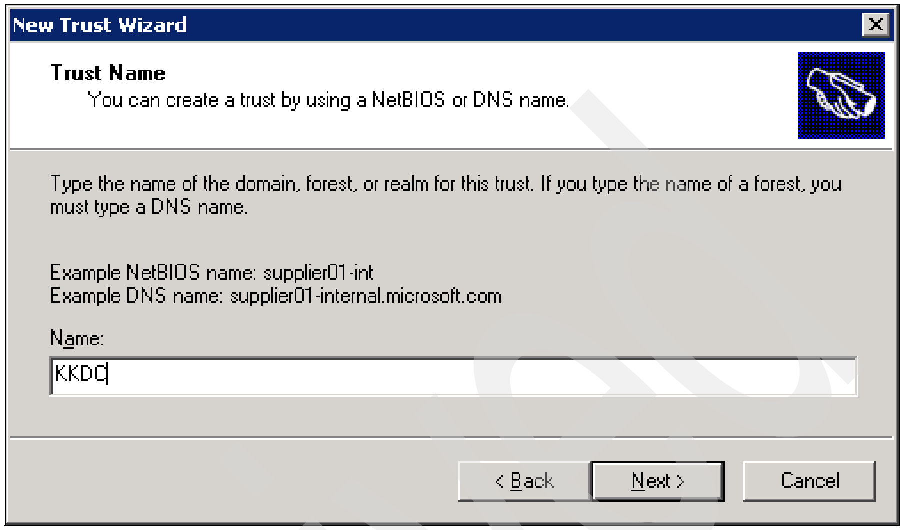
- A Windows domain in this forest or in another forest.
- A Windows NT 4.0 domain.
- A Kerberos V5 realm trust.

A trust is a relationship that enables users in one domain, forest, or realm to be authenticated in a specified domain, forest, or realm.

To continue, click Next.

Figure 4-22 Trust relationship configuration wizard

5. Enter the NetBios name of the domain with which you want to create the trust relationship, in this case `KKDC` as shown in Figure 4-23. Click **Next**.



New Trust Wizard

Trust Name
You can create a trust by using a NetBIOS or DNS name.

Type the name of the domain, forest, or realm for this trust. If you type the name of a forest, you must type a DNS name.

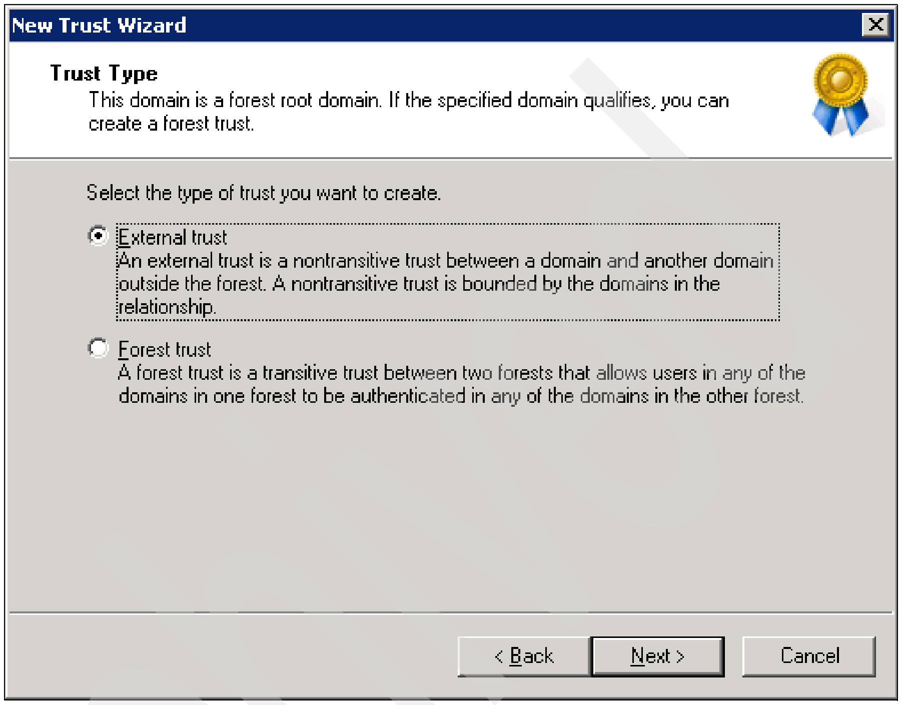
Example NetBIOS name: supplier01-int
Example DNS name: supplier01-internal.microsoft.com

Name: KKDC

Figure 4-23 Providing the name of the trusted realm

Note: If you encounter problems during this step, make sure that the DNS server has the information about the domain with which you are trying to establish the trust relationship. If the DNS server does not have this information, you must create a new host entry in the DNS server with the information about the IP address and the DNS name for the trusted realm.

6. If you are using Windows 2008 Server, you next select the trust type to create. In this scenario, we are creating trust between two forest root domains, so select **External trust** as shown in Figure 4-24. Click **Next**.


New Trust Wizard

Trust Type

This domain is a forest root domain. If the specified domain qualifies, you can create a forest trust.

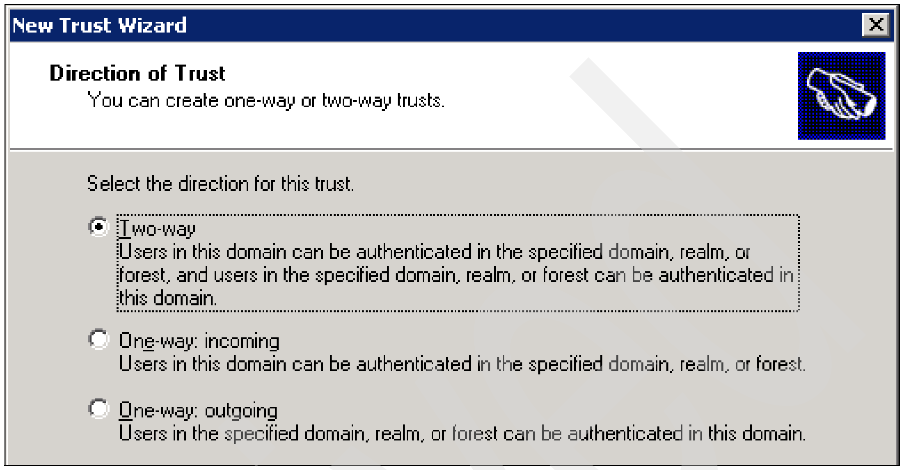
Select the type of trust you want to create.

[✓ External trust]
An external trust is a nontransitive trust between a domain and another domain outside the forest. A nontransitive trust is bounded by the domains in the relationship.

Forest trust
A forest trust is a transitive trust between two forests that allows users in any of the domains in one forest to be authenticated in any of the domains in the other forest.

Figure 4-24 Selecting trust type for Windows 2008 Server

7. Because the trust relationship is being established between the realms, select **Two-way** in the Direction of Trust window, as shown in Figure 4-25. Click **Next**.


New Trust Wizard

Direction of Trust
You can create one-way or two-way trusts.

Select the direction for this trust.

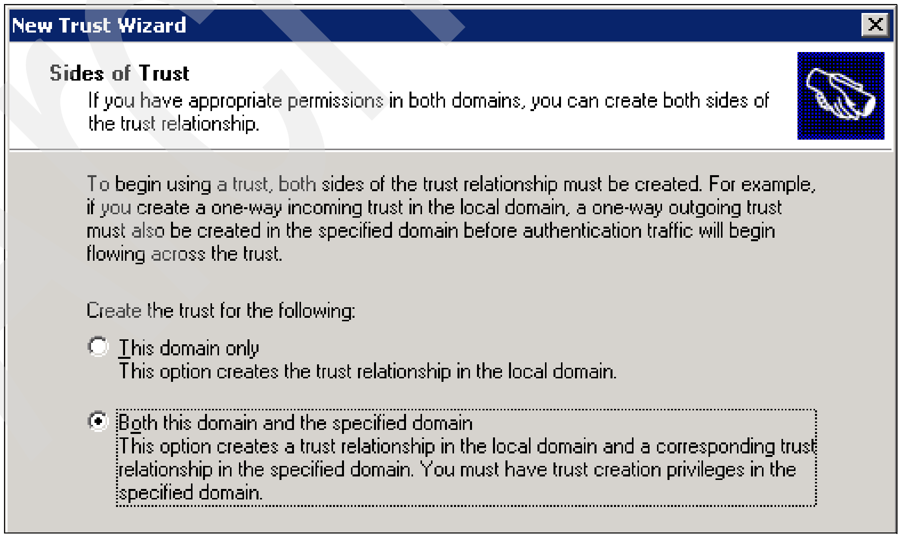
[✓ Two-way]
Users in this domain can be authenticated in the specified domain, realm, or forest, and users in the specified domain, realm, or forest can be authenticated in this domain.

One-way: incoming
Users in this domain can be authenticated in the specified domain, realm, or forest.

One-way: outgoing
Users in the specified domain, realm, or forest can be authenticated in this domain.

Figure 4-25 Selecting a Two-way trust relationship

8. If you have administrative credentials for the second realm, you can choose to have the trust relationship established on both servers by selecting **Both this domain and the specified domain** in the Sides of Trust window, as shown in Figure 4-26. Click **Next**.


New Trust Wizard

Sides of Trust
If you have appropriate permissions in both domains, you can create both sides of the trust relationship.

To begin using a trust, both sides of the trust relationship must be created. For example, if you create a one-way incoming trust in the local domain, a one-way outgoing trust must also be created in the specified domain before authentication traffic will begin flowing across the trust.

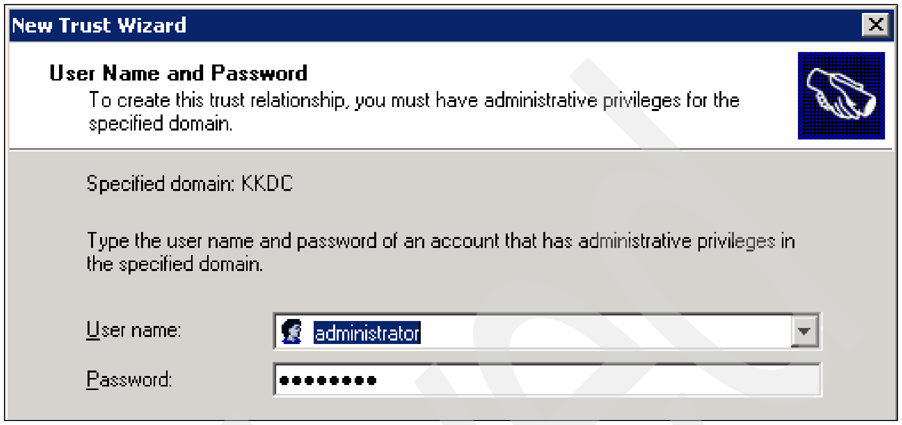
Create the trust for the following:

This domain only
This option creates the trust relationship in the local domain.

[✓ Both this domain and the specified domain]
This option creates a trust relationship in the local domain and a corresponding trust relationship in the specified domain. You must have trust creation privileges in the specified domain.

Figure 4-26 Selecting the servers on which to create the trust

9. Provide the administrative credentials for the domain that you want to establish the trust relationship, in this case `KKDC` as shown in Figure 4-27. Click **Next**.



New Trust Wizard

User Name and Password
To create this trust relationship, you must have administrative privileges for the specified domain.

Specified domain: KKDC

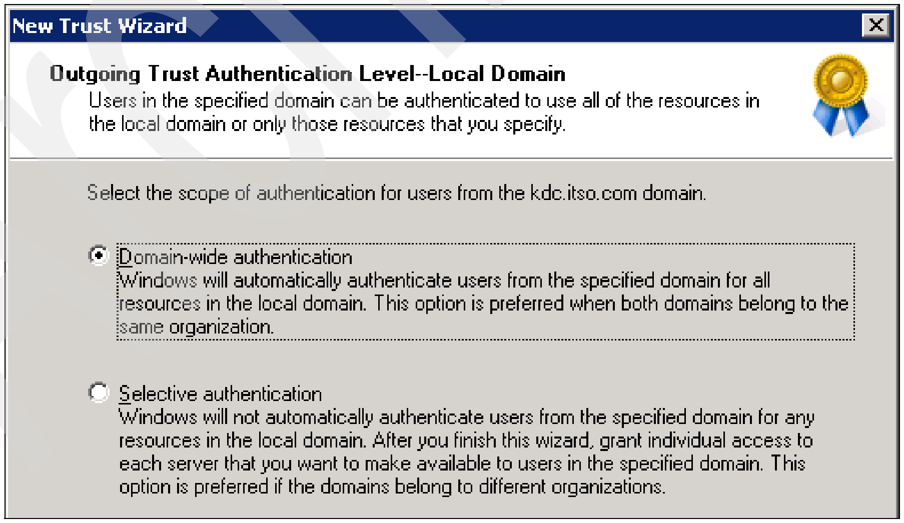
Type the user name and password of an account that has administrative privileges in the specified domain.

User name: administrator

Password: ●●●●●●●●●

Figure 4-27 Providing the administrative credentials

10. For Windows Server 2008, you have the option to select the scope of authentication for users coming from the trusted domain. Selecting the "Domain-Wide authentication" option, as shown in Figure 4-28, enables users from the trusted domain to have access to all resources in the local domain. Click **Next**.



New Trust Wizard

Outgoing Trust Authentication Level--Local Domain
Users in the specified domain can be authenticated to use all of the resources in the local domain or only those resources that you specify.

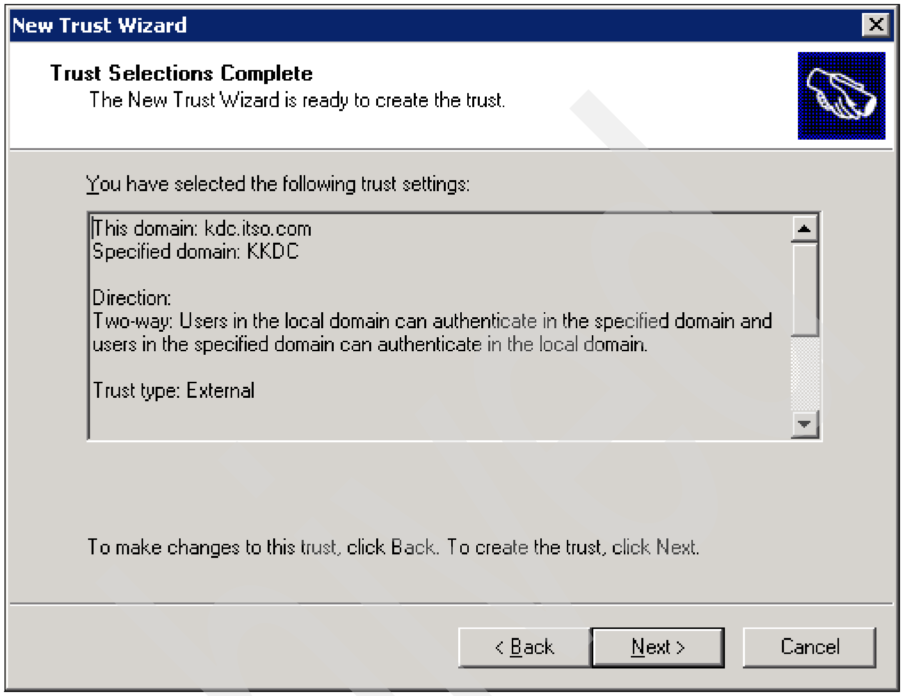
Select the scope of authentication for users from the kdc.itso.com domain.

● Domain-wide authentication
Windows will automatically authenticate users from the specified domain for all resources in the local domain. This option is preferred when both domains belong to the same organization.

○ Selective authentication
Windows will not automatically authenticate users from the specified domain for any resources in the local domain. After you finish this wizard, grant individual access to each server that you want to make available to users in the specified domain. This option is preferred if the domains belong to different organizations.

Figure 4-28 Selecting the scope of authentication for Windows 2008 Server

11. Review the configuration summary, as shown in Figure 4-29, and click **Next** to create the trust.


New Trust Wizard
Trust Selections Complete
The New Trust Wizard is ready to create the trust.
You have selected the following trust settings:

This domain: kdc.itso.com
Specified domain: KKDC

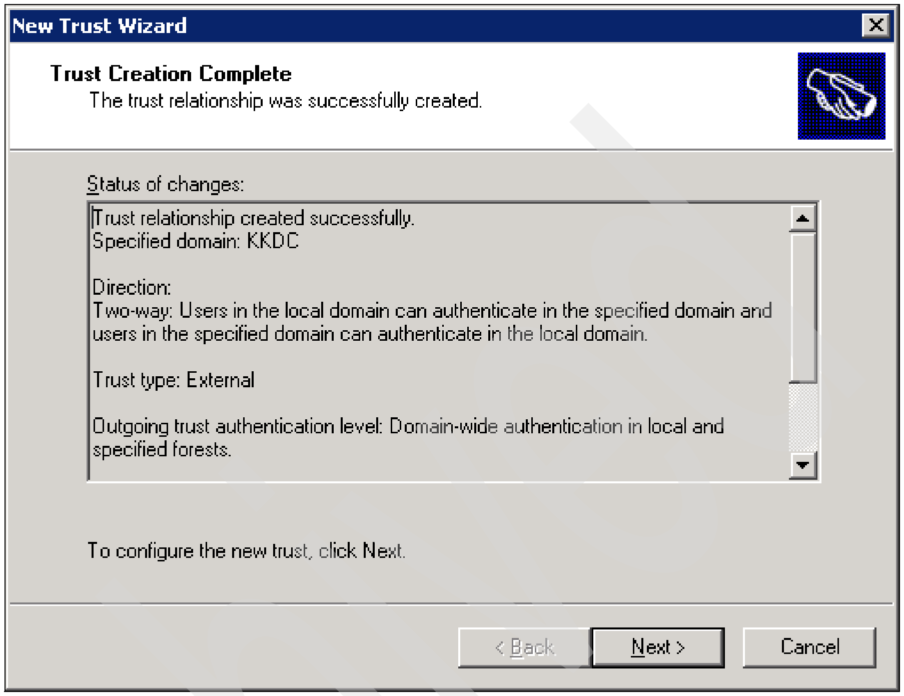
Direction:
Two-way: Users in the local domain can authenticate in the specified domain and users in the specified domain can authenticate in the local domain.

Trust type: External

To make changes to this trust, click Back. To create the trust, click Next.
< Back
Next >
Cancel

Figure 4-29 Creating the trust relationship

12. The trust relationship is created, and the status displays, as shown in Figure 4-30. Click **Next**.


New Trust Wizard

Trust Creation Complete
The trust relationship was successfully created.

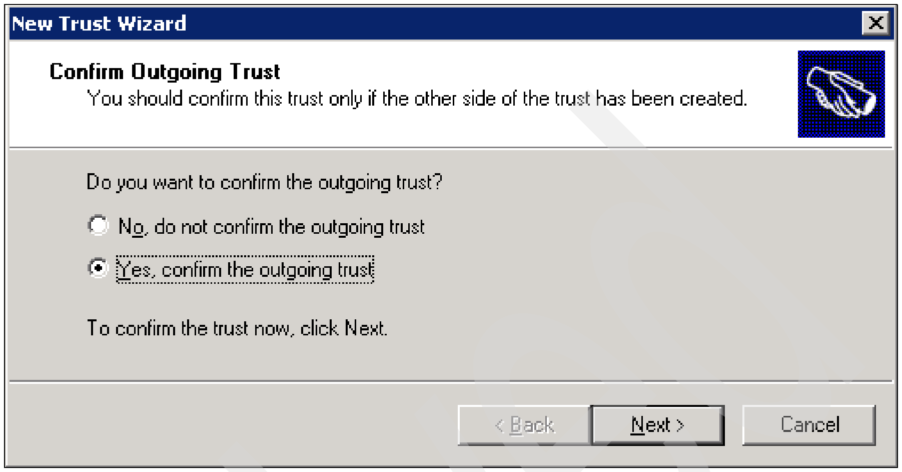
Status of changes:

```
Trust relationship created successfully.
Specified domain: KKDC

Direction:
Two-way: Users in the local domain can authenticate in the specified domain and
users in the specified domain can authenticate in the local domain.

Trust type: External

Outgoing trust authentication level: Domain-wide authentication in local and
specified forests.
```

To configure the new trust, click Next.

Figure 4-30 Trust Creation Complete

13. Next, you confirm the trust relationship. Select **Yes** to confirm outgoing and incoming trusts, as shown in Figure 4-31. Then, click **Next**.



Confirm Outgoing Trust
You should confirm this trust only if the other side of the trust has been created.

Do you want to confirm the outgoing trust?

- No, do not confirm the outgoing trust
- [✓ Yes, confirm the outgoing trust]

To confirm the trust now, click Next.

Figure 4-31 Confirming the outgoing trust relationship

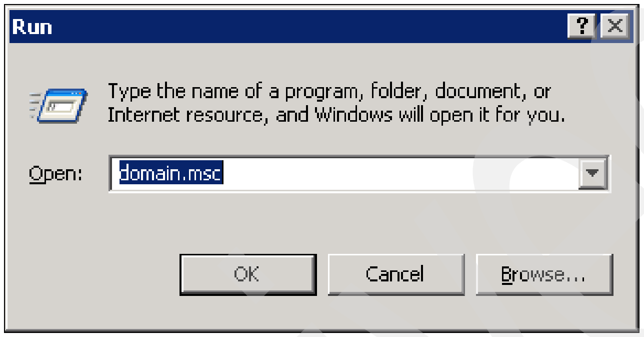
The trust relationship is now created between the realms. You do not need to reboot the servers. The next section explains how to validate the trust relationship.

Note: If you are having any problems with the trust relationship, ensure that the DNS server has the proper information for the realm with which want to create the trust relationship. To check, `ping` the target host using the host name. If the host cannot be found, you have a problem with the DNS configuration.

4.5 Validating the trust relationship

You can use the Active Directory Domain and Trusts console to validate the trusted environment. Using this window, you can validate both outgoing and incoming trusts. Follow these steps:

1. To open the console, select **Start** → **Run**. Type `domain.msc`, and click **OK** as shown in Figure 4-32.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

Open: domain.msc

Figure 4-32 Start the console

2. In the Active Directory Domain and Trusts console, right-click the domain, and select **Properties**, as shown in Figure 4-33.

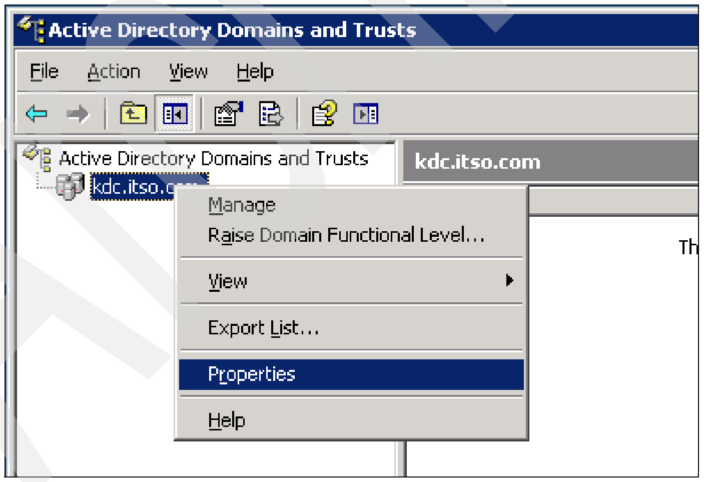
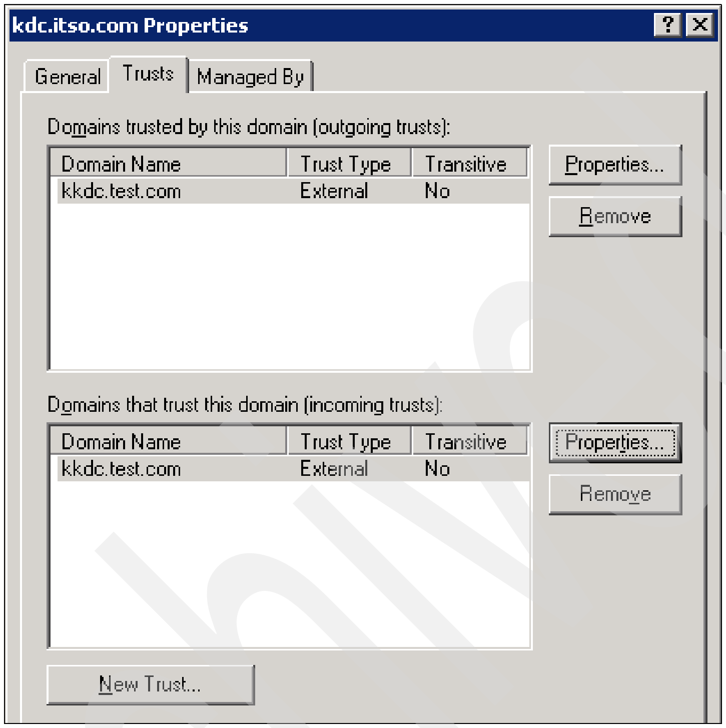

Active Directory Domains and Trusts
File
Action
View
Help
Active Directory Domains and Trusts
kdc.itso.com
kdc.itso.com
Manage
Raise Domain Functional Level...
View
Export List...
Properties
Help
Th

Figure 4-33 Validating realm configuration

3. Select the type of trust that you want to validate (incoming or outgoing). For this example, we validate the *incoming* trust. Click **Properties** to the right of the selection, as shown in Figure 4-34.


kdc.itso.com Properties

General | Trusts | Managed By

Domains trusted by this domain (outgoing trusts):

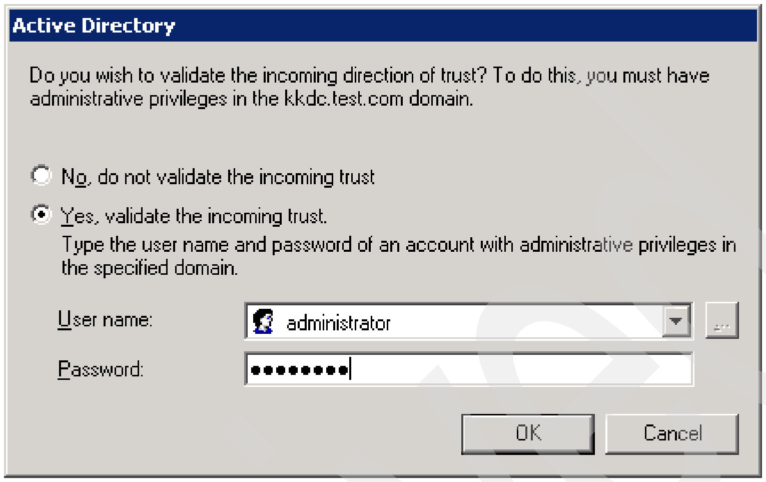
Domain Name	Trust Type	Transitive
kkdc.test.com	External	No

Domains that trust this domain (incoming trusts):

Domain Name	Trust Type	Transitive
kkdc.test.com	External	No

Figure 4-34 Validating incoming trusts

4. In the next window, select **Yes, validate the incoming trust**. Provide the administrative credentials for the target realm, as shown in Figure 4-35. Click **OK**.


Active Directory

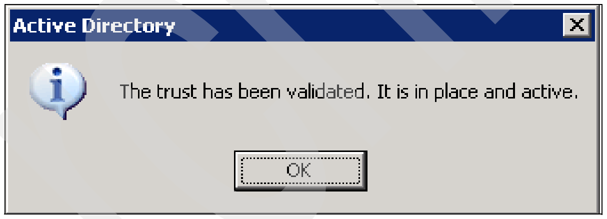
Do you wish to validate the incoming direction of trust? To do this, you must have administrative privileges in the kkdc.test.com domain.

○ No, do not validate the incoming trust
● Yes, validate the incoming trust.
Type the user name and password of an account with administrative privileges in the specified domain.

User name: administrator
Password: ••••••••

Figure 4-35 Providing administrative credentials

5. A message displays with the information about the status of the trust relationship. If everything is working correctly, you see the message as shown in Figure 4-36. Click **OK**.


Active Directory

The trust has been validated. It is in place and active.

Figure 4-36 Trust is validated

4.6 Adding a Kerberos service principal name

The section discusses how to add a new Kerberos service principal name into the Microsoft KDC for Kerberos Services and includes the following topics:

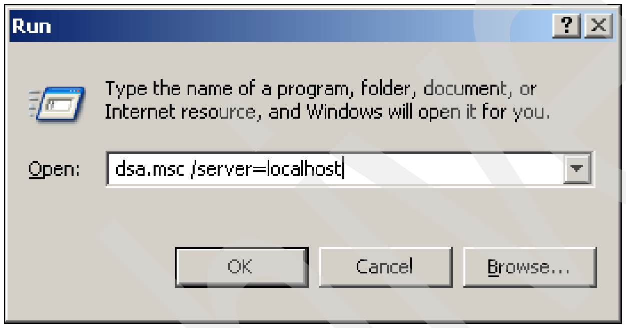
- Creating a Kerberos service principal name
- Creating the Kerberos keytab file

4.6.1 Creating a Kerberos service principal name

First, you need to create a user account in the Microsoft Active Directory. This account is mapped to the Kerberos service principal name (SPN). The SPN identifies the WebSphere Application Server installation as a Kerberized service.

To create this SPN for WebSphere Application Server, follow these steps:

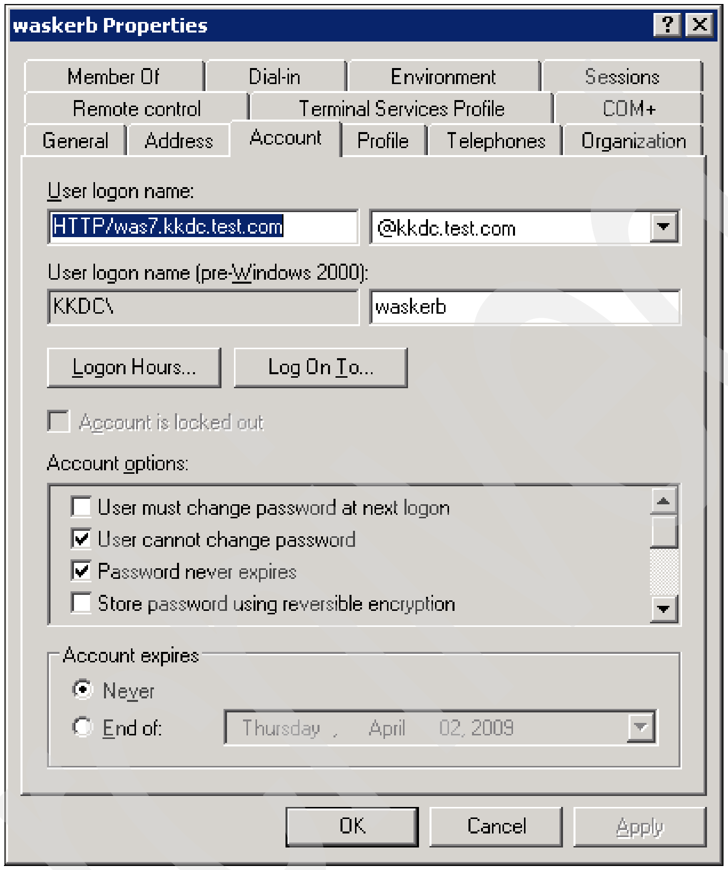
1. Log on as an administrator to the Windows 2003 Server workstation that serves as the domain controller.
2. Invoke the Active Directory users and computers snap-in by selecting **Start** → **Run** and typing `dsa.msc /server=localhost` in the Run window, as shown in Figure 4-37. Click **OK**.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

```
dsa.msc /server=localhost
```

Figure 4-37 Invoking the snap-in

3. Create a user account in the Microsoft Active Directory for WebSphere Application Server. In this scenario, the user account is *waskerb*, as shown in Figure 4-38.


waskerb Properties

Member Of | Dial-in | Environment | Sessions | Remote control | Terminal Services Profile | COM+ | General | Address | Account | Profile | Telephones | Organization

User logon name: HTTP/was7.kkdc.test.com @kkdc.test.com

User logon name (pre-Windows 2000): KKDC\ waskerb

Logon Hours... | Log On To...

☐ 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

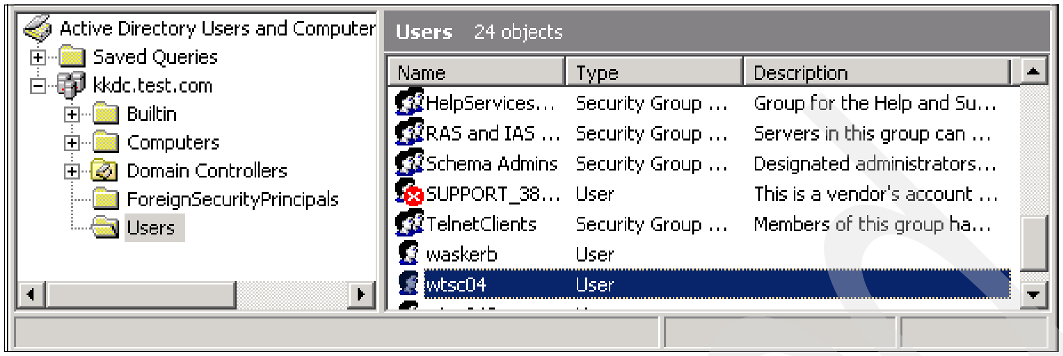
Account expires

● Never
○ End of: Thursday , April 02, 2009

OK | Cancel | Apply

Figure 4-38 The user waskerb properties

The new user account then displays in the list of Windows users and computers, as shown in Figure 4-39.


Active Directory Users and Computer
Saved Queries
kkdc.test.com
Builtin
Computers
Domain Controllers
ForeignSecurityPrincipals
Users

Users 24 objects

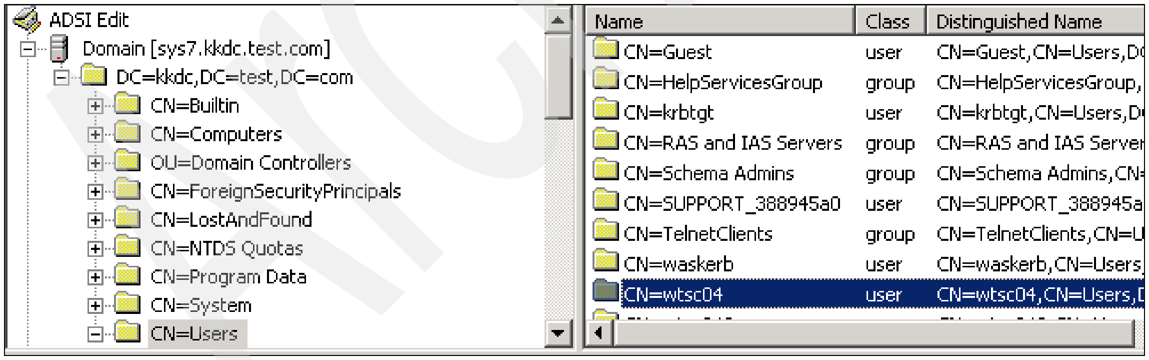
Name	Type	Description
HelpServices...	Security Group ...	Group for the Help and Su...
RAS and IAS ...	Security Group ...	Servers in this group can ...
Schema Admins	Security Group ...	Designated administrators...
SUPPORT_38...	User	This is a vendor's account ...
TelnetClients	Security Group ...	Members of this group ha...
waskerb	User	
wtsc04	User	

Figure 4-39 Windows Active Directory Users and Computers display

4. Invoke the Active Directory service interface snap-in by selecting **Start** → **Run** and then typing `adsiedit.msc` in the Run menu.

> **Note:** To use the Active Directory service interface snap-in (`adsiedit.msc`), the Windows Server Support Tools must be installed.

5. Verify that the new user is created, and check the distinguished name, which in our example is `CN=waskerb,CN=Users,DC=kkdc,DC=test,DC=com`, as shown in Figure 4-40.


ADSI Edit
Domain [sys7.kkdc.test.com]
DC=kkdc,DC=test,DC=com
CN=Builtin
CN=Computers
OU=Domain Controllers
CN=ForeignSecurityPrincipals
CN=LostAndFound
CN=NTDS Quotas
CN=Program Data
CN=System
CN=Users

Name	Class	Distinguished Name
CN=Guest	user	CN=Guest,CN=Users,D
CN=HelpServicesGroup	group	CN=HelpServicesGroup,
CN=krbtgt	user	CN=krbtgt,CN=Users,D
CN=RAS and IAS Servers	group	CN=RAS and IAS Server
CN=Schema Admins	group	CN=Schema Admins,CN=
CN=SUPPORT_388945a0	user	CN=SUPPORT_388945a
CN=TelnetClients	group	CN=TelnetClients,CN=U
CN=waskerb	user	CN=waskerb,CN=Users,
CN=wtsc04	user	CN=wtsc04,CN=Users,D

Figure 4-40 Windows Active Directory Service Interface display

4.6.2 Creating the Kerberos keytab file

Next, after you create the principal name, use the **ktpass** tool to create the Kerberos keytab file and to map it to the Kerberos SPN.

The **ktpass** command allows you to configure a non-Windows Server 2003 Kerberos service (such as WebSphere applications on a z/OS platform) as a security principal in the Windows Server 2003 Active Directory. This tool generates a keytab file that contains the shared key of the service. This keytab file is then transferred to WebSphere Application Server so that the SPNEGO Web authentication can use it.

Use the following **ktpass** command for this task:

```
ktpass -out <output_file> -princ
HTTP/<full_WAS_dns_name>@<windows_Domain> -mapUser <user> -mapOp set
-pass <user_password> -crypto RC4-HMAC-NT
```

For a list of all the options for this command, use **-help**. The following options are the most common options:

-out ***output_file***	Specifies the keytab to produce.
-princ ***principal_name***	Specifies the principal name. The Windows Domain can be found on the Active Directory service interface snap-in.
-mapuser ***user***	Maps the principal to the user.
-pass ***user_password***	Specifies the password to use.
-crypto	Specifies the type of cryptographic system to use. With the *WAS7* Windows account that we set up for the scenarios in this book, RC4-HMAC-NT is selected.

Tip: You can find more information about the keytab files and the ktpass command at:

http://technet2.microsoft.com/WindowsServer/en/library/5090598d-a735-4c73-9e37-1a95a4651fa51033.mspx?mfr=true

Note that there might already be some SPNs related to the Microsoft Windows hosts that have been added to the domain. You can see these SPNs using the **setspn -L *hostname*** command.

You need to add an HTTP SPN for the WebSphere host. Example 4-2 shows the **ktpass** command that we used for the WAS7 system.

Example 4-2 Example output from ktpass utility for Windows host

```
ktpass -out c:\waskerb.keytab -princ
HTTP/was7.kkdc.test.com@KKDC.TEST.COM -mapUser waskerb -mapOp set -pass
* -crypto RC4-HMAC-NT
Targeting domain controller: sys7.kkdc.test.com
Using legacy password setting method
Successfully mapped HTTP/was7.kkdc.test.com to waskerb.
Type the password for HTTP/was7.kkdc.test.com:
Type the password again to confirm:
WARNING: pType and account type do not match. This might cause
problems.
Key created.
Output keytab to c:\waskerb.keytab:
Keytab version: 0x502
keysize 72 HTTP/was7.kkdc.test.com@KKDC.TEST.COM ptype 0
(KRB5_NT_UNKNOWN) vno 5
 etype 0x17 (RC4-HMAC) keylength 16
(0x25cf2dc91be14a3509cbada1743bef84)
```

In this example, not that the Kerberos realm name needs to be in uppercase letters after the Active Directory domain name. Also, we choose the `RC4-HMAC-NT` encryption algorithm.

Important: Microsoft Windows Active Directory supports two different Kerberos encryption types:

- RC4-HMAC-NT
- DES-CBC-MD5

IBM Java Generic Security Service (JGSS) library (and SPNEGO library) support both of these encryption types. RC4-HMAC-NT encryption is supported only with a Windows 2003 Server key distribution center. RC4-HMAC-NT encryption is *not* supported when using a Windows 2000 Server as a Kerberos KDC.

If a warning regarding the account type and the ptype is issued, you can safely ignore it.

Use the **setspn** command to verify the new SPN list, as shown in Example 4-3.

Example 4-3 Checking SPNs using -L flag

```
C:\>setspn.exe -l waskerb
Registered ServicePrincipalNames for
CN=waskerb,CN=Users,DC=kkdc,DC=test,DC=com:

HTTP/was7.kkdc.test.com
```

Attention: Map the HTTP service name to only one host name. If there is more than one entry, remove the user and re-create the keytab again

Tip: For more information about the **setspn** tool, see:

http://technet2.microsoft.com/WindowsServer/en/library/318642de-15a3-4478-beb0-8022696def511033.mspx?mfr=true

5

Setting up trust between an AIX KDC and a z/OS KDC

This chapter describes how to configure a cross-realm trust relationship between an AIX key distribution center (KDC) and a z/OS KDC. This process includes the following major steps:

1. Setting up the AIX KDC for trust
2. Setting up the z/OS KDC for trust
3. Verifying the cross-realm trust setup

In this chapter, we discuss the following topics:

- Introduction to cross-realm trust
- Configuring the cross-realm trust relationship

5.1 Introduction to cross-realm trust

As an organization expands, it introduces more departments, servers, and client workstations. The Kerberos administrator might decide to create separate Kerberos realms for each new department for ease of management.

In addition, there is an unavoidable need to access the Kerberos service that is running on a foreign realm. In such situations, the Kerberos administrator needs to configure how the users from one Kerberos realm can access any service running in another realm. This kind of configuration is called *cross-realm trust* configuration.

Although the process to configure a cross-realm trust can be involved, the information in this chapter can help. This chapter discusses how to set up a cross-realm trust between an AIX KDC and a z/OS KDC, including all the necessary steps and examples. The process that we outline in this chapter makes the following basic assumptions:

- The AIX KDC is already installed and configured. For more information, refer Chapter 3, "Configuring IBM Network Authentication Service KDC on AIX" on page 29.
- The z/OS KDC is already installed and configured. For more information, refer Chapter 2, "Setting up a KDC on a z/OS system" on page 13.
- The time on all the participating systems is synchronized.
- The reader has a prior understanding of the Kerberos realm and its configuration and management.

Table 5-1 provides an overview of the basic Kerberos configuration of both KDCs for this example.

Table 5-1 Configuration of the KDCs involved in the trust relationship

	AIX (IBM NAS) KDC	**z/OS KDC**
Realm Name	ITSO.RAL.IBM.COM	ITSO.IBM.COM
Database Type	LDAP	SAF
Host name	saw921-sys6.itso.ral.ibm.com	wtsc60.itso.ibm.com
TGT Service principal	krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM	krbtgt/ITSO.IBM.COM@ITSO.IBM.COM

5.2 Configuring the cross-realm trust relationship

To configure the cross-realm trust relationship, you need to complete a series steps on each KDC. We describe those steps in detail in this section.

5.2.1 Configuring the AIX KDC for cross-realm trust

To configure the AIX KDC for cross-realm trust, you need to complete the following basic steps:

1. Add z/OS realm information to the AIX KDC LDAP database.
2. Create the cross-realm TGT service principals.
3. Update the `krb5.conf` file.

Add z/OS realm information to the AIX KDC LDAP database

Note: This step is required only when using an LDAP database for the AIX KDC. If you are not using an LDAP database, then you can skip this section and go directly to "Create the cross-realm TGT service principals" on page 93.

To establish trust between the AIX KDC and the z/OS KDC, you need to add the z/OS realm information in the AIX KDC LDAP database. Remember that you used the `realm_add.ldif` file to add the AIX KDC realm information (as described in 3.2.3, "Add Kerberos realm information in IBM Tivoli Directory Server" on page 35). You use the same file to add the z/OS KDC realm information.

To add z/OS realm information to the AIX KDC LDAP database, follow these steps:

1. Copy the `realm_add.ldif` file to a new location. Change the name of the file to `realm_add.ldif.zOS`, and make the following updates:
   - Change all occurrences of the realm name to the z/OS realm name.
   - Change the `krbDeleteType` value to 4.

   Example 5-1 shows the new files with the changes noted in bold text.

   Example 5-1 The updated realm_add.ldif.zOS

```
dn: ou=raleigh,o=ibm,c=us
ou: Raleigh
objectclass: organizationalUnit


dn: krbrealmName-V2=ITSO.IBM.COM, ou=raleigh,o=ibm,c=us
```

```
objectclass: KrbRealm-V2
objectclass: KrbRealmExt
krbrealmName-V2: ITSO.IBM.COM
krbprincSubtree: krbrealmName-V2=ITSO.IBM.COM, ou=raleigh,o=ibm,c=us
krbDeleteType: 4
krbMaxFailAuth: 0
krbDisableTimeInterval: 0


dn: cn=principal, krbrealmName-V2=ITSO.IBM.COM,
ou=raleigh,o=ibm,c=us
objectclass: container
cn: principal

dn: cn=policy, krbrealmName-V2=ITSO.IBM.COM, ou=raleigh,o=ibm,c=us
objectclass: container
cn: policy
```

2. Add the information in the file to the LDAP database using the **ldapadd** command provided by LDAP, as shown in Example 5-2.

Example 5-2 Adding z/OS realm information in the LDAP database

```
[saw921-sys6][/usr/krb5/ldif]# ldapadd -h sys4.itso.ral.ibm.com -D
cn=root -w adminpwd01 -f /usr/krb5/ldif/realm_add.ldif.zOS -v -c
ldap_init(sys4.itso.ral.ibm.com, 389)
add ou:
        BINARY (7 bytes) Raleigh
add objectclass:
        BINARY (18 bytes) organizationalUnit
Operation 0 adding new entry ou=raleigh,o=ibm,c=us
ldap_add: Already exists

add objectclass:
        BINARY (11 bytes) KrbRealm-V2
        BINARY (11 bytes) KrbRealmExt
add krbrealmName-V2:
        BINARY (12 bytes) ITSO.IBM.COM
add krbprincSubtree:
        BINARY (51 bytes) krbrealmName-V2=ITSO.IBM.COM,
ou=raleigh,o=ibm,c=us
add krbDeleteType:
        BINARY (1 bytes) 4
add krbMaxFailAuth:
        BINARY (1 bytes) 0
```

```
add krbDisableTimeInterval:
        BINARY (1 bytes) 0
Operation 1 adding new entry krbrealmName-V2=ITSO.IBM.COM,
ou=raleigh,o=ibm,c=us

add objectclass:
        BINARY (9 bytes) container
add cn:
        BINARY (9 bytes) principal
Operation 2 adding new entry cn=principal,
krbrealmName-V2=ITSO.IBM.COM, ou=raleigh,o=ibm,c=us

add objectclass:
        BINARY (9 bytes) container
add cn:
        BINARY (6 bytes) policy
Operation 3 adding new entry cn=policy,
krbrealmName-V2=ITSO.IBM.COM, ou=raleigh,o=ibm,c=us
```

After you add the realm information, you next create the cross-realm TGT service principals.

Create the cross-realm TGT service principals

For the AIX KDC to be able to issue an initial ticket on behalf of the z/OS counterpart, you need to create the cross-realm TGT service principals. You create these principals in both realms with same passwords. Notice that the z/OS KDC always converts the password to uppercase characters. Therefore, keep the password in uppercase on both the KDCs.

Note: Keeping the same password for cross-realm TGT service principals in both realms is very important for trust establishment.

The naming convention for the cross-realm TGT service principal name is:

- `krbtgt/<X.REALM.COM>@<Y.REALM.COM>`
- `krbtgt/<Y.REALM.COM>@<X.REALM.COM>`

Therefore, the principals that you create are as follows:

- `krbtgt/ITSO.RAL.IBM.COM@ITSO.IBM.COM`
- `krbtgt/ITSO.IBM.COM@ITSO.RAL.IBM.COM`

Example 5-3 shows how to create these principles in the AIX KDC. For more information, refer to 3.3.2, “Creating a Kerberos principal” on page 42.

Example 5-3 Creating the cross-realm TGT service principals

```
kadmin:  addprinc -pw <pswd> krbtgt/ITSO.RAL.IBM.COM@ITSO.IBM.COM
WARNING: no policy specified for krbtgt/ITSO.RAL.IBM.COM@ITSO.IBM.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "krbtgt/ITSO.RAL.IBM.COM@ITSO.IBM.COM" created.

kadmin:  addprinc -pw <pswd> krbtgt/ITSO.IBM.COM@ITSO.RAL.IBM.COM
WARNING: no policy specified for krbtgt/ITSO.IBM.COM@ITSO.RAL.IBM.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "krbtgt/ITSO.IBM.COM@ITSO.RAL.IBM.COM" created.
kadmin:  q
```

Remember the passwords that you just entered, because you will need them again while creating these principals on the z/OS KDC.

Update the krb5.conf file

Next, you need to update the `/etc/krb5/krb5.conf` file with the z/OS KDC realm information. Open this file, and add the z/OS realm information under the `[realms]` and `[domain_realm]` sections. Example 5-4 shows the file after the changes, which are shown in bold font.

Example 5-4 The krb5.conf on AIX KDC updated with z/OS KDC information

```
[libdefaults]
        default_realm = ITSO.RAL.IBM.COM
        default_keytab_name = FILE:/etc/krb5/krb5.keytab
        default_tkt_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts
des-cbc-md5 des-cbc-crc
        default_tgs_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts
des-cbc-md5 des-cbc-crc

[realms]
        ITSO.RAL.IBM.COM = {
                kdc = saw921-sys6.itso.ral.ibm.com:88
                admin_server = saw921-sys6.itso.ral.ibm.com:749
                default_domain = itso.ral.ibm.com
                vdb_plugin_lib = /usr/lib/libkrb5ldplug.a
        }
```

```
# This is for cross-realm z/OS KDC.
        ITSO.IBM.COM = {
                kdc = wtsc60.itso.ibm.com:88
                admin_server = wtsc60.itso.ibm.com:749
        }

[domain_realm]
        .itso.ral.ibm.com = ITSO.RAL.IBM.COM
        saw921-sys6.itso.ral.ibm.com = ITSO.RAL.IBM.COM
        wtsc60.itso.ibm.com = ITSO.IBM.COM

[logging]
        kdc = FILE:/var/krb5/log/krb5kdc.log
        admin_server = FILE:/var/krb5/log/kadmin.log
        default = FILE:/var/krb5/log/krb5lib.log
```

Be sure to re-start the IBM NAS daemons for the changes to take effect. For more information, refer to 3.3.6, “Starting and stopping IBM NAS Kerberos daemons” on page 48.

5.2.2 Configuring the z/OS KDC for cross-realm trust

To configure the z/OS KDC for cross-realm trust, you need to complete the following basic steps:

1. Create the cross-realm TGT service principals.
2. Update the `krb5.conf` file.

Create the cross-realm TGT service principals

In the z/OS KDC, first you need to create the cross-realm TGT service principals. (Remember that you need to create the same two TGT service principals with the same upper case password as you did for the AIX KDC.) In the z/OS KDC, create the principals as shown in Example 5-5.

Example 5-5 Creating cross-realm TGT principal on z/OS KDC

```
RDEFINE REALM /.../ITSO.RAL.IBM.COM/KRBTGT/ITSO.IBM.COM
KERB(PASSWORD(<pswd>) ENCRYPT(DES NODESD DES3 AES128))

RDEFINE REALM /.../ITSO.IBM.COM/KRBTGT/ITSO.RAL.IBM.COM
KERB(PASSWORD(<pswd>) ENCRYPT(DES NODESD DES3 AES128))
```

For more information about how to create principals in the z/OS KDC, refer to 2.3.1, “Creating the Kerberos service principal name” on page 19.

Note: For the cross-realm setup to work, the key version number and the encryption keys of the TGT service principals must be same on both the KDCs. If they are not same, you must edit the principal details and make them exactly the same.

Update the krb5.conf file

Next, you need to update the `krb5.conf` file for the z/OS KDC with the AIX KDC realm information. You need to add the information under the `[realms]` and `[domain_realm]` sections in this file. Example 5-6 shows the `krb5.conf` file after the changes, which are shown bold font.

Example 5-6 The krb5.conf on z/OS KDC updated with AIX KDC information

```
[libdefaults]

default_realm = ITSO.IBM.COM
default_keytab_name = FILE:/etc/skrb/krb5.keytab
kdc_default_options = 0x40000010
use_dns_lookup = 0

default_tkt_enctypes = des-cbc-md5
default_tgs_enctypes = des-cbc-md5

[realms]

ITSO.IBM.COM = {
    kdc = wtsc60.itso.ibm.com:88
    kpasswd_server = wtsc60.itso.ibm.com:464
    default_domain = itso.ibm.com
}

ITSO.RAL.IBM.COM = {
    kdc = saw921-sys6.itso.ral.ibm.com:88
    kpasswd_server = saw921-sys6.itso.ral.ibm.com:464
    default_domain = itso.ral.ibm.com
}

[domain_realm]

.itso.ibm.com = ITSO.IBM.COM
wtsc60.itso.ibm.com = ITSO.IBM.COM
.itso.ral.ibm.com = ITSO.RAL.IBM.COM
saw921-sys6.itso.ral.ibm.com = ITSO.RAL.IBM.COM
```

Be sure to re-start the z/OS KDC after you make these changes.

Now, the cross-realm trust setup is complete, and you can verify it.

5.2.3 Verifying the cross-realm trust setup

At a minimum, after you set up the cross-realm trust, you need to verify that a client in one realm can get an initial ticket and a service ticket from other realm.

Getting an initial ticket

For testing purposes, we created one principal called *REDBOOK* in the z/OS KDC. This test attempts to get the initial ticket for REDBOOK from the AIX KDC, as shown in Example 5-7.

Example 5-7 Getting an initial ticket for the z/OS KDC user from the AIX KDC

```
[saw921-sys6][/]# /usr/krb5/bin/kinit REDBOOK@ITSO.IBM.COM
Password for REDBOOK@ITSO.IBM.COM: <---- ENTER PASSWORD HERE

[saw921-sys6][/]# /usr/krb5/bin/klist -e
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  REDBOOK@ITSO.IBM.COM

Valid starting     Expires            Service principal
07/16/09 12:44:47  07/17/09 12:43:46  krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
        Etype (skey, tkt):  AES-256 CTS mode with 96-bit SHA-1 HMAC,
DES cbc mode with RSA-MD5

[saw921-sys6][/]# /usr/krb5/bin/kdestroy

[saw921-sys6][/]#
```

Notice that in the **klist** output in Example 5-7, the initial ticket is for the `REDBOOK@ITSO.IBM.COM` is a z/OS principal, and the TGT principal (`krbtgt/ITSO.IBM.COM@ITSO.IBM.COM`) is again from the z/OS realm. The **kdestroy** command cleans up the ticket that you have just acquired.

Getting a service ticket

The real test of a cross-realm trust setup is to receive a service ticket from the other realm. To acquire a service ticket in this example, we used the **kvno** command provided by IBM Network Authentication Service. The **kvno** command performs a TGS request for the specified principal and shows the current key version number of the KDC. An initial ticket of the current realm must be present for the **kvno** command to work.

To verify that you can receive a service ticket:

1. First, get an initial ticket for the current AIX realm, as shown in Example 5-8.

Example 5-8 Getting initial ticket for current realm

```
[saw921-sys6][/]# /usr/krb5/bin/kinit admin/admin
Password for admin/admin@ITSO.RAL.IBM.COM: <-- ENTER PASSWORD HERE

[saw921-sys6][/]# /usr/krb5/bin/klist -e
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  admin/admin@ITSO.RAL.IBM.COM

Valid starting     Expires            Service principal
07/16/09 13:07:32  07/17/09 13:07:30  krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
        Etype (skey, tkt):  Triple DES cbc mode with HMAC/sha1, Triple DES cbc mode
with HMAC/sha1
```

2. Now, issue the **kvno** command for the TGT principal of the z/OS KDC, `krbtgt/ITSO.IBM.COM@ITSO.IBM.COM`, as shown in Example 5-9.

Example 5-9 Getting a service ticket by using kvno command.

```
[saw921-sys6][/]# /usr/krb5/bin/kvno krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
kvno = 3
```

If this command fails, refer to 18.4, "Common problems when configuring Kerberos trust realms" on page 482.

After you run the **kvno** command successfully, you should be able to see the ticket using the **klist** command. Example 5-10 shows the service tickets for all three TGT principals.

Example 5-10 Examining the service ticket using klist

```
[saw921-sys6][/]# /usr/krb5/bin/klist -e
Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
Default principal:  admin/admin@ITSO.RAL.IBM.COM

Valid starting     Expires            Service principal
07/16/09 13:07:32  07/17/09 13:07:30  krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
        Etype (skey, tkt):  Triple DES cbc mode with HMAC/sha1, Triple DES cbc mode
with HMAC/sha1
07/16/09 13:07:44  07/17/09 13:07:30  krbtgt/ITSO.IBM.COM@ITSO.RAL.IBM.COM
        Etype (skey, tkt):  Triple DES cbc mode with HMAC/sha1, Triple DES cbc mode
with HMAC/sha1
07/16/09 13:08:42  07/17/09 13:07:30  krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
```

```
        Etype (skey, tkt):  Triple DES cbc mode with HMAC/sha1, Triple DES cbc mode
with HMAC/sha1
[saw921-sys6][/]#
```

Just a quick review of the role of these three service principals:

- `krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM`

  The default TGT service for AIX KDC realm (ITSO.RAL.IBM.COM).

- `krbtgt/ITSO.IBM.COM@ITSO.RAL.IBM.COM`

  The cross-realm TGT principal

- `krbtgt/ITSO.IBM.COM@ITSO.IBM.COM`

  The default TGT service for z/OS KDC realm (ITSO.IBM.COM).

Finally, if you can run the **kvno** command successfully and can receive all three TGT principals as describe in this section, then the cross-realm trust setup between the AIX KDC and the z/OS KDC is working correctly.

Setting up trust between a Microsoft Kerberos KDC and a z/OS KDC

This chapter describes how to create the trust relationship between a Microsoft Kerberos key distribution center (KDC) and a z/OS Kerberos KDC. This process includes configuring the following components:

1. Trust on the Microsoft KDC
2. The Microsoft Windows client
3. Trust on the z/OS KDC

This chapter includes the following topics:

- Introduction to the Microsoft Kerberos KDC and the z/OS KDC cross-realm trust
- Configuring trust on the Microsoft KDC
- Configuring the Microsoft Windows client
- Configuring trust on the z/OS KDC
- Validating the trust

6.1 Introduction to the Microsoft Kerberos KDC and the z/OS KDC cross-realm trust

When you create a relationship between a Microsoft Kerberos KDC and a z/OS KDC, you create a two-way trust relationship between the two realms. This trust enables users who are authenticated in the Active Directory domain to access WebSphere applications running on z/OS.

The following components are assumed to be installed before you begin the configuration of this environment:

- Microsoft Active Directory is installed and configured. This system must be a domain controller and domain name server (DNS). Refer to Chapter 4, “Setting up Microsoft Active Directory and Kerberos KDC” on page 51 for information about setting up a Microsoft Active Directory.
- The Kerberos realm for Microsoft Active Directory is the name of the Active Directory domain in uppercase letters.
- A Microsoft Windows client that belongs to the Active Directory domain is installed and configured.
- The z/OS KDC system is installed, configured, and started. Refer to Chapter 2, “Setting up a KDC on a z/OS system” on page 13 for information about setting up the z/OS KDC.
- The z/OS KDC system realm name is defined in uppercase letters to in order to configure it for trust with Active Directory.
- The time is synchronized on all systems.
- The reader of this information has a basic understanding of z/OS system administration.

Table 6-1 lists the system information for the systems and KDCs that we used for the examples in this chapter. The table includes the host name and realm information for each system. Note that the realm names are in uppercase letters.

Table 6-1 System information

	Windows Active Directory	Windows Client	z/OS KDC
Operating system	Windows 2008	Windows 2003	z/OS v1r9
Host name	saw921-sys8.itso.ral.ibm.com	saw921-sys9.itso.ral.ibm.com	wtsc60.itso.ibm.com

	Windows Active Directory	Windows Client	z/OS KDC
IP address	9.42.170.222	9.42.170.223	9.12.4.18
Realm name	KDC2008.ITSO.COM	Member of KDC2008.ITSO.COM realm	ITSO.IBM.COM

6.2 Configuring trust on the Microsoft KDC

Complete the following basic steps to configure the trust relationship on the Microsoft Kerberos KDC:

1. Install the prerequisites.
2. Configure the location of the z/OS KDC.
3. Configure trust to the z/OS KDC.
4. Configure the DNS.

We describe these steps in detail in the sections that follow.

6.2.1 Install the prerequisites

For the Windows domain controller, you need the following prerequisites for Windows 2003:

- Windows Server 2003 Service Pack 1 or higher
- Windows Server 2003 Resource Kit Tools
- Windows Server 2003 Service Pack 1 Support Tools

You need the following prerequisites for Windows 2008:

- Windows Server 2008 Resource Kit Tools
- Windows Server 2008 Support Tools

The Windows Server Resource Kit Tools include the `klist.exe` executable. You can use the **`klist`** command to view and delete Kerberos tickets that are granted to the current logon session. The tools also include the `kerbtray.exe` executable, which adds a Kerberos Tray tool icon to the desktop toolbar. You can use the Kerberos Tray tool to display ticket information.

The Windows Server Support Tools include the `ksetup.exe` executable. You can use the **`ksetup`** command to add a Kerberos realm entry by specifying the KDC server and kpasswd server. In this chapter the z/OS KDC realm entry is added on the Active Directory system.

6.2.2 Configure the location of the z/OS KDC

Using the **ksetup** command, you can configure the Microsoft KDC so that it can connect to the z/OS KDC server and kpasswd server.

Run the following commands from a Windows command prompt:

```
ksetup /AddKdc <RealmName> [KdcName]
ksetup /AddKpasswd <Realmname> <KpasswdName>
```

Specify the z/OS KDC realm for *RealmName* and the z/OS host name for *KdcName* and *KpasswdName*, as shown in Example 6-1.

Example 6-1 Using ksetup with /AddKdc and /AddKpasswd

```
ksetup /AddKdc ITSO.IBM.COM wtsc60.itso.ibm.com
ksetup /AddKpasswd ITSO.IBM.COM wtsc60.itso.ibm.com
```

Note that if you run **ksetup /AddKdc** without the *KdcName*, DNS is used to locate the KDC.

6.2.3 Configure trust to the z/OS KDC

To configure the trust relationship to the z/OS KDC realm, complete the following steps on the Active Directory system:

1. Open Active Directory Domains and Trusts by clicking **Start** → **Programs** → **Administrative Tools** → **Active Directory Domains and Trusts**.
2. Look for the Windows domain. Right-click the domain, and click **Properties**, as shown in Figure 6-1.

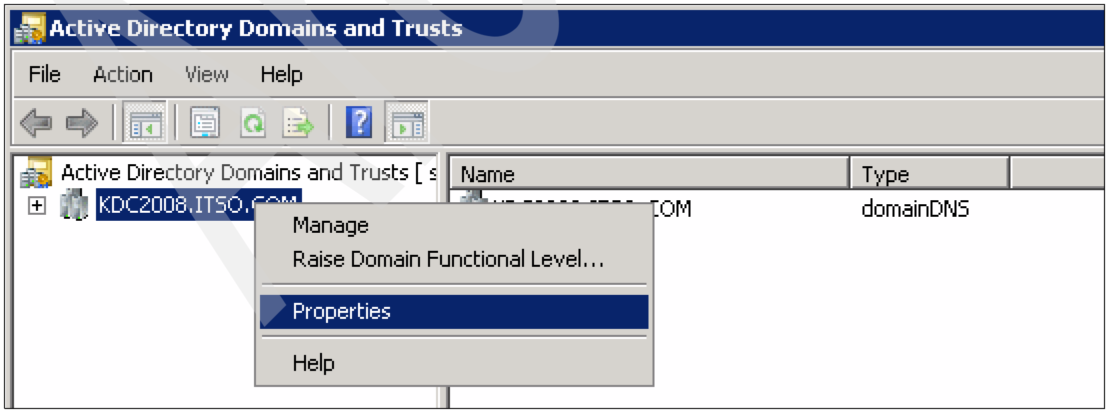

Active Directory Domains and Trusts
File
Action
View
Help
Active Directory Domains and Trusts [s
KDC2008.ITSO.COM
Name
Type
domainDNS
Manage
Raise Domain Functional Level...
Properties
Help

Figure 6-1 Active Directory Domains and Trust window

3. Go to the Trusts tab, and then click **New Trust** to add a realm to trust, as shown in Figure 6-2.

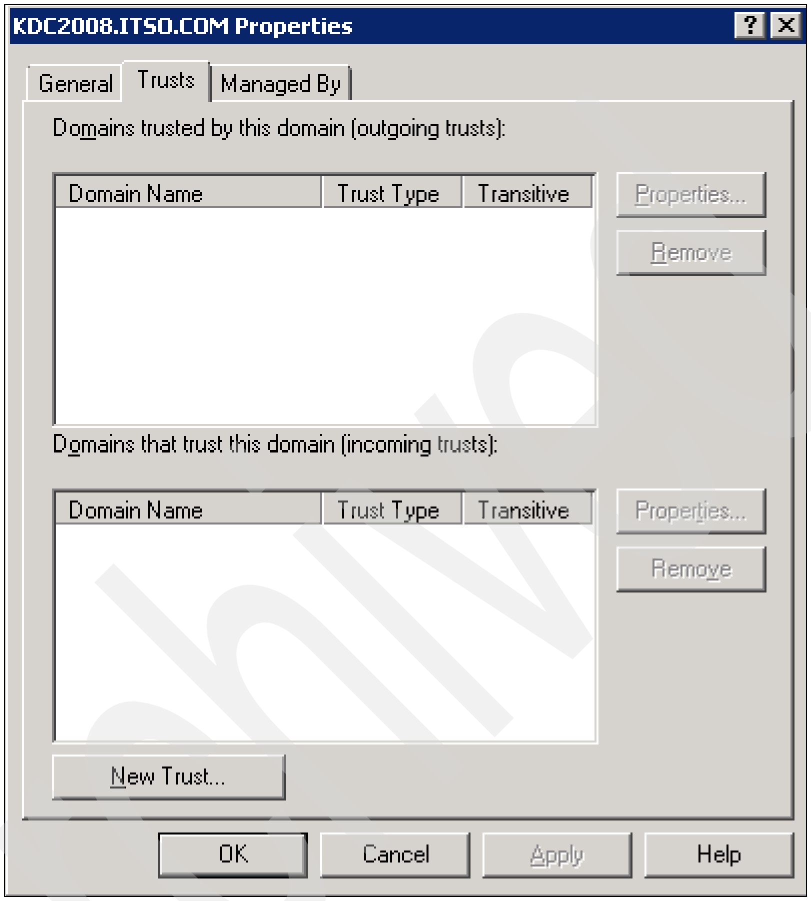
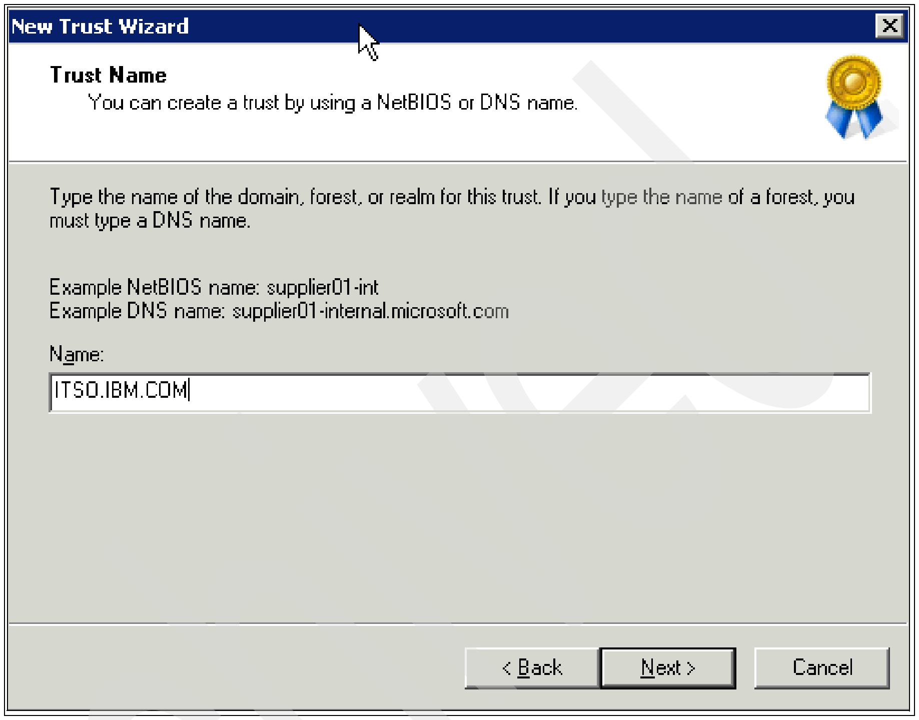


Figure 6-2 KDC2008.ITSO.COM Properties panel

4. Click **Next** on the first panel of the New Trust wizard.

5. For the Trust Name, enter the z/OS realm name in uppercase letters, and then click **Next**. In this example, the z/OS realm name is `ITSO.IBM.COM`, as shown in Figure 6-3.


New Trust Wizard

Trust Name

You can create a trust by using a NetBIOS or DNS name.

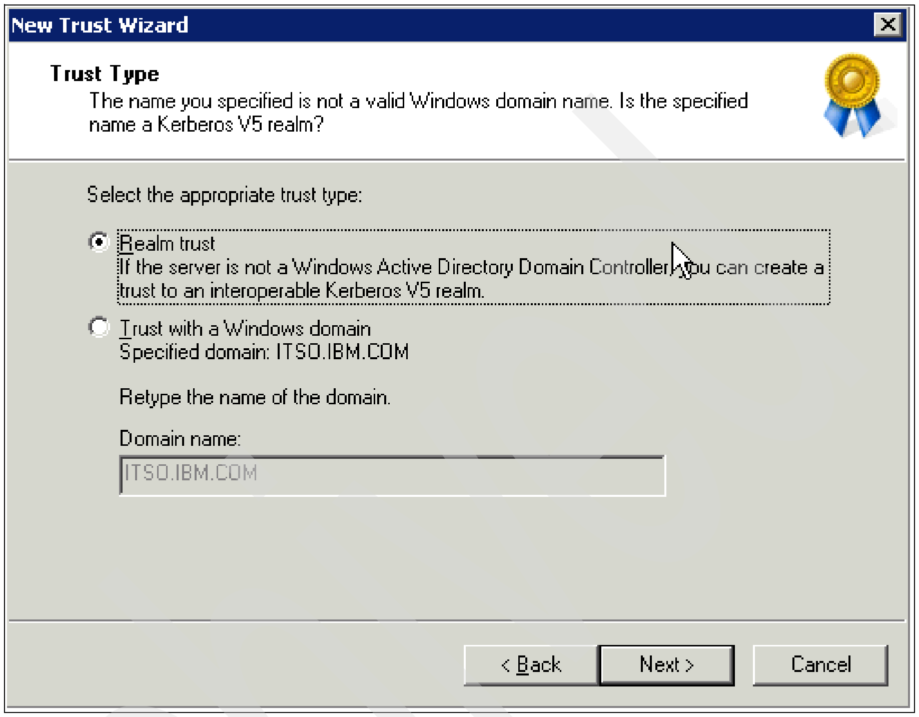
Type the name of the domain, forest, or realm for this trust. If you type the name of a forest, you must type a DNS name.

Example NetBIOS name: supplier01-int
Example DNS name: supplier01-internal.microsoft.com

Name: ITSO.IBM.COM

Figure 6-3 New Trust Wizard, Trust Name panel

6. For the Trust Type, click **Realm trust**, and then click **Next**, as shown in Figure 6-4.



New Trust Wizard

Trust Type

The name you specified is not a valid Windows domain name. Is the specified name a Kerberos V5 realm?

Select the appropriate trust type:

[✓ Realm trust]
If the server is not a Windows Active Directory Domain Controller, you can create a trust to an interoperable Kerberos V5 realm.

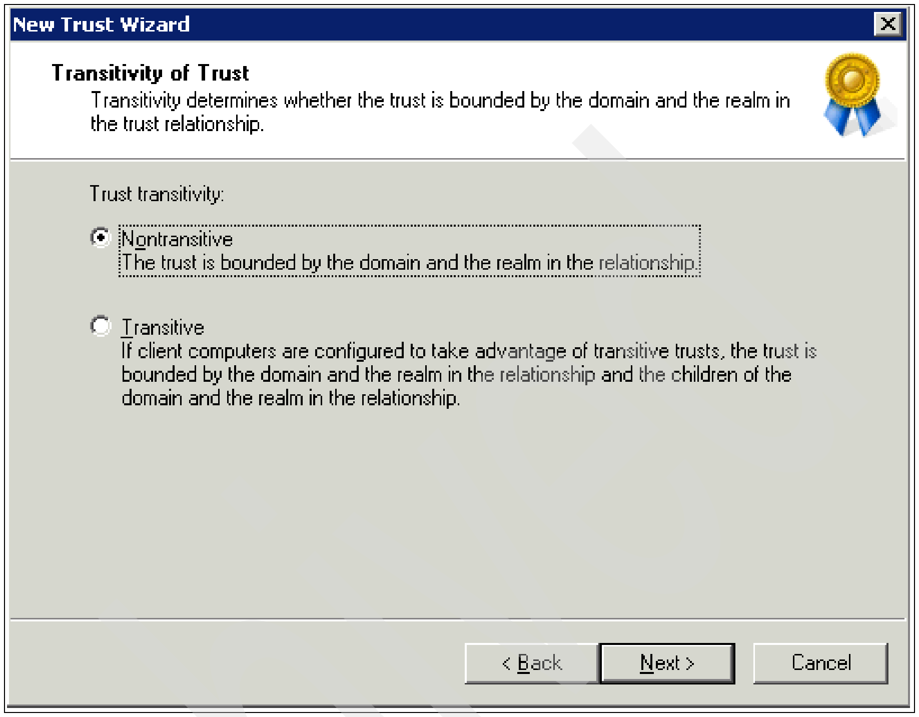
Trust with a Windows domain
Specified domain: ITSO.IBM.COM

Retype the name of the domain.

Domain name: ITSO.IBM.COM

Figure 6-4 New Trust Wizard, Trust Type panel

7. For the trust transitivity, click **Nontransive**, and click **Next** (Figure 6-5).



New Trust Wizard

Transitivity of Trust

Transitivity determines whether the trust is bounded by the domain and the realm in the trust relationship.

Trust transitivity:

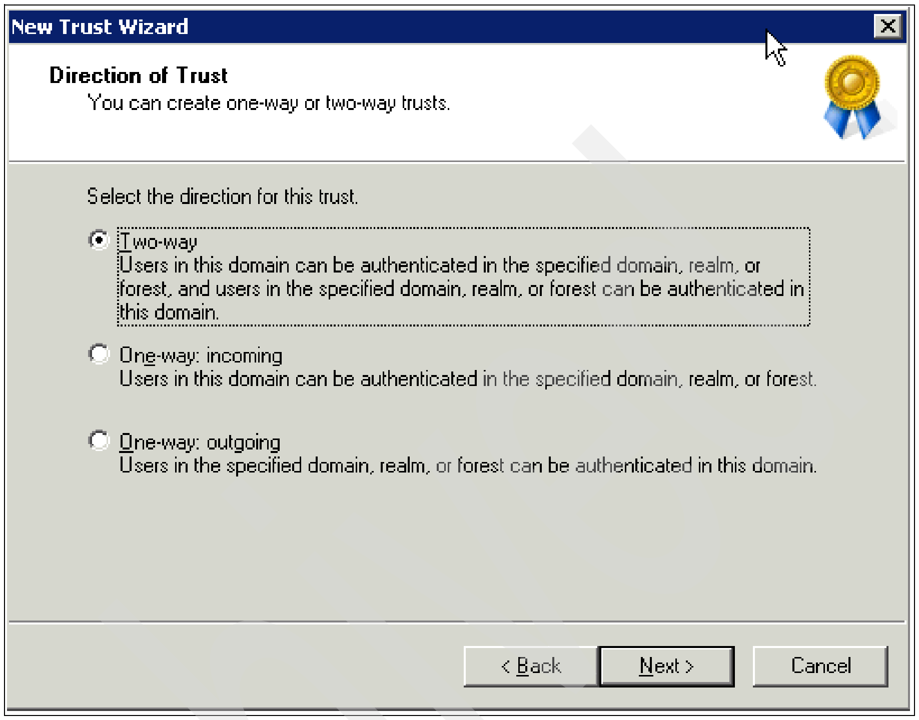
[✓ Nontransitive]
The trust is bounded by the domain and the realm in the relationship.

Transitive
If client computers are configured to take advantage of transitive trusts, the trust is bounded by the domain and the realm in the relationship and the children of the domain and the realm in the relationship.

< Back | Next > | Cancel

Figure 6-5 New Trust Wizard, Transitivity of Trust panel

8. For the trust direction, click **Two-way,** and click **Next** (Figure 6-6).


New Trust Wizard

Direction of Trust

You can create one-way or two-way trusts.

Select the direction for this trust.

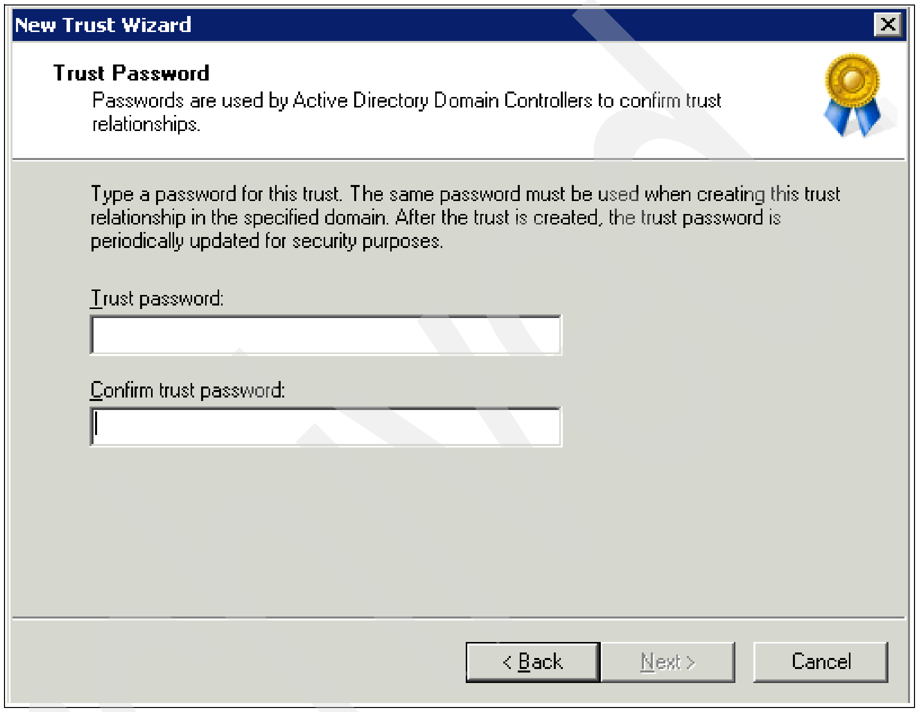
[✓ Two-way]
Users in this domain can be authenticated in the specified domain, realm, or forest, and users in the specified domain, realm, or forest can be authenticated in this domain.

One-way: incoming
Users in this domain can be authenticated in the specified domain, realm, or forest.

One-way: outgoing
Users in the specified domain, realm, or forest can be authenticated in this domain.

Figure 6-6 New Trust Wizard, Direction of Trust panel

9. For the Trust Password window, shown in Figure 6-7, enter a password in uppercase letters, and then click **Next**. Because this same password is used by the RACF RDEFINE command when configuring trust on the z/OS KDC, the RDEFINE command converts the characters to uppercase. The trust is not valid if this password does not match.


New Trust Wizard

Trust Password

Passwords are used by Active Directory Domain Controllers to confirm trust relationships.

Type a password for this trust. The same password must be used when creating this trust relationship in the specified domain. After the trust is created, the trust password is periodically updated for security purposes.

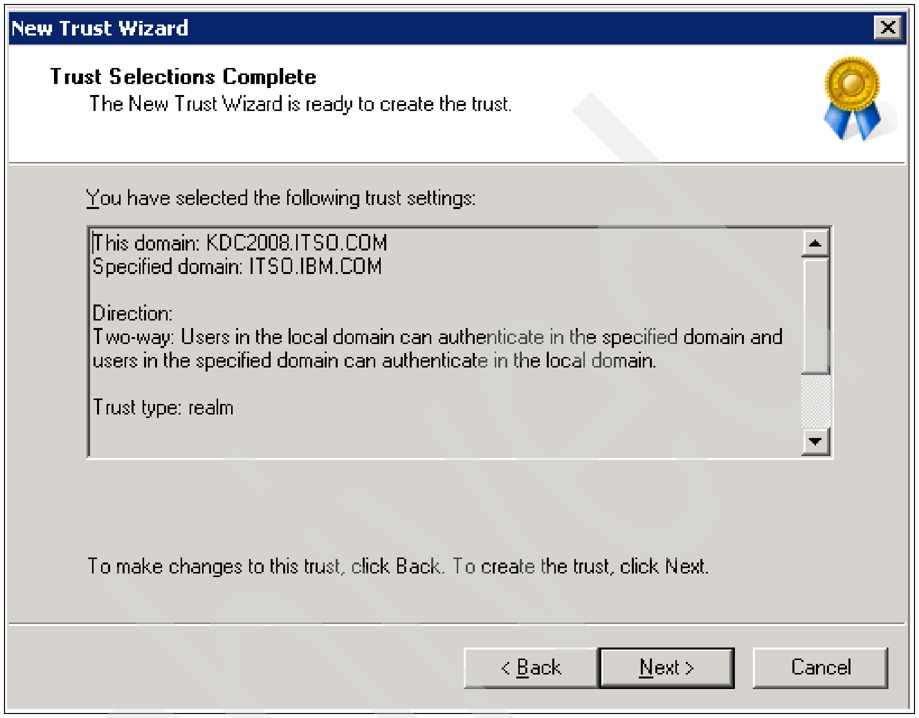
Trust password:

Confirm trust password:

Figure 6-7 New Trust Wizard, Trust Password panel

Note: Remember the trust password. You must use the same password, in uppercase, that you used when configuring trust on the z/OS KDC.

10. Review the New Trust summary, shown in Figure 6-8, and click **Next** to add the trust.


New Trust Wizard

Trust Selections Complete

The New Trust Wizard is ready to create the trust.

You have selected the following trust settings:

This domain: KDC2008.ITSO.COM
Specified domain: ITSO.IBM.COM

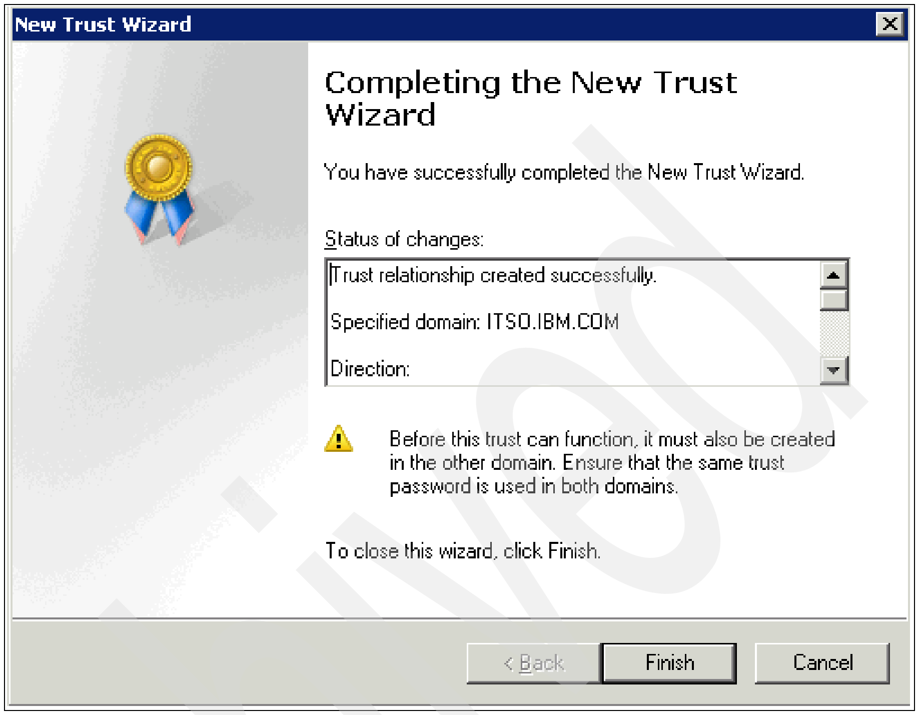
Direction:
Two-way: Users in the local domain can authenticate in the specified domain and users in the specified domain can authenticate in the local domain.

Trust type: realm

To make changes to this trust, click Back. To create the trust, click Next.

Figure 6-8 New Trust Wizard, Summary panel

11.When the new trust completes, click **Finish** (Figure 6-9).



Completing the New Trust Wizard

You have successfully completed the New Trust Wizard.

Status of changes:

Trust relationship created successfully.

Specified domain: ITSO.IBM.COM

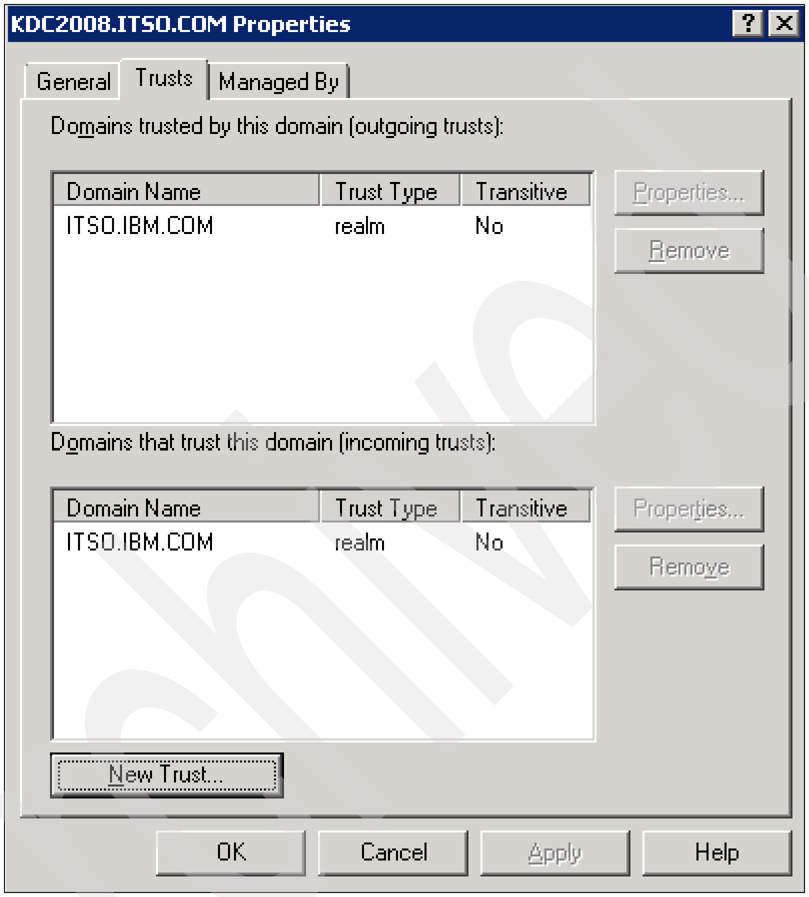
Direction:

Before this trust can function, it must also be created in the other domain. Ensure that the same trust password is used in both domains.

To close this wizard, click Finish.

Figure 6-9 New Trust Wizard, Complete panel

Back in the Trusts tab, the z/OS realm is now listed in the outgoing trust and incoming trust, as shown in Figure 6-10, because this trust was set up as a two-way trust.



Domains trusted by this domain (outgoing trusts):

Domain Name	Trust Type	Transitive
ITSO.IBM.COM	realm	No

Domains that trust this domain (incoming trusts):

Domain Name	Trust Type	Transitive
ITSO.IBM.COM	realm	No

Figure 6-10 KDC2008.ITSO.COM final Properties panel

12. Reboot the system for these changes to take effect.

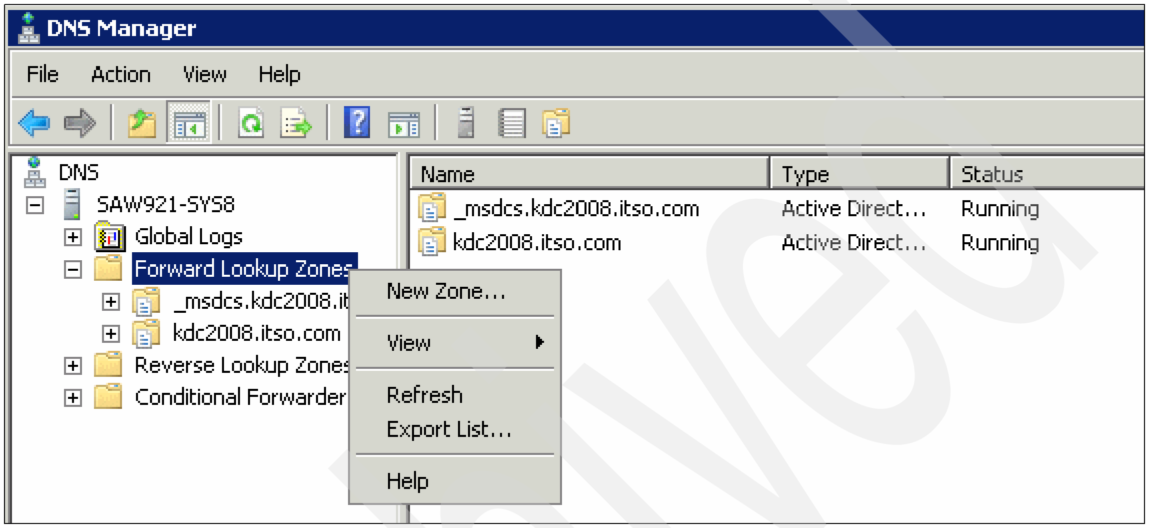
6.2.4 Configure the DNS

For DNS on the Windows system to resolve the z/OS KDC host, you need to configure the forward lookup zone and reverse lookup zone. Updates to the DNS are done through the DNS manager. To open the DNS manager, click **Start** → **Programs** → **Administrative Tools** → **DNS**.

Configure the forward lookup zone

First, configure a forward lookup zone for the z/OS KDC system by following these steps:

1. In the DNS manager window, right-click **Forward Lookup Zone** and click **New Zone**, as shown in Figure 6-11.

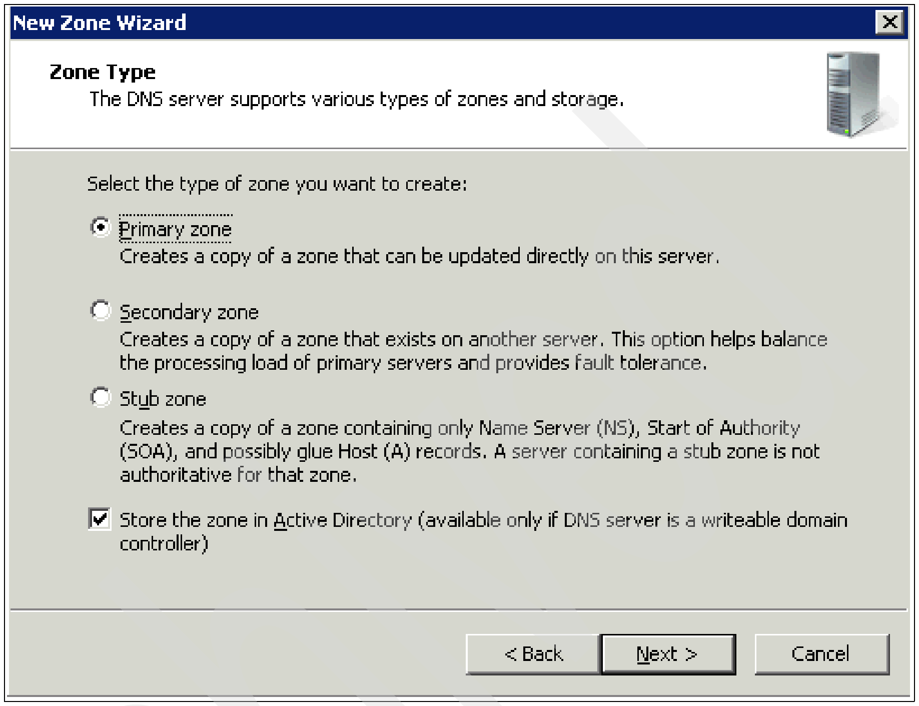


Name	Type	Status
_msdcs.kdc2008.itso.com	Active Direct...	Running
kdc2008.itso.com	Active Direct...	Running

Figure 6-11 DNS Manager window

2. Click **Next** on the first panel of the New Zone wizard.

3. Click **Primary Zone** and **Store the zone in Active Directory**, and then click **Next**, as shown in Figure 6-12.


New Zone Wizard

Zone Type
The DNS server supports various types of zones and storage.

Select the type of zone you want to create:

[✓ Primary zone]
Creates a copy of a zone that can be updated directly on this server.

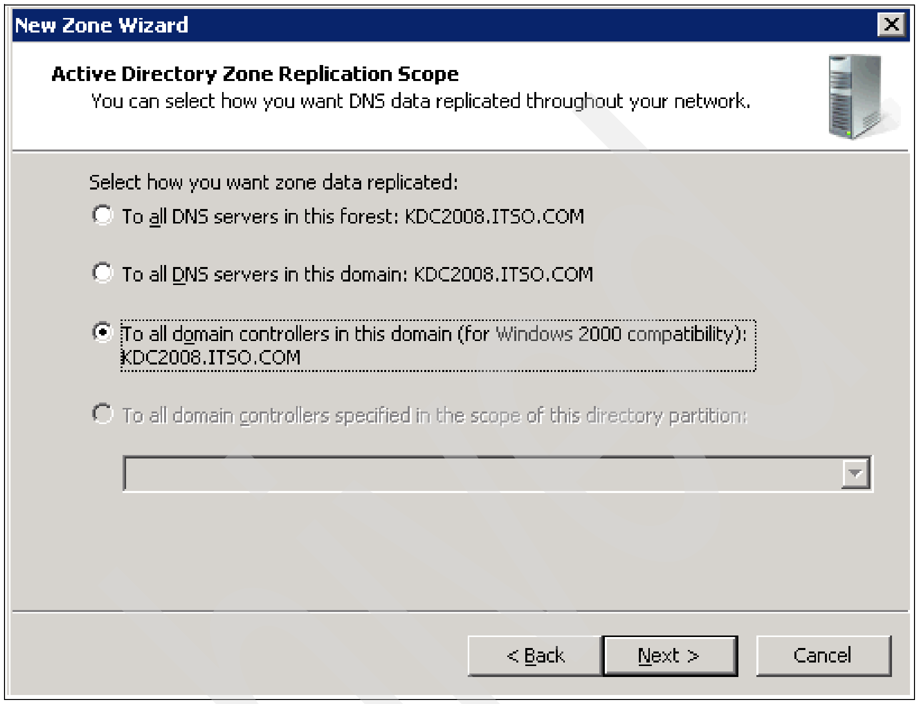
Secondary zone
Creates a copy of a zone that exists on another server. This option helps balance the processing load of primary servers and provides fault tolerance.

Stub zone
Creates a copy of a zone containing only Name Server (NS), Start of Authority (SOA), and possibly glue Host (A) records. A server containing a stub zone is not authoritative for that zone.

[✓ Store the zone in Active Directory (available only if DNS server is a writeable domain controller)]

Figure 6-12 Selecting the Zone Type

4. For the replication scope, click **To All domain controllers in this domain**, and then click **Next**, as shown in Figure 6-13.


New Zone Wizard

Active Directory Zone Replication Scope
You can select how you want DNS data replicated throughout your network.

Select how you want zone data replicated:

( ) To all DNS servers in this forest: KDC2008.ITSO.COM

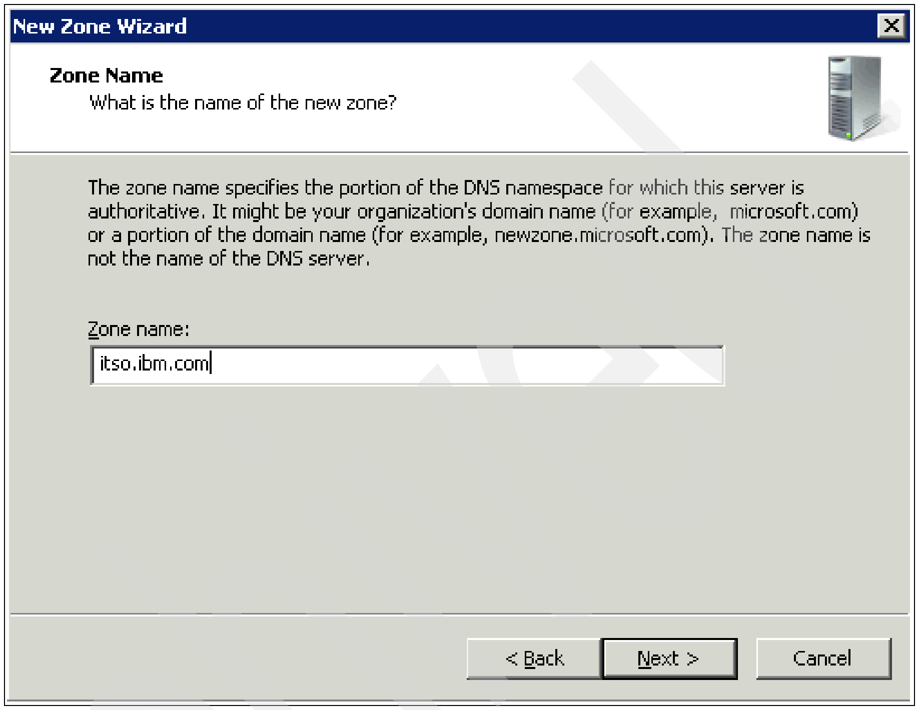
( ) To all DNS servers in this domain: KDC2008.ITSO.COM

[✓ To all domain controllers in this domain (for Windows 2000 compatibility): KDC2008.ITSO.COM]

( ) To all domain controllers specified in the scope of this directory partition:

Figure 6-13 Selecting the Replication Scope

5. For the zone name, specify the z/OS system domain name, and then click **Next**. In this example, the z/OS system host name is `wtsc60.itso.ibm.com`, so the domain name is `itso.ibm.com`, as shown in Figure 6-14.

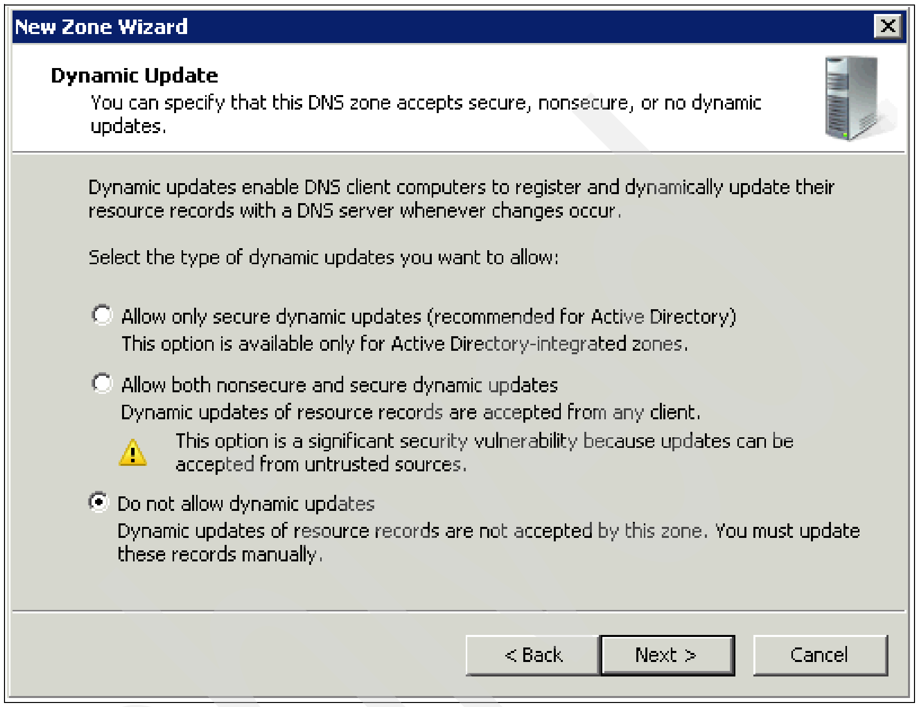


The zone name specifies the portion of the DNS namespace for which this server is authoritative. It might be your organization's domain name (for example, microsoft.com) or a portion of the domain name (for example, newzone.microsoft.com). The zone name is not the name of the DNS server.

Zone name: itso.ibm.com

Figure 6-14 Entering the zone name

6. For the dynamic update setting, click **Do not allow dynamic updates** as shown in Figure 6-15, and then click **Next**.



Dynamic Update

You can specify that this DNS zone accepts secure, nonsecure, or no dynamic updates.

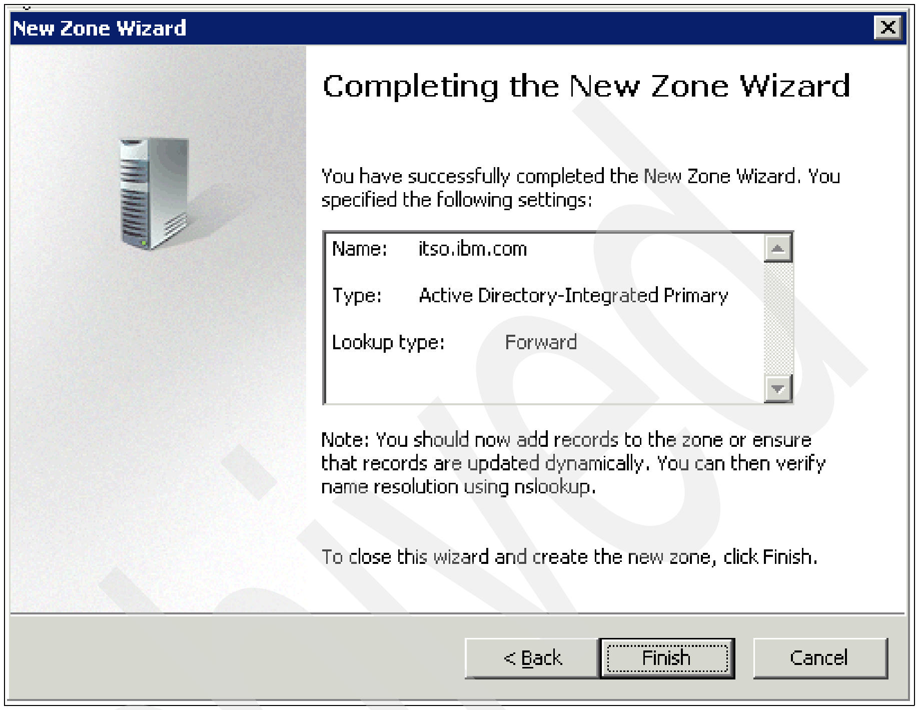
Dynamic updates enable DNS client computers to register and dynamically update their resource records with a DNS server whenever changes occur.

Select the type of dynamic updates you want to allow:

- ( ) Allow only secure dynamic updates (recommended for Active Directory)
  This option is available only for Active Directory-integrated zones.
- ( ) Allow both nonsecure and secure dynamic updates
  Dynamic updates of resource records are accepted from any client.
  ⚠ This option is a significant security vulnerability because updates can be accepted from untrusted sources.
- [✓ Do not allow dynamic updates]
  Dynamic updates of resource records are not accepted by this zone. You must update these records manually.

Figure 6-15 Selecting the Dynamic Update option

7. Confirm the new forward zone summary shown in Figure 6-16, and then click **Finish**.



Completing the New Zone Wizard

You have successfully completed the New Zone Wizard. You specified the following settings:

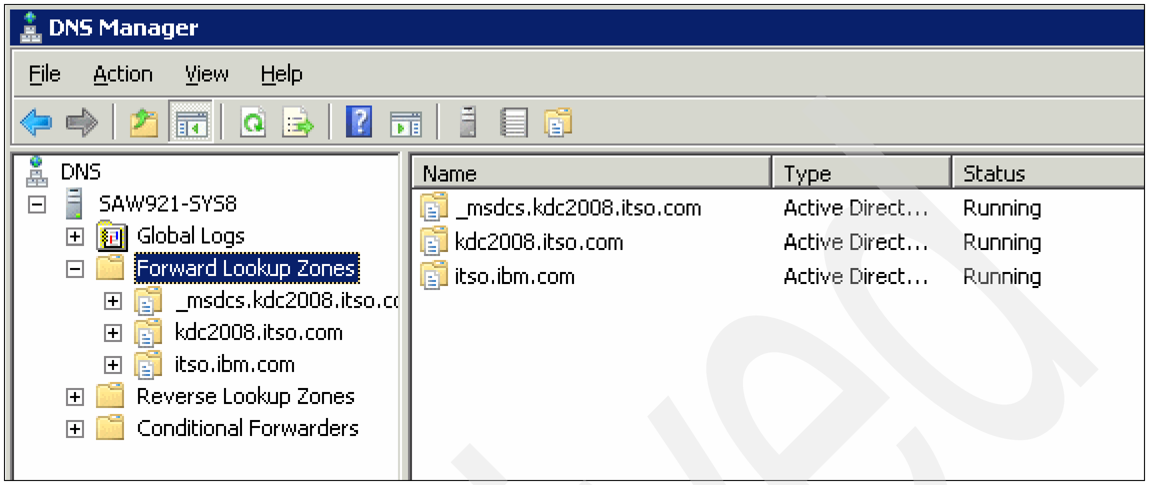
```
Name:      itso.ibm.com

Type:      Active Directory-Integrated Primary

Lookup type:        Forward
```

Note: You should now add records to the zone or ensure that records are updated dynamically. You can then verify name resolution using nslookup.

To close this wizard and create the new zone, click Finish.

Figure 6-16 Summary

8. In the DNS window, confirm that an entry for the z/OS domain was created under Forward Lookup Zones, as shown in Figure 6-17.



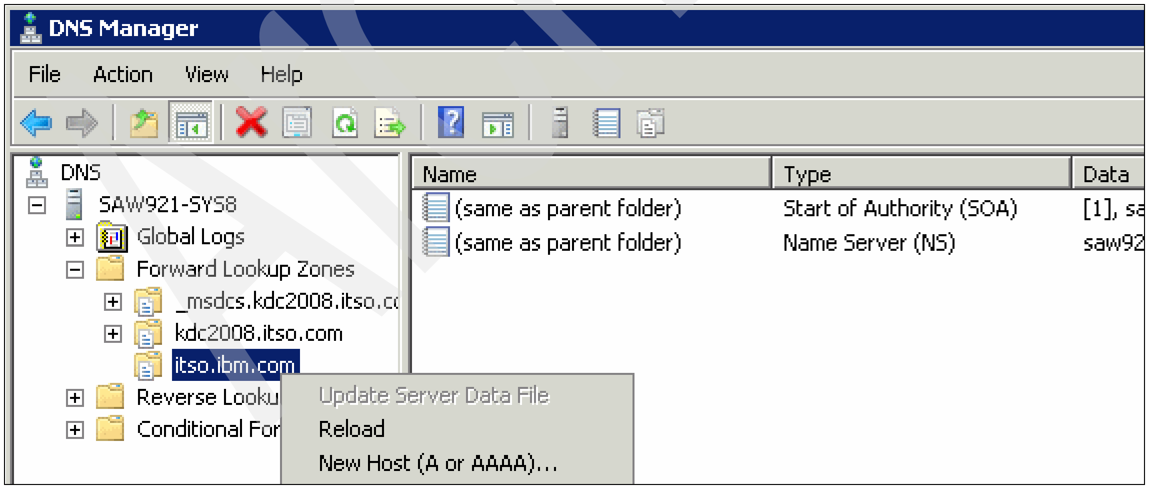
Name	Type	Status
_msdcs.kdc2008.itso.com	Active Direct...	Running
kdc2008.itso.com	Active Direct...	Running
itso.ibm.com	Active Direct...	Running

Figure 6-17 Final DNS Manager window

Create a new host entry for the zone

Next, you need to create a new host entry for the new zone by following these steps in the DNS Manager:

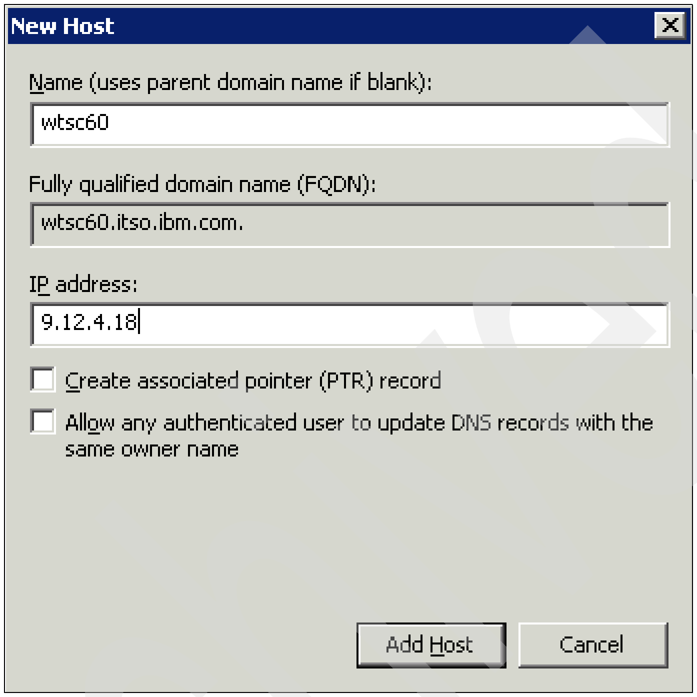
1. Right-click the z/OS domain and click **New Host (A or AAAA)**, as shown in Figure 6-18.



Name	Type	Data
(same as parent folder)	Start of Authority (SOA)	[1], sa
(same as parent folder)	Name Server (NS)	saw92

Figure 6-18 Create a new host on the z/OS domain

2. Specify the host name and IP address of the z/OS system, and click **Add Host**, as shown in Figure 6-19. The host name is the short name of the z/OS system. After entering this name, the full z/OS system name displays in the fully qualified domain name (FQDN) field.



Name (uses parent domain name if blank): wtsc60

Fully qualified domain name (FQDN): wtsc60.itso.ibm.com.

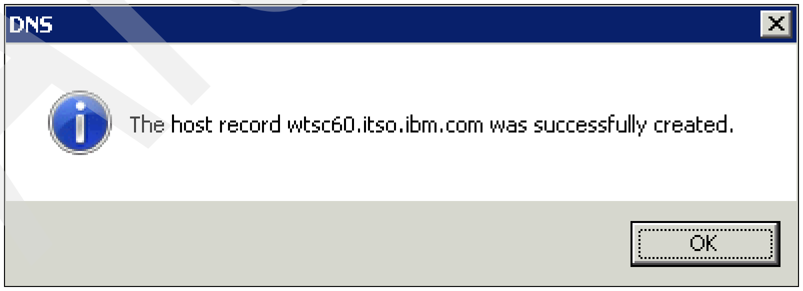
IP address: 9.12.4.18

☐ Create associated pointer (PTR) record

☐ Allow any authenticated user to update DNS records with the same owner name

Figure 6-19 Specify z/OS host name and IP address

3. After the host record is added, you receive a message similar to that shown in Figure 6-20. Click **OK**.



The host record wtsc60.itso.ibm.com was successfully created.

Figure 6-20 Host record is added

4. Click **Done** to close the New Host window (Figure 6-21).

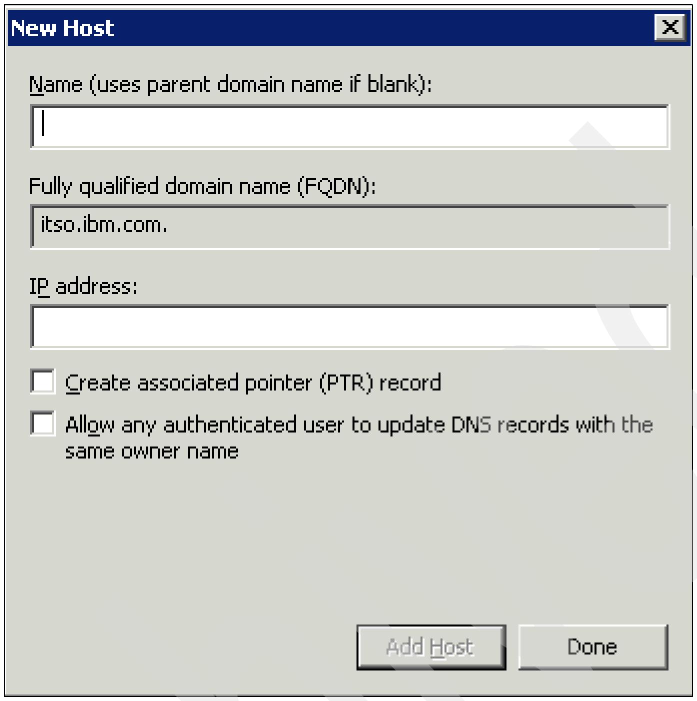
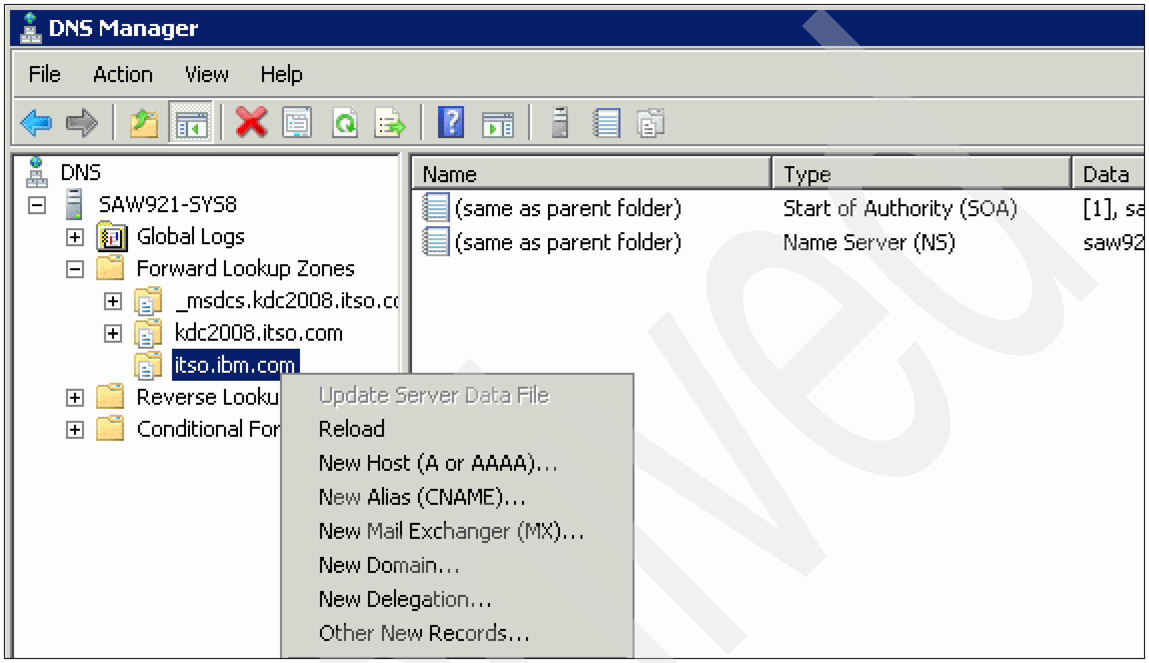

New Host
Name (uses parent domain name if blank):
Fully qualified domain name (FQDN):
itso.ibm.com.
IP address:
Create associated pointer (PTR) record
Allow any authenticated user to update DNS records with the same owner name
Add Host
Done

Figure 6-21 Done creating host

Create a text record to define the realm name

Next, you need to create a text record type to define the z/OS realm name:

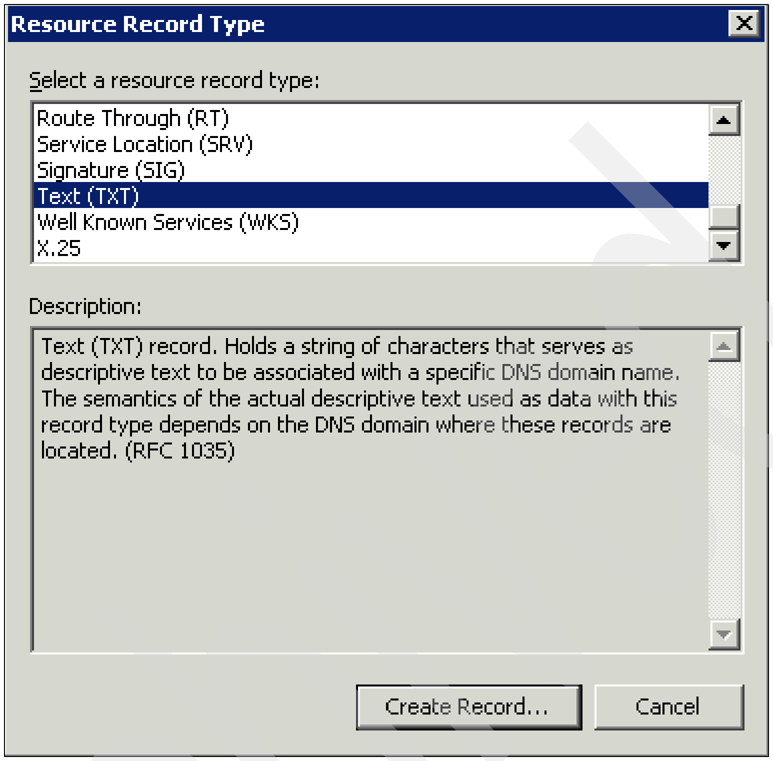
1. Right-click the z/OS domain, and click **Other New Records**, as shown in Figure 6-22.



Name	Type	Data
(same as parent folder)	Start of Authority (SOA)	[1], sa
(same as parent folder)	Name Server (NS)	saw92

Figure 6-22 Create other new records to create text record

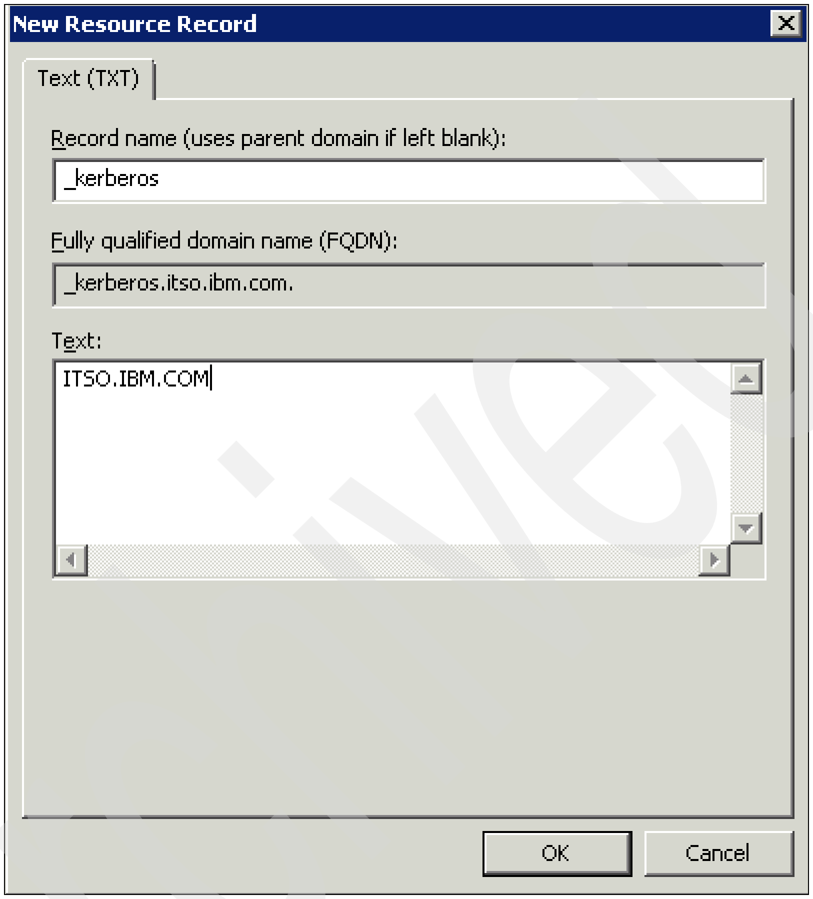
2. For the record type, click **TEXT (TXT)**, and click **Create Record** (Figure 6-23).



Text (TXT) record. Holds a string of characters that serves as descriptive text to be associated with a specific DNS domain name. The semantics of the actual descriptive text used as data with this record type depends on the DNS domain where these records are located. (RFC 1035)

Figure 6-23 Create a text record

3. Set the record name as `_kerberos`, and set the text as the z/OS realm name, in this case `ITSO.IBM.COM` as shown in Figure 6-24. Click **OK**.



Record name (uses parent domain if left blank): _kerberos

Fully qualified domain name (FQDN): _kerberos.itso.ibm.com.

Text: ITSO.IBM.COM

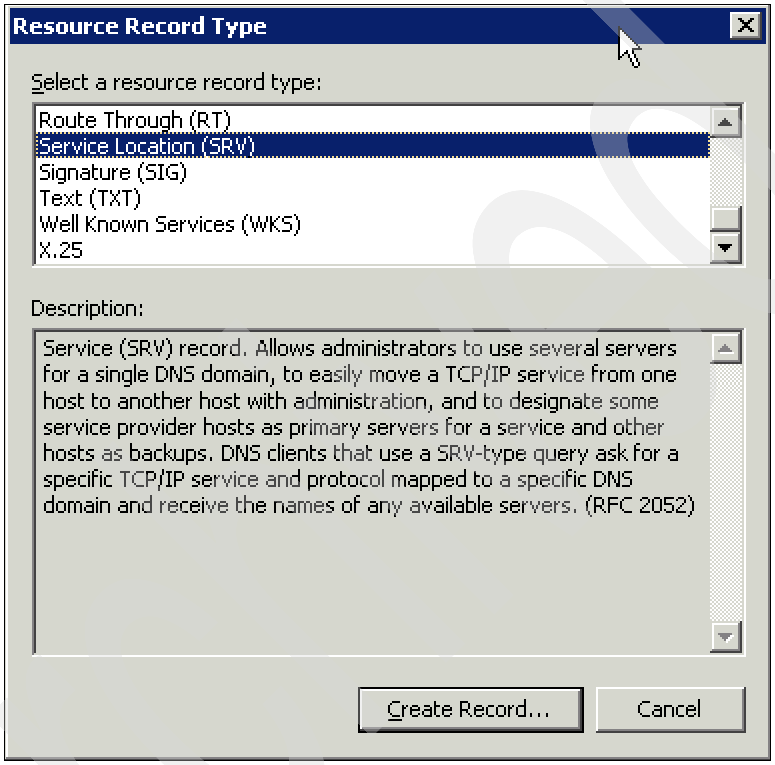
Figure 6-24 Set _kerberos record as z/OS realm name

Create the service location records

The z/OS Kerberos realm is a foreign realm to the Microsoft realm. The Microsoft DNS server can locate the z/OS Kerberos realm if you add a service location record to the Microsoft DNS server.

To create service location records for the `_kerberos` service:

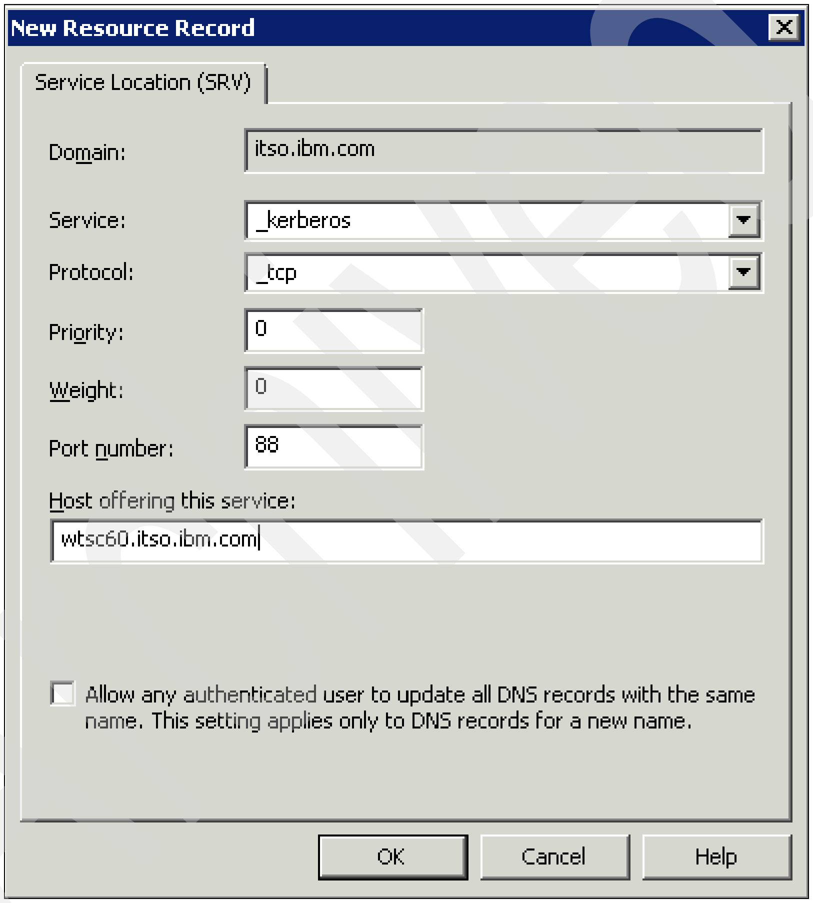
1. Right-click the z/OS domain, and click **Other New Records**, as shown in Figure 6-22 on page 123.
2. For the record type, click **Service Location (SRV),** and click **Create Record** (Figure 6-25).



Service (SRV) record. Allows administrators to use several servers for a single DNS domain, to easily move a TCP/IP service from one host to another host with administration, and to designate some service provider hosts as primary servers for a service and other hosts as backups. DNS clients that use a SRV-type query ask for a specific TCP/IP service and protocol mapped to a specific DNS domain and receive the names of any available servers. (RFC 2052)

Figure 6-25   Create service location record

3. In the New Resource Record dialog box, shown in Figure 6-26, complete the following fields:
   a. Set the service name to `_kerberos`.
   b. Set the protocol to `_tcp`.
   c. Set the "Host offering this service" field to the fully qualified host name of the z/OS system.
   d. Click **OK**.

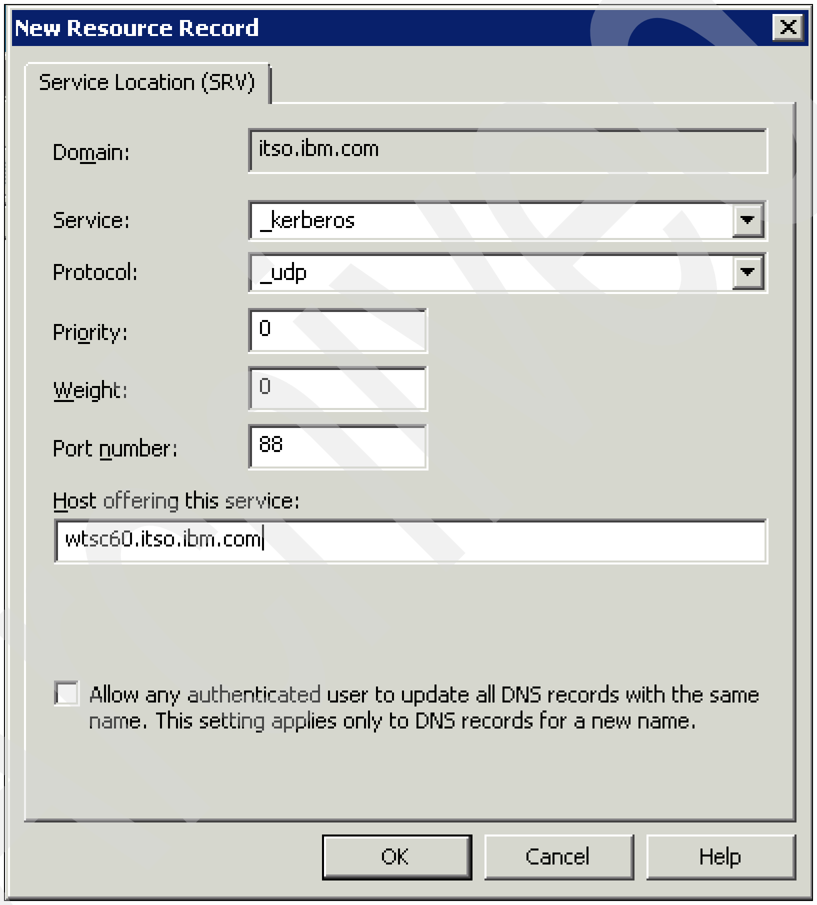


Domain: itso.ibm.com
Service: _kerberos
Protocol: _tcp
Priority: 0
Weight: 0
Port number: 88
Host offering this service: wtsc60.itso.ibm.com

☐ Allow any authenticated user to update all DNS records with the same name. This setting applies only to DNS records for a new name.

Figure 6-26 Set _kerberos service and _tcp protocol

4. Create a second service location record by completing the following fields as shown in Figure 6-27:
   a. Set the service name to `_kerberos`.
   b. Set the protocol to `_udp`.
   c. Set the “Host offering this service” field to the fully qualified host name of the z/OS system.
   d. Click **OK**.


New Resource Record

Service Location (SRV)

Domain: itso.ibm.com

Service: _kerberos

Protocol: _udp

Priority: 0

Weight: 0

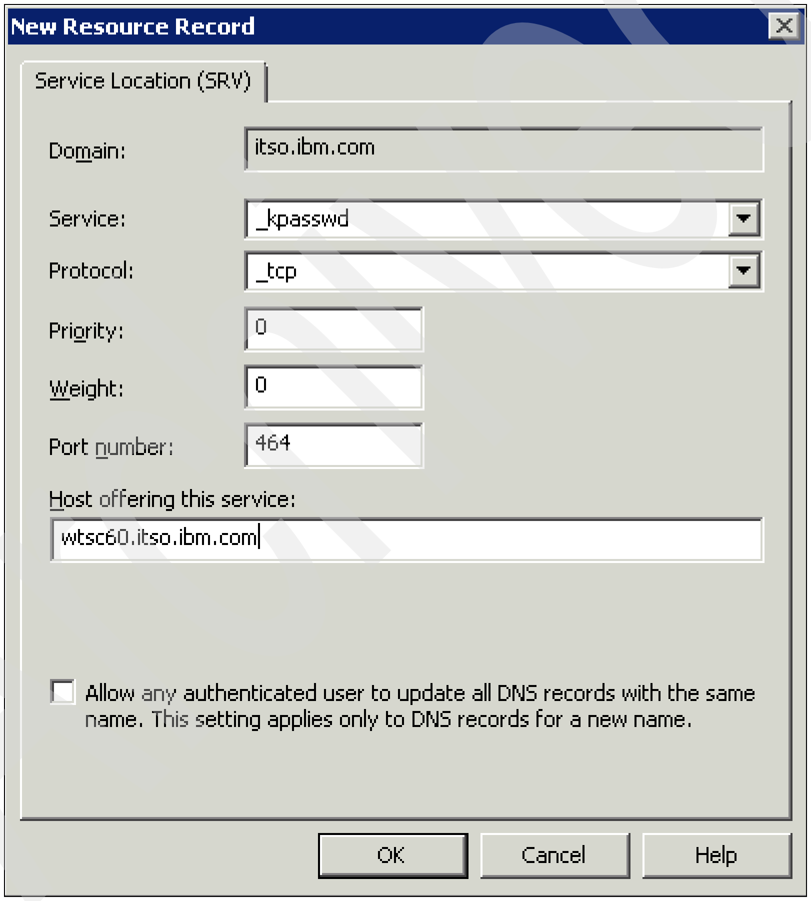
Port number: 88

Host offering this service: wtsc60.itso.ibm.com

☐ Allow any authenticated user to update all DNS records with the same name. This setting applies only to DNS records for a new name.

Figure 6-27 Set _kerberos service and _udp protocol

5. Create a third service location record by completing the following fields, as shown in Figure 6-28:
   a. Set the service name to `_kpasswd`.
   b. Set the protocol to `_tcp`.
   c. Enter the z/OS password server port in the "Port number" field. The default is `464`.
   d. Set the "Host offering this service" field to the fully qualified host name of the z/OS system.
   e. Click **OK**.


New Resource Record

Service Location (SRV)

Domain: itso.ibm.com

Service: _kpasswd

Protocol: _tcp

Priority: 0

Weight: 0

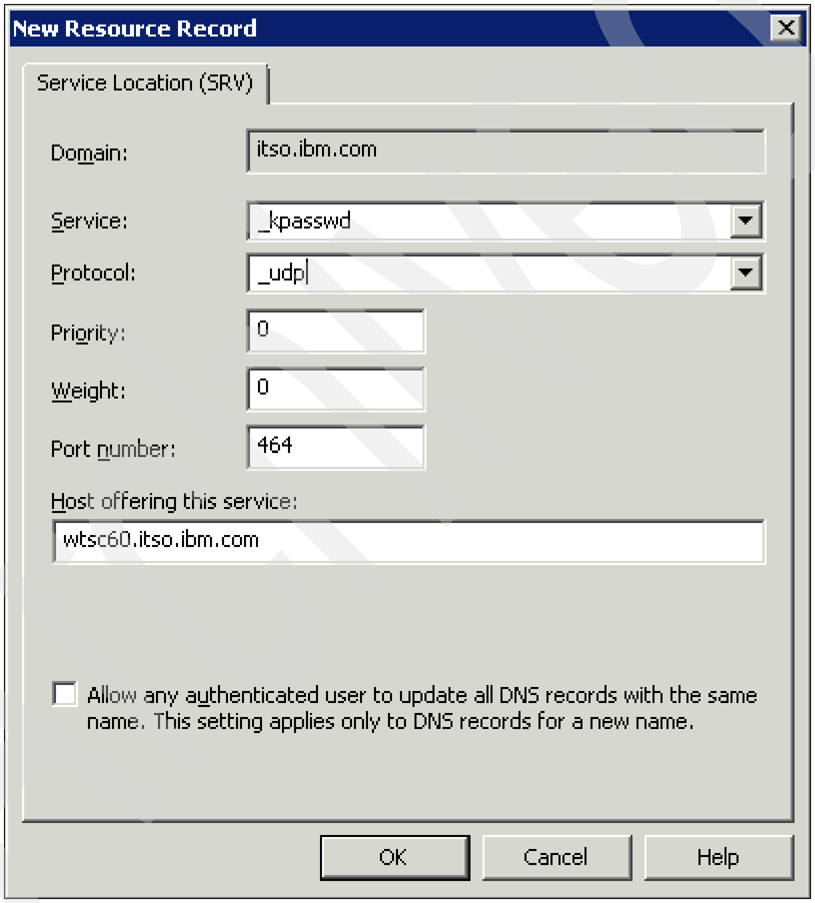
Port number: 464

Host offering this service:
wtsc60.itso.ibm.com

☐ Allow any authenticated user to update all DNS records with the same name. This setting applies only to DNS records for a new name.

Figure 6-28 Set _kpasswd service and _tcp protocol

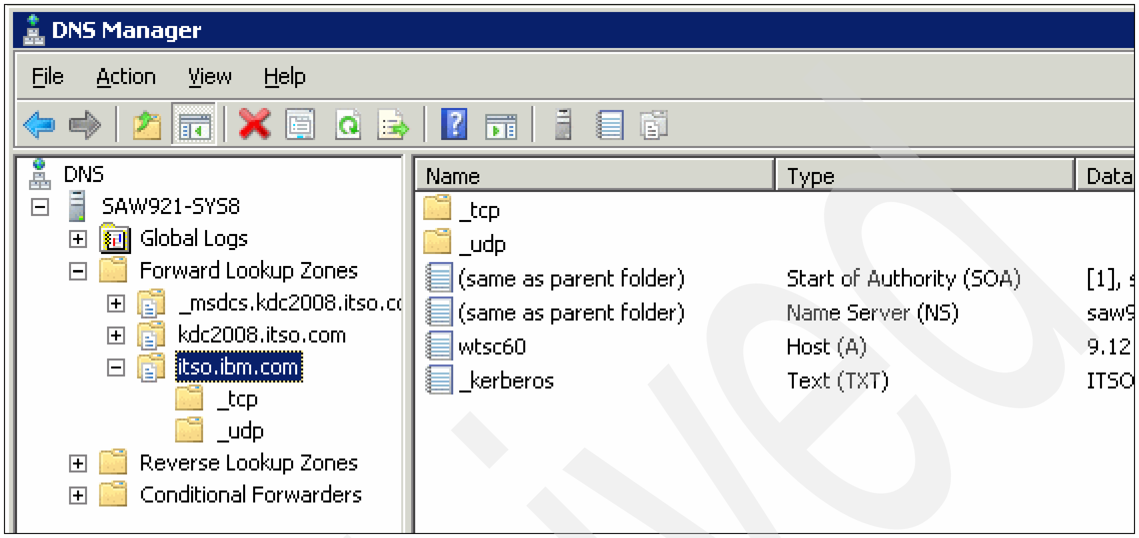
6. Create a fourth service location record by completing the following fields, as shown in Figure 6-29:
   a. Set the service name to `_kpasswd`.
   b. Set the protocol to `_udp`.
   c. Set the port number to `464`.
   d. Set the "Host offering this service" field to the fully qualified host name of the z/OS system.
   e. Click **OK**.



Domain: itso.ibm.com
Service: _kpasswd
Protocol: _udp
Priority: 0
Weight: 0
Port number: 464
Host offering this service: wtsc60.itso.ibm.com
☐ Allow any authenticated user to update all DNS records with the same name. This setting applies only to DNS records for a new name.

Figure 6-29 Set _kpasswd service and _udp protocol

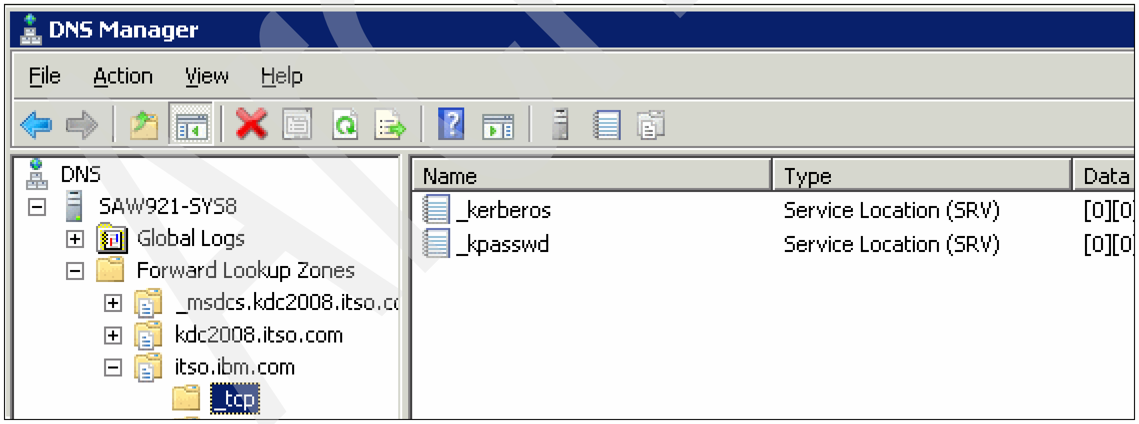
7. In the DNS Manager window, click the z/OS domain. A text record for `_kerberos` and a host record for the z/OS system short name display, as shown in Figure 6-30.



Name	Type	Data
_tcp		
_udp		
(same as parent folder)	Start of Authority (SOA)	[1], s
(same as parent folder)	Name Server (NS)	saw9
wtsc60	Host (A)	9.12
_kerberos	Text (TXT)	ITSO

Figure 6-30 Entry for _kerberos and z/OS system short name added

8. In the DNS window, click **_tcp** and **_udp** under the z/OS domain. A service location record for `_kerberos` and `_kpasswd` with the z/OS fully qualified host name display, as shown in Figure 6-31.



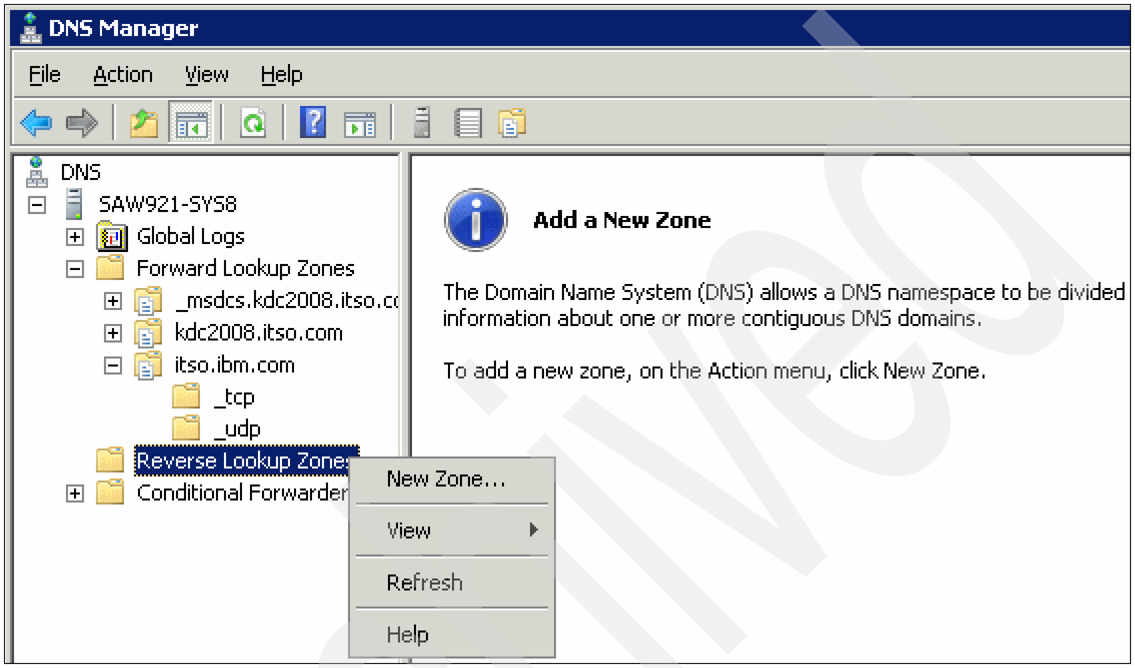
Name	Type	Data
_kerberos	Service Location (SRV)	[0][0
_kpasswd	Service Location (SRV)	[0][0

Figure 6-31 Service location records for _kerberos and _kpasswd for both _tcp and _udp

Create a reverse lookup zone

Now, you need to create a reverse lookup zone for the z/OS system:

1. Right-click the **Reverse Lookup Zone** folder, and click **New Zone**, as shown in Figure 6-32.



Add a New Zone

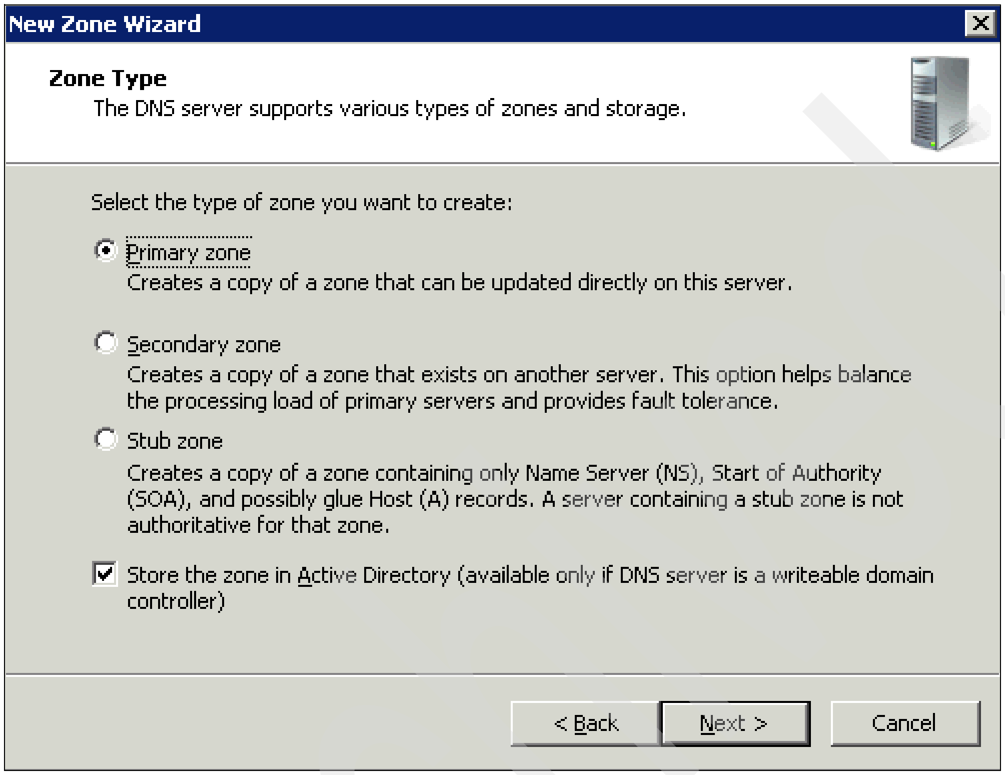
The Domain Name System (DNS) allows a DNS namespace to be divided information about one or more contiguous DNS domains.

To add a new zone, on the Action menu, click New Zone.

Figure 6-32 Create new reverse lookup zone

2. Click **Next** on the first pane of the New Zone wizard.

3. In the Zone Type window, click **Primary Zone**, and click **Store the zone in Active Directory**, as shown in Figure 6-33. Then, click **Next**.


New Zone Wizard

Zone Type

The DNS server supports various types of zones and storage.

Select the type of zone you want to create:

(●) Primary zone
Creates a copy of a zone that can be updated directly on this server.

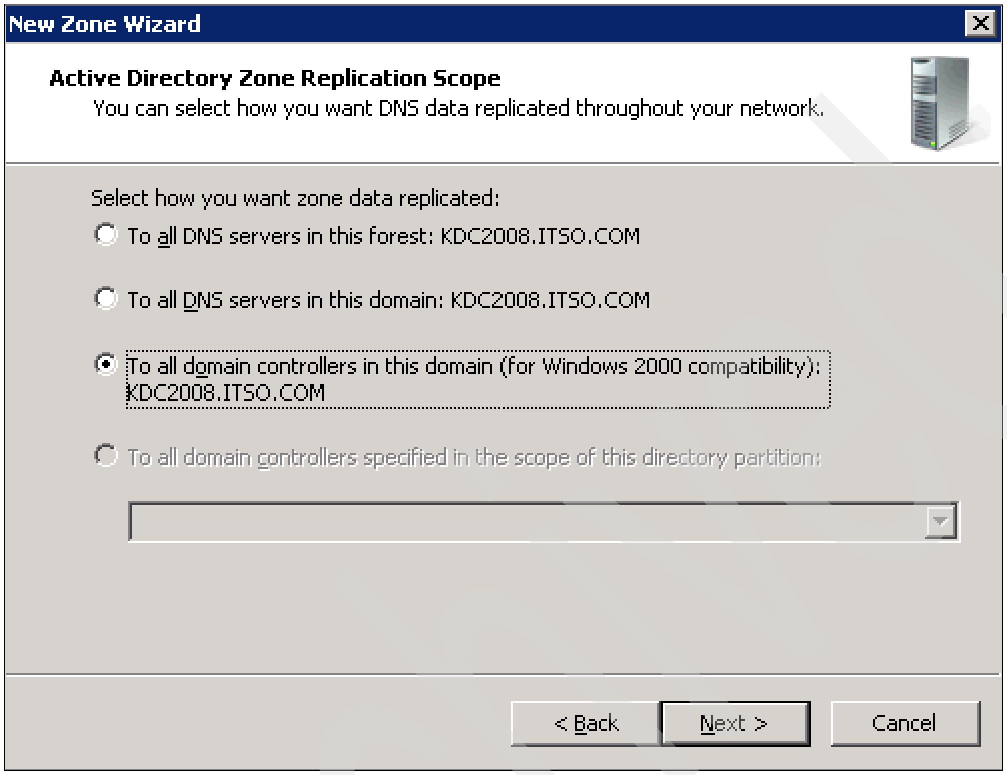
( ) Secondary zone
Creates a copy of a zone that exists on another server. This option helps balance the processing load of primary servers and provides fault tolerance.

( ) Stub zone
Creates a copy of a zone containing only Name Server (NS), Start of Authority (SOA), and possibly glue Host (A) records. A server containing a stub zone is not authoritative for that zone.

[✓] Store the zone in Active Directory (available only if DNS server is a writeable domain controller)

Figure 6-33 Select Primary zone type

4. For the replication scope, click **To all domain controllers in this domain**, as shown in Figure 6-34, and then click **Next**.


New Zone Wizard

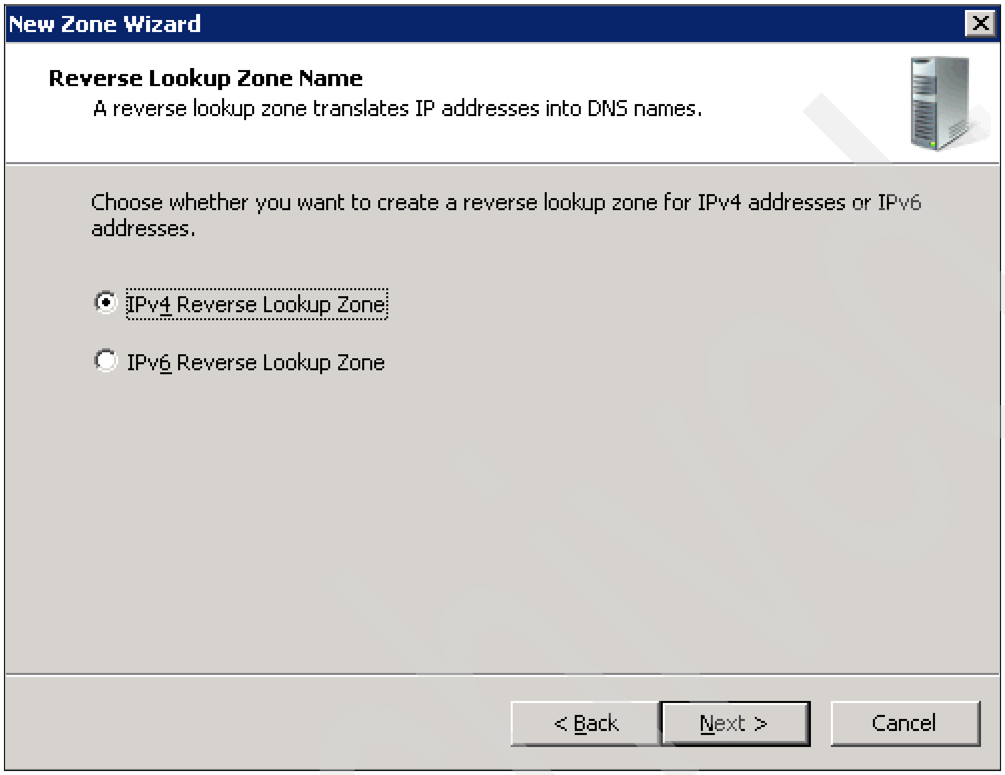
Active Directory Zone Replication Scope
You can select how you want DNS data replicated throughout your network.

Select how you want zone data replicated:

- ○ To all DNS servers in this forest: KDC2008.ITSO.COM
- ○ To all DNS servers in this domain: KDC2008.ITSO.COM
- ● To all domain controllers in this domain (for Windows 2000 compatibility): KDC2008.ITSO.COM
- ○ To all domain controllers specified in the scope of this directory partition:

Figure 6-34 Select replication scope

5. If prompted, select the appropriate IP zone, as shown in Figure 6-35, and then click **Next**.


New Zone Wizard

Reverse Lookup Zone Name

A reverse lookup zone translates IP addresses into DNS names.

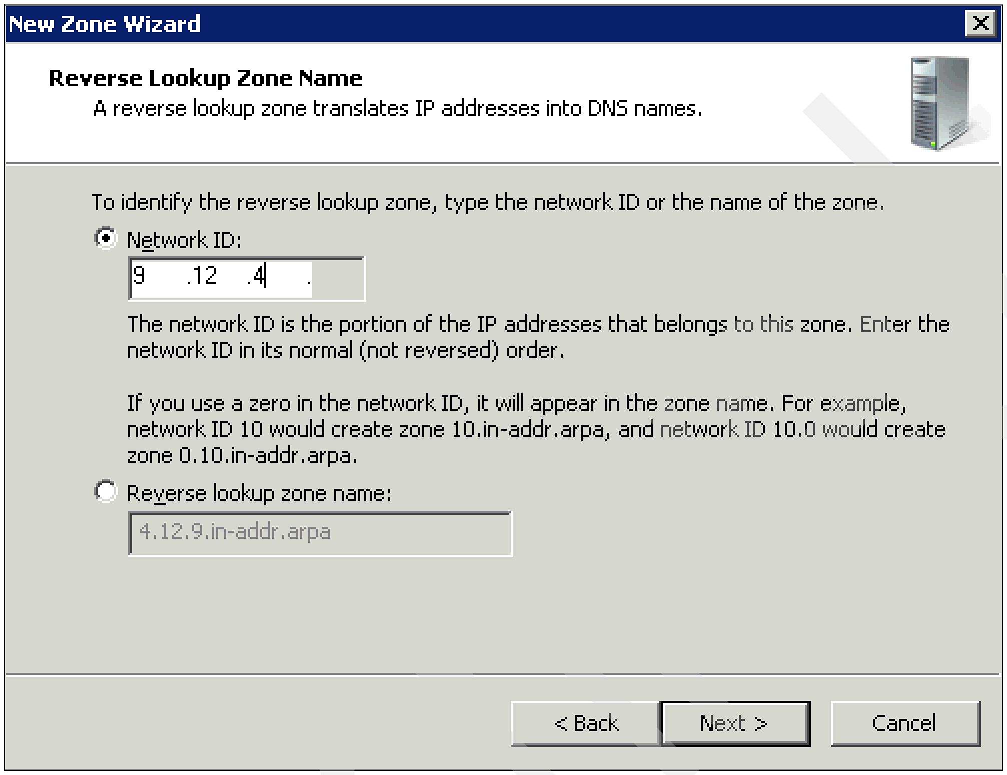
Choose whether you want to create a reverse lookup zone for IPv4 addresses or IPv6 addresses.

[✓ IPv4 Reverse Lookup Zone]

IPv6 Reverse Lookup Zone

Figure 6-35 Select IP zone

6. Set the network ID (as shown in Figure 6-36), and then click **Next**. The network ID is the first three octets of the z/OS system's IP address.


New Zone Wizard

Reverse Lookup Zone Name

A reverse lookup zone translates IP addresses into DNS names.

To identify the reverse lookup zone, type the network ID or the name of the zone.

[● Network ID:]

9 .12 .4 .

The network ID is the portion of the IP addresses that belongs to this zone. Enter the network ID in its normal (not reversed) order.

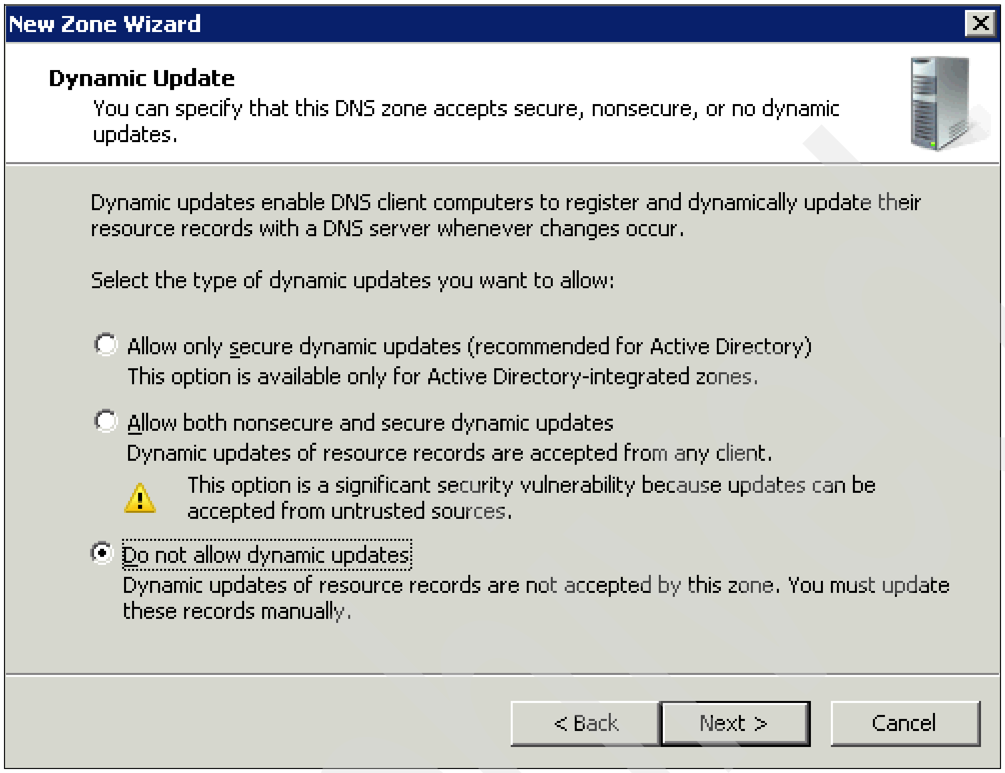
If you use a zero in the network ID, it will appear in the zone name. For example, network ID 10 would create zone 10.in-addr.arpa, and network ID 10.0 would create zone 0.10.in-addr.arpa.

○ Reverse lookup zone name:

4.12.9.in-addr.arpa

Figure 6-36 Set network ID

7. Click **Do not allow dynamic updates**, and then click **Next** (Figure 6-37).


New Zone Wizard

Dynamic Update

You can specify that this DNS zone accepts secure, nonsecure, or no dynamic updates.

Dynamic updates enable DNS client computers to register and dynamically update their resource records with a DNS server whenever changes occur.

Select the type of dynamic updates you want to allow:

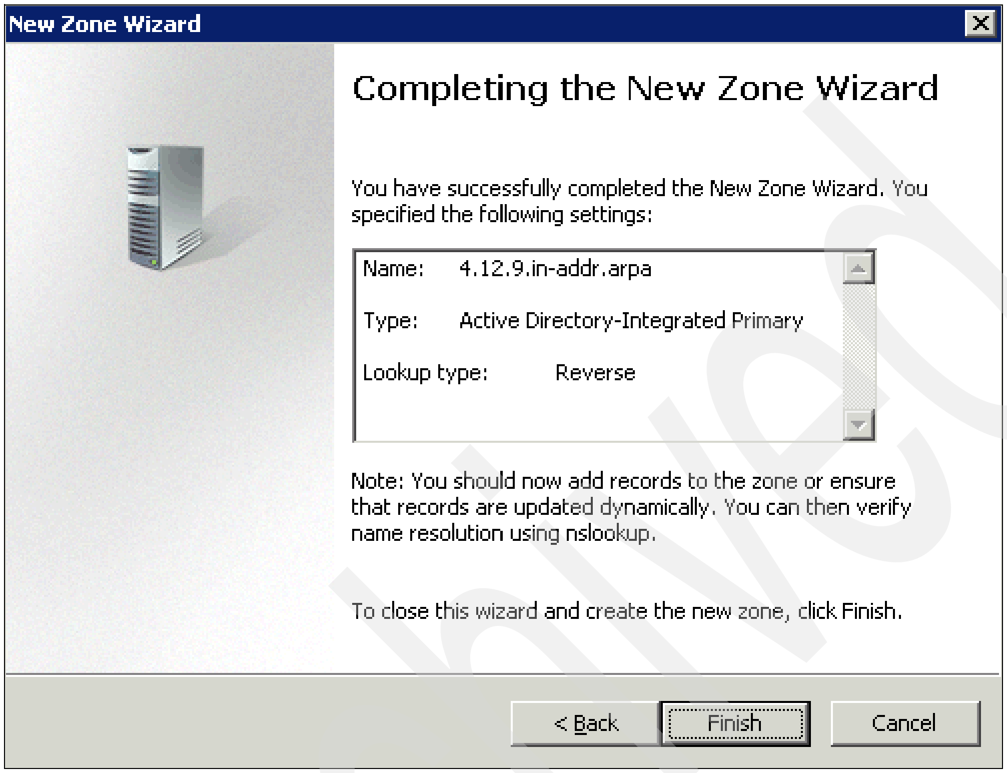
( ) Allow only secure dynamic updates (recommended for Active Directory)
This option is available only for Active Directory-integrated zones.

( ) Allow both nonsecure and secure dynamic updates
Dynamic updates of resource records are accepted from any client.
This option is a significant security vulnerability because updates can be accepted from untrusted sources.

[✓ Do not allow dynamic updates]
Dynamic updates of resource records are not accepted by this zone. You must update these records manually.

Figure 6-37 Do not allow dynamic updates

8. Verify the new reverse lookup zone summary, as shown in Figure 6-38, and then click **Finish**.



Completing the New Zone Wizard

You have successfully completed the New Zone Wizard. You specified the following settings:

```
Name:     4.12.9.in-addr.arpa

Type:     Active Directory-Integrated Primary

Lookup type:        Reverse
```

Note: You should now add records to the zone or ensure that records are updated dynamically. You can then verify name resolution using nslookup.

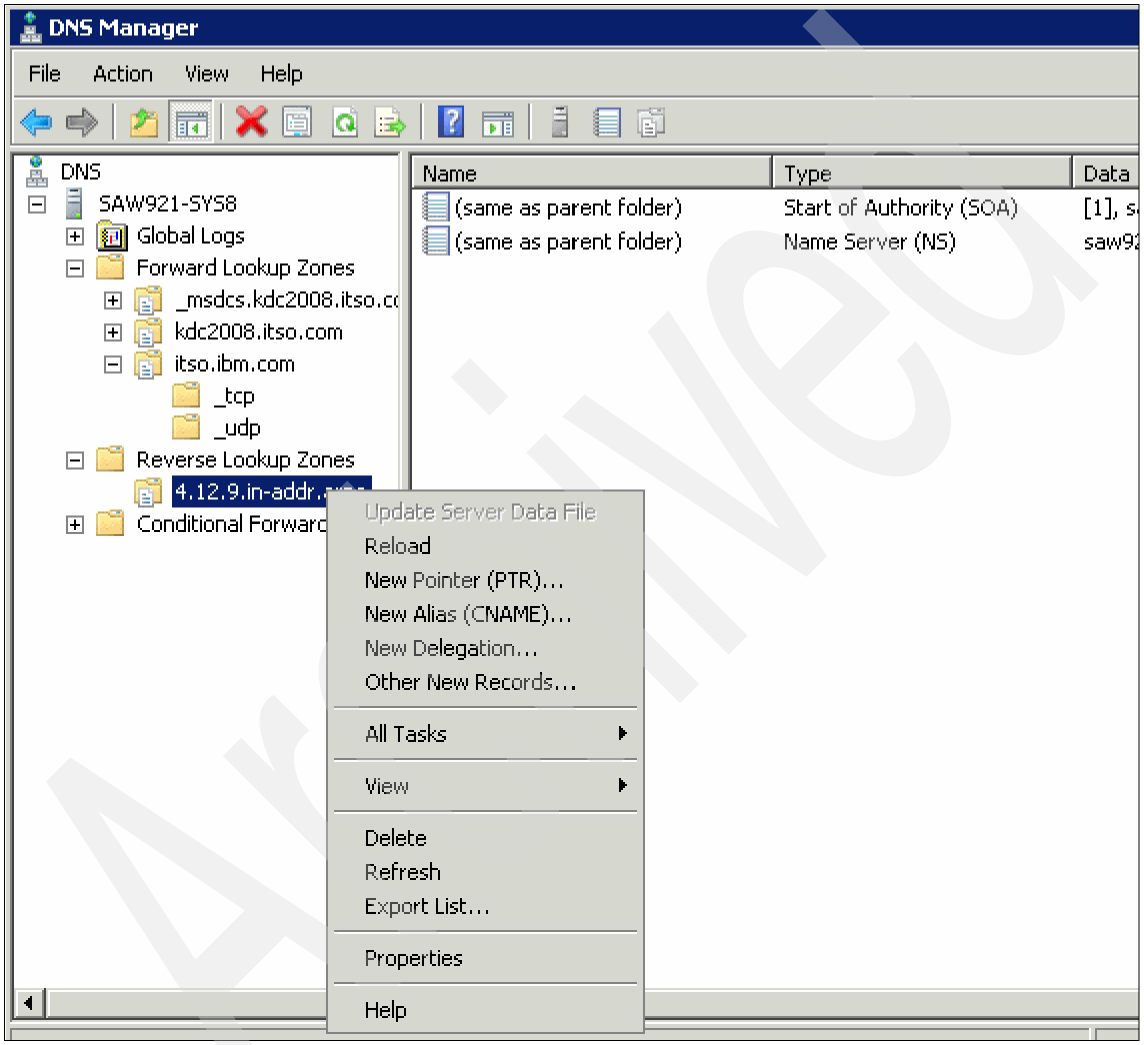
To close this wizard and create the new zone, click Finish.

Figure 6-38 Verify new reverse lookup zone summary

Create a pointer record

Next, you need to create a new pointer record for the new reverse lookup zone:

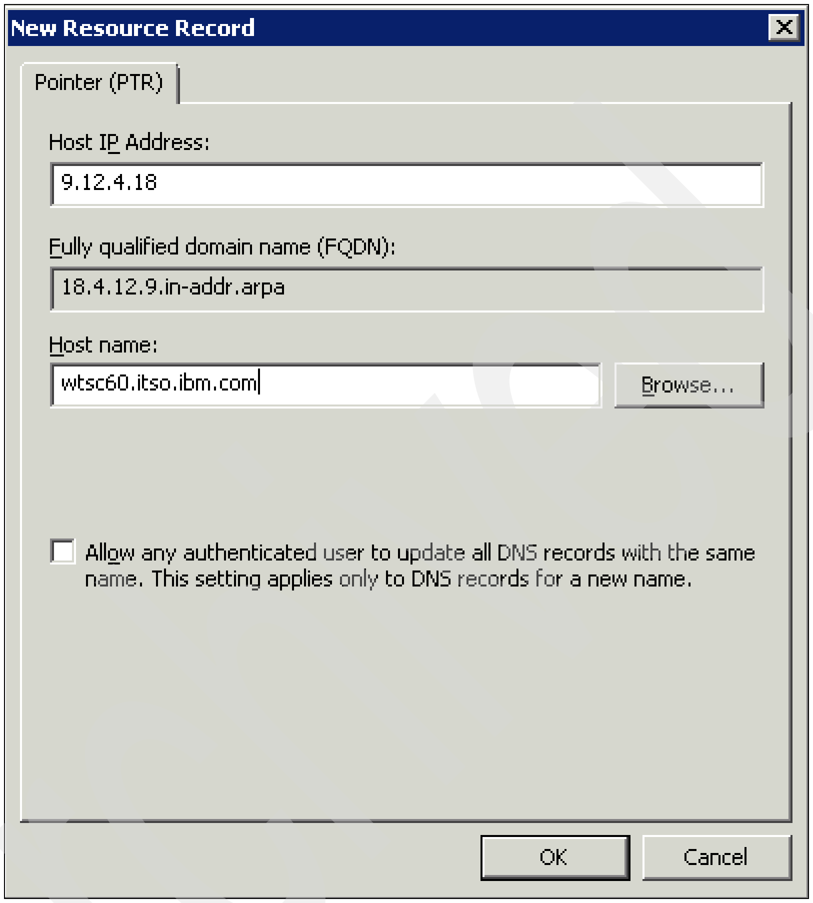
1. Right-click the reverse lookup zone, and click **New Pointer (PTR)**, as shown in Figure 6-39.



Name	Type	Data
(same as parent folder)	Start of Authority (SOA)	[1], s
(same as parent folder)	Name Server (NS)	saw92

Figure 6-39 Select New Pointer

2. For the pointer record, set the z/OS system IP address and fully qualified host name, as shown in Figure 6-40, and click **OK**.



Host IP Address: 9.12.4.18

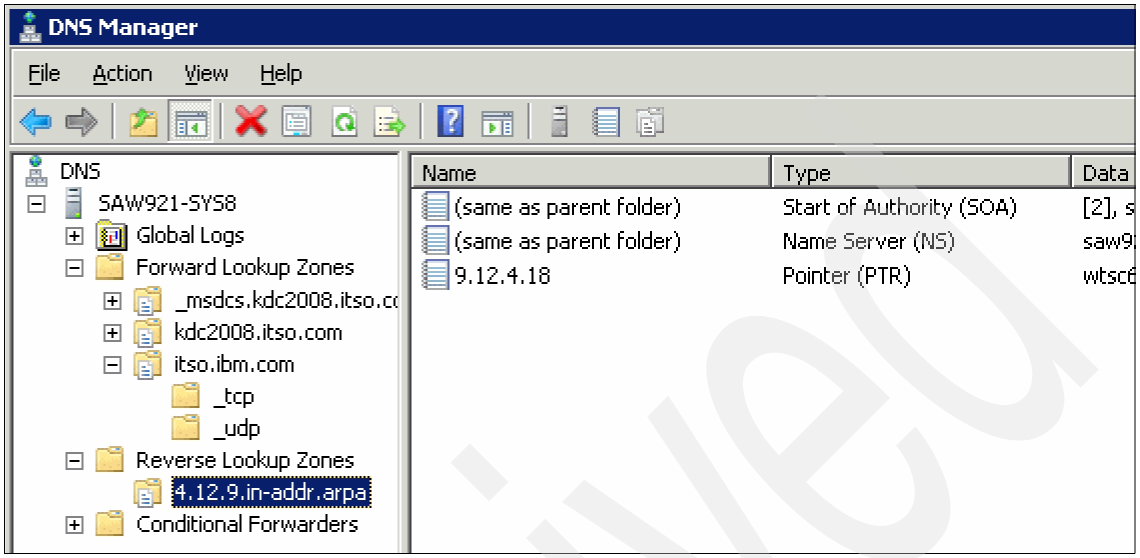
Fully qualified domain name (FQDN): 18.4.12.9.in-addr.arpa

Host name: wtsc60.itso.ibm.com

☐ Allow any authenticated user to update all DNS records with the same name. This setting applies only to DNS records for a new name.

Figure 6-40 Set z/OS system IP address and fully qualified host name

3. In the DNS window, click the z/OS reverse lookup zone. A pointer record for the z/OS system displays (Figure 6-41).



Name	Type	Data
(same as parent folder)	Start of Authority (SOA)	[2], s
(same as parent folder)	Name Server (NS)	saw9
9.12.4.18	Pointer (PTR)	wtsc6

Figure 6-41 Pointer record for z/OS reverse lookup zone

4. Reboot the system for these changes to take effect.

6.3 Configuring the Microsoft Windows client

To configure the Windows client to resolve the z/OS KDC realm, complete the following steps:

1. Install the prerequisites.
2. Configure the location of the z/OS KDC.
3. Modify the Kerberos registry settings.

Note: Some of these steps require that you modify the Microsoft Windows registry. As always, use caution when modifying the registry.

We explain these steps in the sections that follow.

6.3.1 Install the prerequisites

As with the Windows Active Directory prerequisites listed in 6.2.1, “Install the prerequisites” on page 103, you need the following prerequisites for the Windows 2003 client:

- Windows Server 2003 Resource Kit Tools
- Windows Server 2003 Support Tools

6.3.2 Configure the location of the z/OS KDC

First, you need to configure Windows so that it can connect to the z/OS KDC server. Run the following command from a Windows command prompt:

```
ksetup /AddKdc <RealmName> [KdcName]
```

Specify the z/OS KDC realm for *RealmName* and host name for *KdcName*. If you run **ksetup /AddKdc** without the *KdcName*, DNS is used to locate the KDC.

Example 6-2 shows an example of these commands.

Example 6-2 The ksetup command with /AddKdc

```
ksetup /AddKdc ITSO.IBM.COM wtsc60.itso.ibm.com
```

You can also run `ksetup.exe` to view the Kerberos realm information, as shown in Example 6-3.

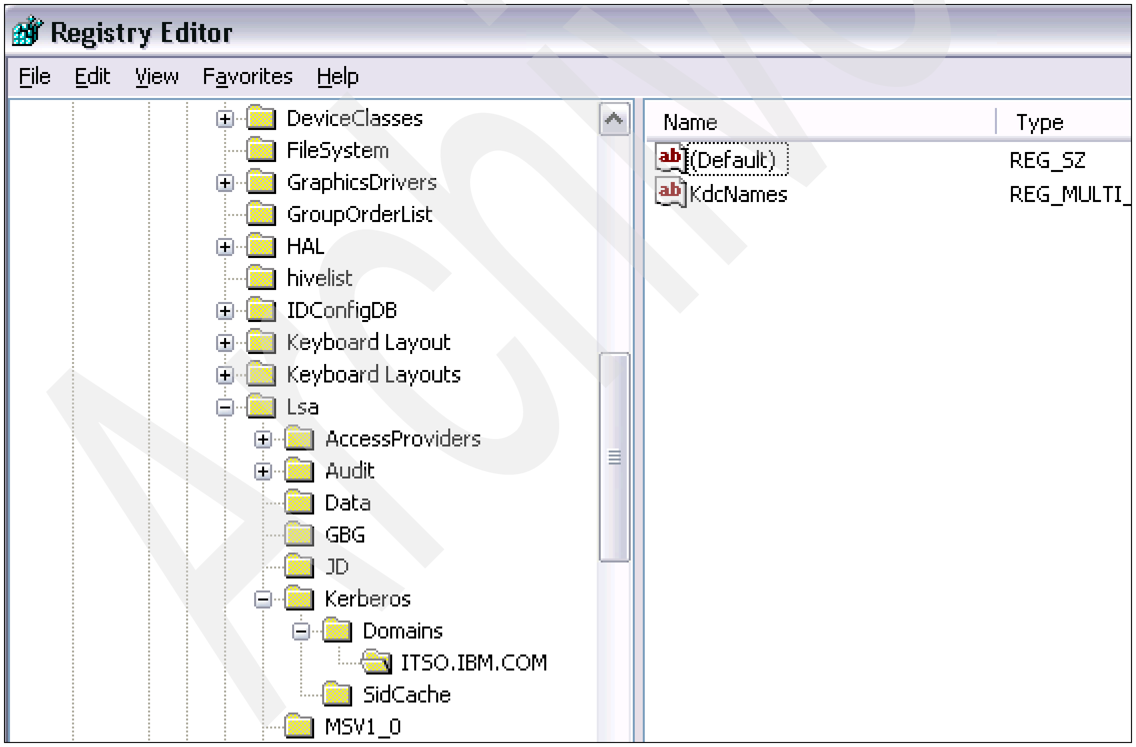
Example 6-3 The ksetup command with no arguments

```
default realm = KDC2008.ITSO.COM (NT Domain)
ITSO.IBM.COM:
        kdc = wtsc60.itso.ibm.com
        Realm Flags = 0x0 none
No user mappings defined.
```

6.3.3 Modify the Kerberos registry settings

Windows does not provide a wizard or a tool to help configure the client for a foreign Kerberos realm. To configure the client to properly route Kerberos principal requests to the correct realm, follow these steps:

1. Open **`regedit`** by clicking **Start** → **Run** and typing `regedit.exe`. Then, click **OK**.
2. Click **HKEY_LOCAL_MACHINE** → **SYSTEM** → **CurrentControlSet** → **Control** → **Lsa** → **Kerberos** → **Domains**.
3. The Domains subkey contains information about non-Windows Kerberos realms. Verify that the **`ksetup`** tool added an entry in the Domains subkey for the z/OS KDC realm.
4. Click the z/OS KDC realm under the Domains subkey. There should be a key entry for KdcNames with the z/OS system host name as the value, as shown in Figure 6-42.



Name	Type
(Default)	REG_SZ
KdcNames	REG_MULTI_

Figure 6-42 Domains subkey

5. Next, create a subkey with the host-to-realm mapping information by clicking **HKEY_LOCAL_MACHINE** → **SYSTEM** → **CurrentControlSet** → **Control** → **Lsa** → **Kerberos**.
6. Then, right-click **Kerberos**, and click **New** → **Key**, as shown in Figure 6-43.

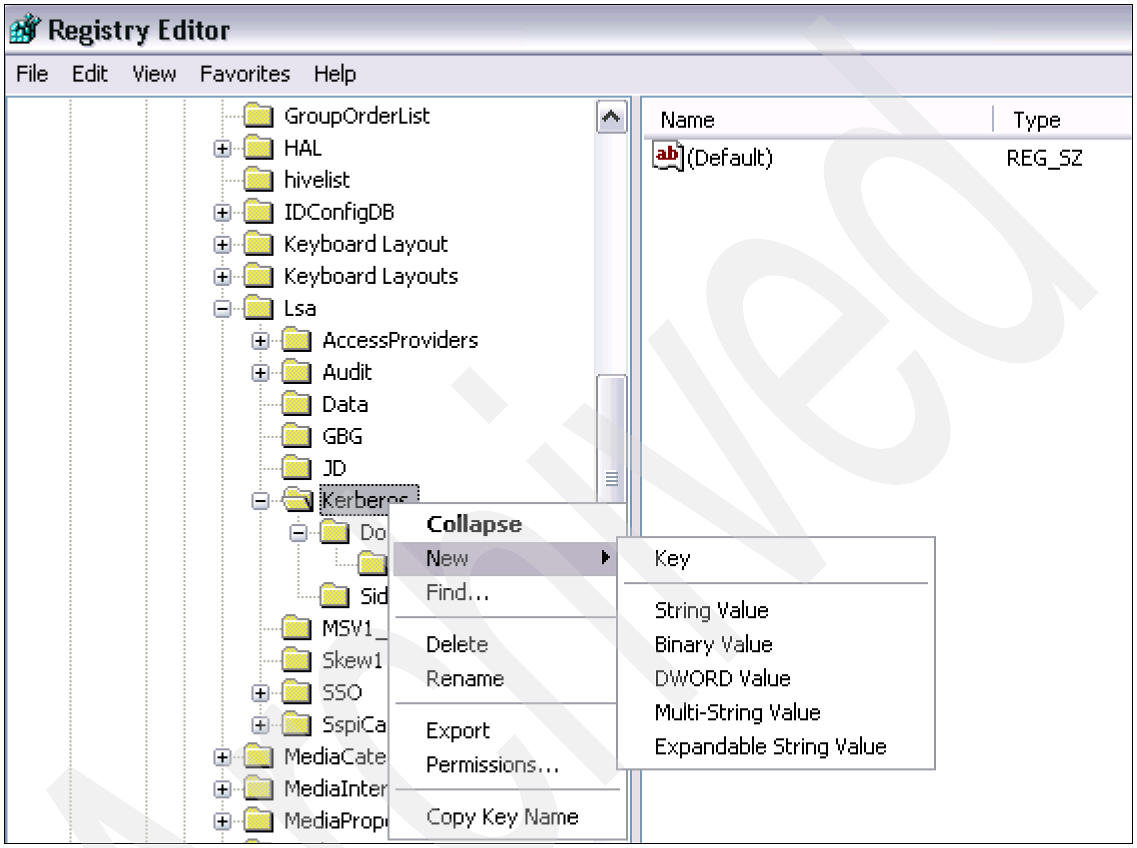


Figure 6-43 Create new key under Kerberos

7. Enter `HostToRealm` for the key name.

8. Right-click **HostToRealm**, and click **New** → **Key**. Enter the z/OS KDC realm name, `ITSO.IBM.COM`, as shown in Figure 6-44.

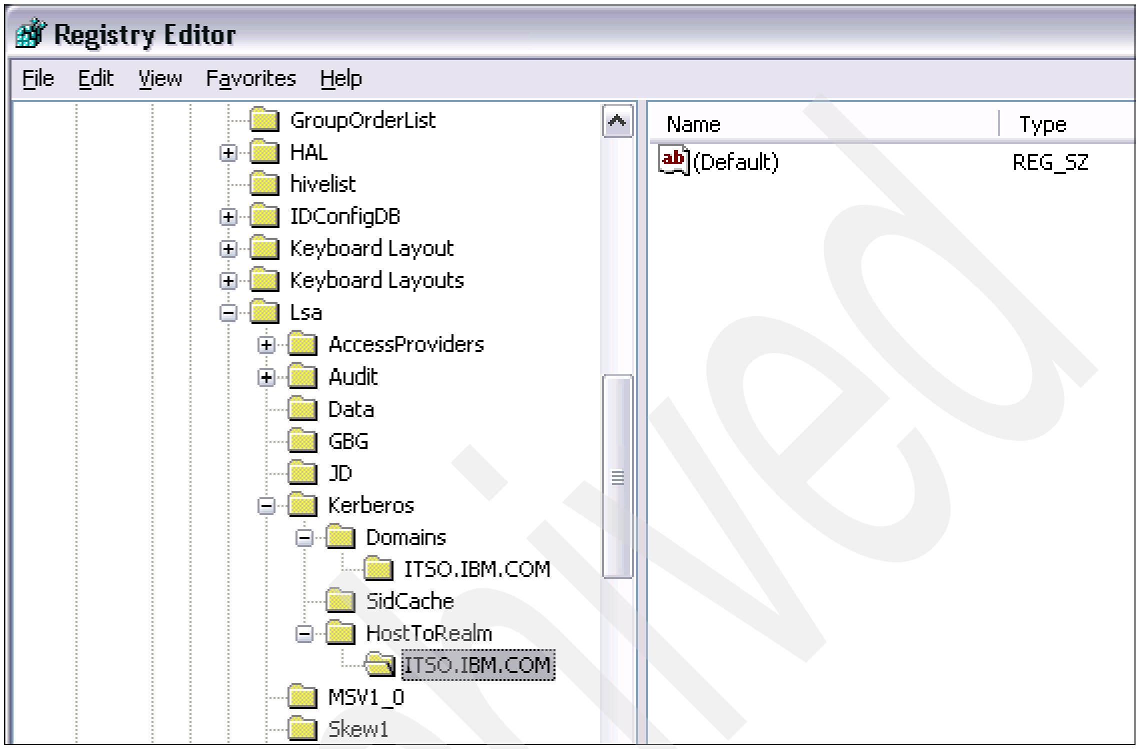


Figure 6-44 Set HostToRealm subkey

9. Right-click the z/OS KDC realm, and click **New** → **Multi-String Value** as shown in Figure 6-45.

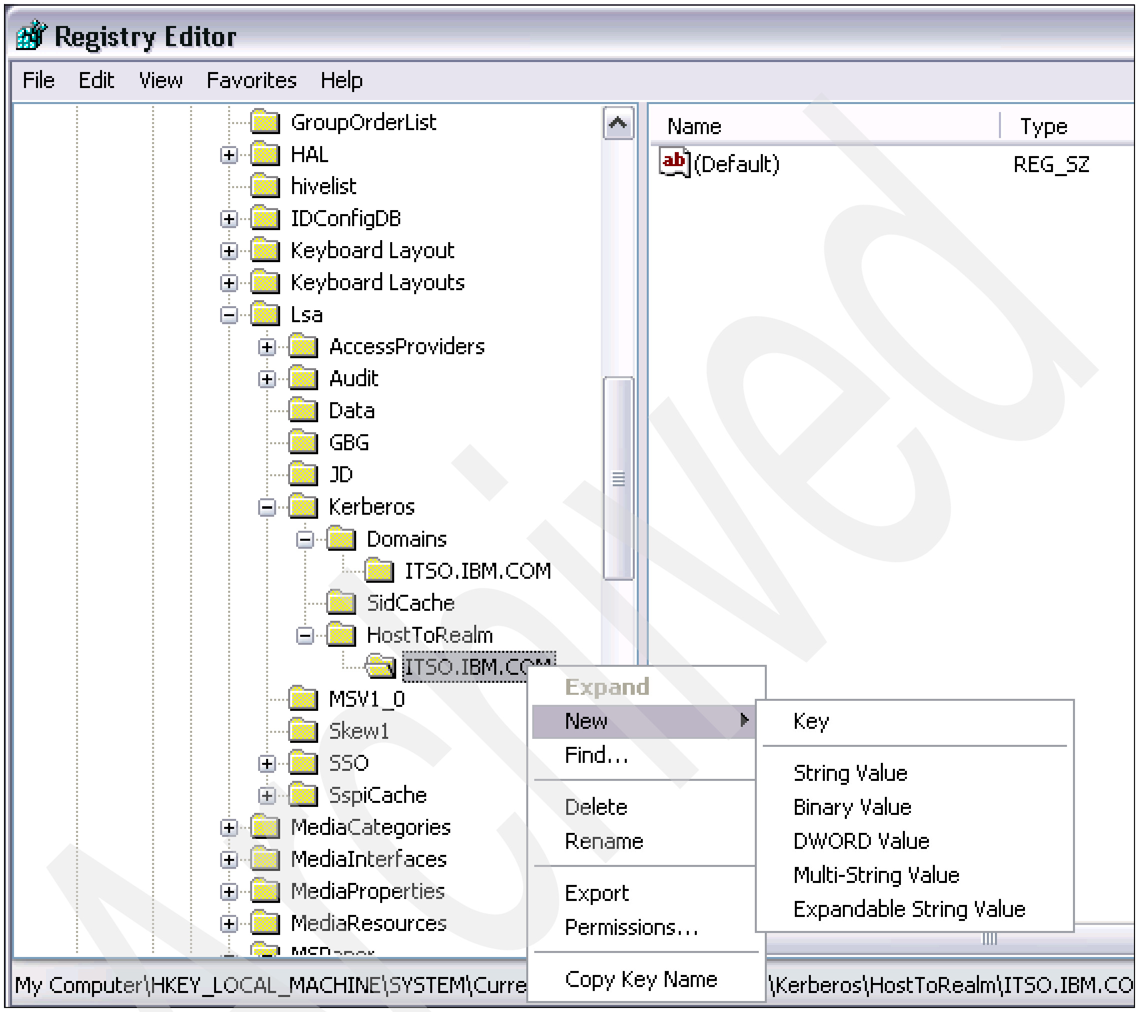
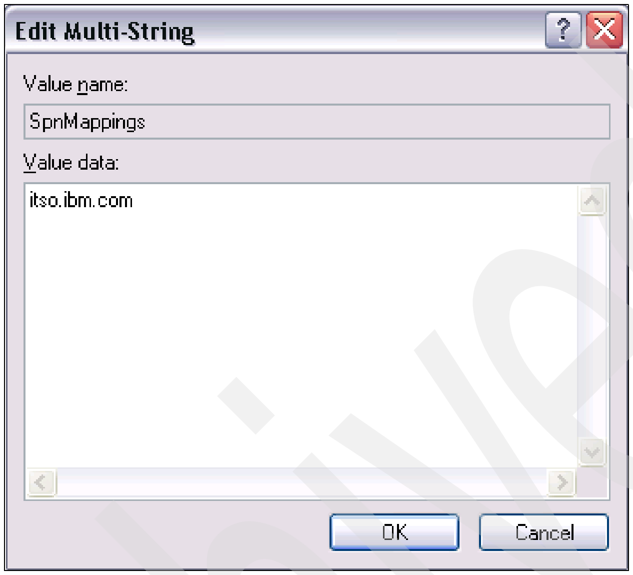


Figure 6-45 Create Multi-String Value key

10. Enter `SpnMappings`, and click **OK**.

11. Double-click **SpnMappings** to modify the value. Enter the z/OS system domain name for the value, and click **OK**. As shown in Figure 6-46, the domain is `itso.ibm.com` because the z/OS system host name is `wtsc60.itso.ibm.com`.



SpnMappings

itso.ibm.com

Figure 6-46 Enter z/OS domain as SpnMappings value

12. Exit **`regedit`**, and restart the Windows client for these changes to take effect.

6.4 Configuring trust on the z/OS KDC

This chapter assumes that the z/OS KDC is already installed, configured, and started. In this example, the z/OS KDC realm, which was created, is ITSO.IBM.COM. The configuration process uses the ISPF command shell to run RACF commands.

To configure trust on the z/OS KDC, follow these steps:

1. Start a TSO session to the z/OS system and log in to ISPF.
2. Select the option for the **ISPF Command Shell**.

3. Run the RDEFINE REALM commands to define the incoming and outgoing cross trust realm between the Windows and z/OS KDC, as follows:

   ```
   RDEFINE REALM /.../<Windows realm>/KRBTGT/<zOS realm>
   KERB(PASSWORD(<trust password>) ENCRYPT(<encrypt type>))
   RDEFINE REALM /.../<zOS realm>/KRBTGT/<Windows realm>
   KERB(PASSWORD(<trust password>) ENCRYPT(<encrypt type>))
   ```

   Specify the values for `<Windows realm>`, `<zOS realm>`, `<trust password>`, and `<encrypt type>`. For the incoming cross trust realm, you do not need to specify the ENCRYPT argument if you do not want to restrict incoming encryption types. However, encrypting the outgoing cross trust realm is recommended, because you do not want to encrypt a ticket that the remote KDC cannot decrypt. The encryption type you specify for the outgoing cross trust realm has to be supported on the Windows Active Directory.

   Note: Remember to use the same trust password that you used when creating the trust on Windows Active Directory. This trust password must be in all uppercase letters.

   Example 6-4 shows an example of these commands with the realms, password, and encryption type set.

   Example 6-4 Define trusted realms

   ```
   RDEFINE REALM /.../KDC2008.ITSO.COM/KRBTGT/ITSO.IBM.COM
   KERB(PASSWORD(MS&Z1TRUST) ENCRYPT(DES NODESD NODES3))
   RDEFINE REALM /.../ITSO.IBM.COM/KRBTGT/KDC2008.ITSO.COM
   KERB(PASSWORD(MS&Z1TRUST) ENCRYPT(DES NODESD NODES3))
   ```

4. Run the RALTER REALM commands to change an existing trust between the Windows realm and z/OS KDC realm, as follows:

   ```
   RALTER REALM /.../<Windows realm>/KRBTGT/<zOS realm>
   KERB(PASSWORD(<trust password>) ENCRYPT(<encrypt type>))
   RALTER REALM /.../<zOS realm>/KRBTGT/<Windows realm>
   KERB(PASSWORD(<trust password>) ENCRYPT(<encrypt type>))
   ```

   Specify the values for `<Windows realm>`, `<zOS realm>`, `<trust password>`, and `<encrypt type>`.

Example 6-5 shows an example of these commands where the encryption type is changed.

Example 6-5 Alter trusted realms

```
RALTER REALM /.../KDC2008.ITSO.COM/KRBTGT/ITSO.IBM.COM
KERB(PASSWORD(MS&Z1TRUST) ENCRYPT(DES NODESD DES3 AES128))
RALTER REALM /.../ITSO.IBM.COM/KRBTGT/KDC2008.ITSO.COM
KERB(PASSWORD(MS&Z1TRUST) ENCRYPT(DES NODESD DES3 AES128))
```

5. You do not need to restart the z/OS KDC after you create or change the cross trust realm. However, the KDC can cache these entries for up to 60 seconds, so wait 60 seconds for these changes to take effect.

6.5 Validating the trust

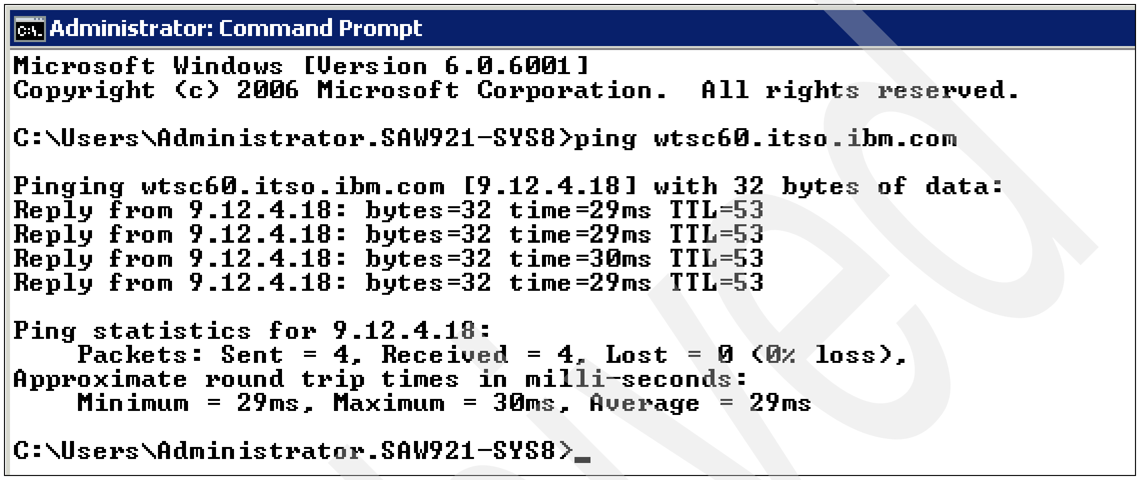
This section includes steps to validate specific areas of the cross realm trust setup.

Note: For a scenario using trust between a Microsoft Kerberos KDC and a z/OS KDC, see Chapter 15, “SSO to WebSphere Application Server for z/OS and DB2 using Microsoft Kerberos KDC and z/OS KDC trust” on page 391.

To validate specific ares of the trust setup, follow these steps:

1. On the Windows Active Directory system, verify the DNS settings and see if you can resolve the z/OS KDC system. Follow these steps:

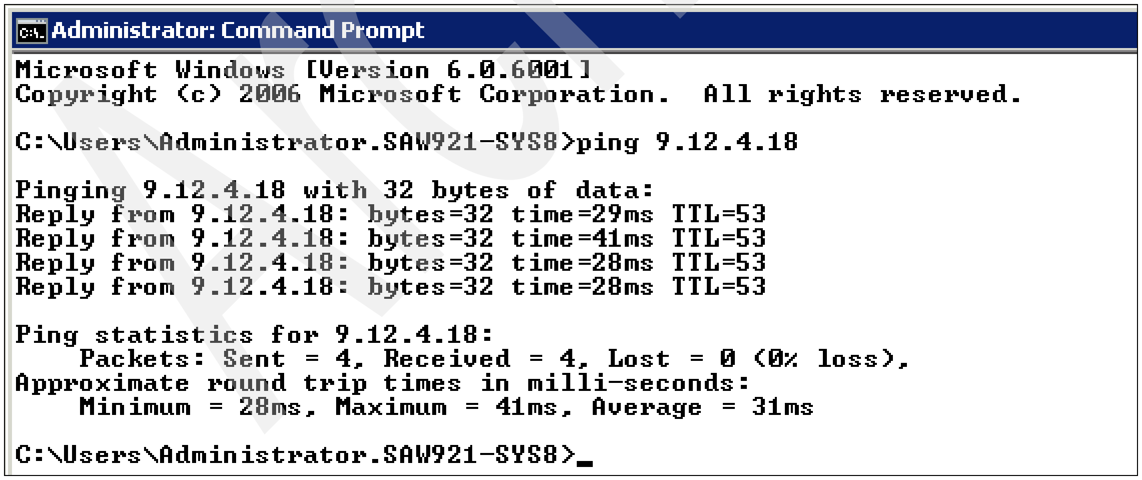
   a. Start a command prompt and ping the z/OS host name, as shown in Figure 6-47.



```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Users\Administrator.SAW921-SYS8>ping wtsc60.itso.ibm.com

Pinging wtsc60.itso.ibm.com [9.12.4.18] with 32 bytes of data:
Reply from 9.12.4.18: bytes=32 time=29ms TTL=53
Reply from 9.12.4.18: bytes=32 time=29ms TTL=53
Reply from 9.12.4.18: bytes=32 time=30ms TTL=53
Reply from 9.12.4.18: bytes=32 time=29ms TTL=53

Ping statistics for 9.12.4.18:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 29ms, Maximum = 30ms, Average = 29ms

C:\Users\Administrator.SAW921-SYS8>_
```

Figure 6-47   PIng z/OS host name

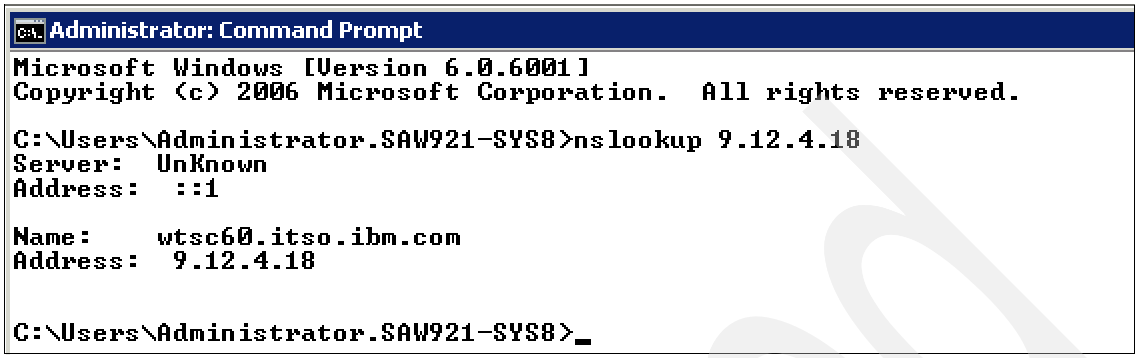
   b. From the command prompt, ping the z/OS system's IP address, as shown in Figure 6-48.



```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Users\Administrator.SAW921-SYS8>ping 9.12.4.18

Pinging 9.12.4.18 with 32 bytes of data:
Reply from 9.12.4.18: bytes=32 time=29ms TTL=53
Reply from 9.12.4.18: bytes=32 time=41ms TTL=53
Reply from 9.12.4.18: bytes=32 time=28ms TTL=53
Reply from 9.12.4.18: bytes=32 time=28ms TTL=53

Ping statistics for 9.12.4.18:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 28ms, Maximum = 41ms, Average = 31ms

C:\Users\Administrator.SAW921-SYS8>_
```

Figure 6-48   PIng z/OS system IP address

c. From the command prompt, run **`nslookup`** on the z/OS system's IP address to see if DNS resolves the host name (Figure 6-49).


Administrator: Command Prompt

```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation.  All rights reserved.

C:\Users\Administrator.SAW921-SYS8>nslookup 9.12.4.18
Server:  UnKnown
Address:  ::1

Name:    wtsc60.itso.ibm.com
Address:  9.12.4.18


C:\Users\Administrator.SAW921-SYS8>_
```

Figure 6-49 Running nslookup to resolve z/OS host name

2. On the z/OS system, run the RLIST REALM command to view an existing trust between the Windows realm and z/OS KDC realm, as follows:

```
RLIST REALM /.../<Windows realm>/KRBTGT/<zOS realm> KERB NORACF
RLIST REALM /.../<zOS realm>/KRBTGT/<Windows realm> KERB NORACF
```

   Specify the values for <*`Windows realm`*> and <*`zOS realm`*>.

   The command output shows the trusted realm name and encryption types. Example 6-6 shows an example of the output.

Example 6-6 List trust realm

```
CLASS      NAME
-----      ----
REALM      /.../KDC2008.ITSO.COM/KRBTGT/ITSO.IBM.COM

KERB INFORMATION
----------------
KEY VERSION= 001
KEY ENCRYPTION TYPE= DES NODES3 NODESD AES128 AES256
```

7

Single sign-on to WebSphere Application Server using SPNEGO

This chapter provides information about the use of single sign-on (SSO) using Simple and Protected GSSAPI Negotiation (SPNEGO) Web authentication with WebSphere Application Server V7. It provides an end-to-end scenario that illustrates how to configure the three main components of the solution:

- The Microsoft Active Directory and key distribution center (KDC)
- WebSphere Application Server V7
- The Web browser client

This scenario is identical to the scenario that we describe in Chapter 8, “Single sign-on to WebSphere Application Server for z/OS using SPNEGO” on page 173, with the exception of the WebSphere Application Server. This chapter is targeted toward the users of WebSphere Application Server on distributed platforms. It includes the following topics:

- Scenario overview
- Configuring the Microsoft Active Directory server
- Configuring WebSphere Application Server
- Configuring Web browsers for SPNEGO
- Validating SSO with SPNEGO Web authentication

7.1 Scenario overview

The scenario that we use in this chapter illustrates SSO capability between a Windows domain and WebSphere Application Server V7. In this scenario, the user logs on to the Microsoft domain controller with an identity known to Microsoft Active Directory (`FXAVIER`). The user then accesses a Web application from a browser, and the user's identity is propagated to WebSphere Application Server. This identity is mapped to Microsoft Active Directory. The scenario allows *FXAVIER* to access secured applications successfully on WebSphere Application Server without being prompted for user ID and password.

Note: This scenario features the following components:

- SSO from a Web browser client to WebSphere Application Server
- Microsoft Kerberos KDC
- A Web browser client
- WebSphere Application Server configuration:
  - User registry: Microsoft Active Directory
  - Authentication mechanism: LTPA
  - SPNEGO
  - Kerberos authentication and configuration file
  - Sample application: DefaultApplication (snoop servlet)

7.1.1 Step-by-step flow for the scenario

Figure 7-1 shows the environment for the scenario in this chapter. The Microsoft domain controller is also the Domain Name System (DNS) and Kerberos KDC. A Windows server hosts WebSphere Application Server V7. The Microsoft Active Directory Domain repository is configured as the current user registry in the global security settings for WebSphere.

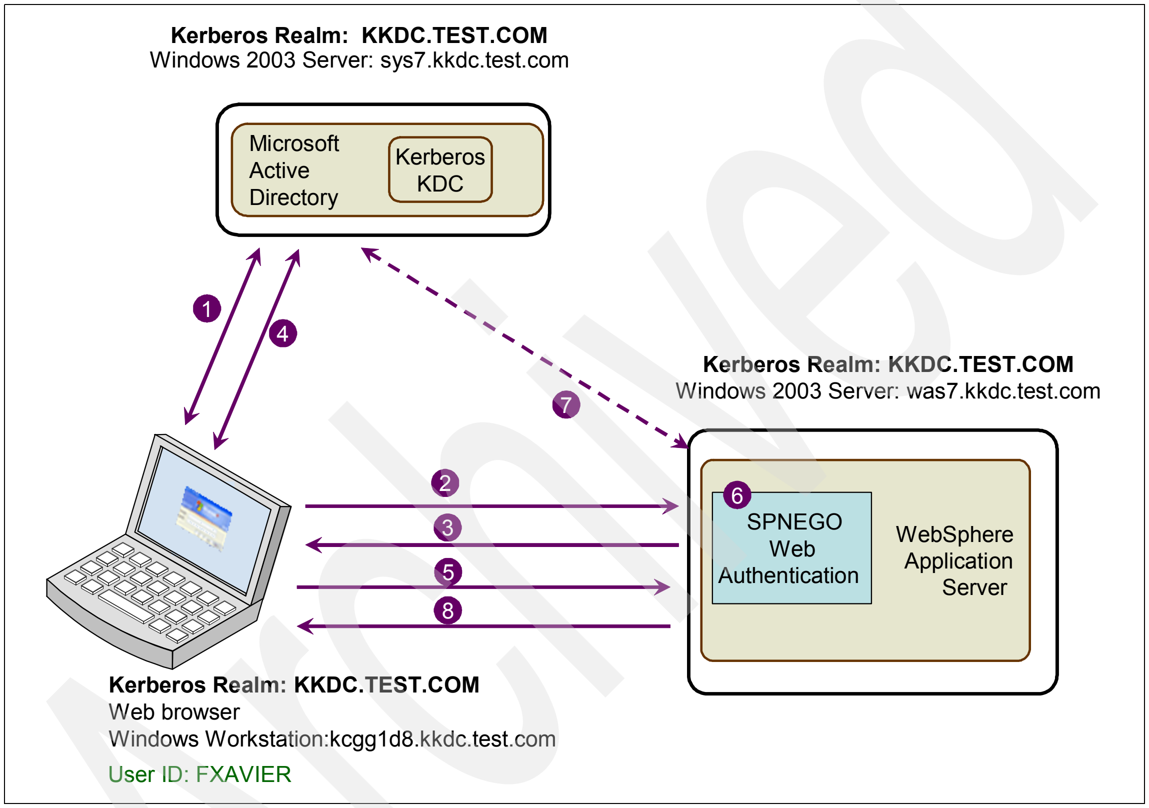

Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: sys7.kkdc.test.com
Microsoft Active Directory
Kerberos KDC
1
4
7
Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: was7.kkdc.test.com
2
3
5
8
6
SPNEGO Web Authentication
WebSphere Application Server
Kerberos Realm: KKDC.TEST.COM
Web browser
Windows Workstation:kcgg1d8.kkdc.test.com
User ID: FXAVIER

Figure 7-1 SPNEGO and Windows SSO scenario

The numbers in the figure correspond to the following steps in the process:

1. To begin, the user logs on to the Windows domain from the workstation. After the user enters a user name and password, the Windows logon sends the information to the workstation local security authority, which passes the request to the Kerberos authentication package.

   The client sends an initial authentication request (AS_REQ), which includes the user's credentials and an encrypted time stamp to the KDC. This request is a request for authentication and a ticket-granting ticket (TGT).

The first Microsoft domain controller in the domain generates the `krbtgt/domain_name` ticket. The KDC uses the secret key to decrypt the time stamp and issues a TGT to the client. The AS_REP is encrypted with the user's key and returned to the user.

2. Next, the user attempts to access the Web application. The user requests a secured Web resource using a browser, which sends an HTTP get request to WebSphere Application Server.
3. WebSphere Application Server and SPNEGO answer to the client browser with an HTTP challenge header 401 containing the `Authenticate: Negotiate` status.
4. The client browser recognizes the negotiate header because the client browser is configured to support integrated Windows authentication. The client parses the requested URL for the host name. The client uses the host name to form the target Kerberos service principal name (SPN), `HTTP/<`*`fully qualified host name`*`>`, to request a Kerberos service ticket from the Kerberos ticket-granting service (TGS) in the Microsoft Kerberos KDC (TGS_REQ). The TGS then issues a service ticket (TGS_REP) to the client.
5. The client gets the SPNEGO token from the previous step. The Kerberos service ticket proves both the user's identity and permissions to the service, and the service's identity to the user. The client responds to the WebSphere Application Server `Authenticate: Negotiate` challenge with the SPNEGO token in the request HTTP header.
6. SPNEGO Web authentication in WebSphere Application Server sees the HTTP header with the SPNEGO token. SPNEGO validates the SPNEGO token and gets the identity (principal) of the user.
7. After WebSphere gets the identity of the user (`FXAVIER`), it validates the user in Microsoft Active Directory. When the identification process is executed, WebSphere executes the related Java code (servlets, JSPs, EJBs, and so on) and checks authorizations.
8. If all access is granted, WebSphere Application Server sends the response with an HTTP 200. WebSphere also includes in the response an LTPA token in the form of a cookie.

For all subsequent requests, by default, the LTPA token is used to access resources securely. Thus, the SPNEGO protocol is executed once for the first request.

The key point to notice here is that the user is not prompted to authenticate to WebSphere Application Server. The user's existing Microsoft Active Directory credentials are sent to the WebSphere Application Server. The user's access uses the Windows Kerberos credentials.

7.1.2 Scenario components

This scenario includes the follow main components:

- The application server

  WebSphere Application Server for Windows V7.0.0.5

- The Kerberos KDC

  Microsoft Windows Server 2003 SP2 with Microsoft Active Directory. The Windows 2003 Support Tools are installed.

- The client workstation

  Microsoft Windows XP Professional 2002 SP2. The Windows Server 2003 Resource Kit Tools are installed. The workstation has both Microsoft Internet Explorer v6.0.2900 and Mozilla Firefox v3.0.6.

  Note: The client workstation must be a separate machine. It cannot be the same as the system where WebSphere Application Server is running.

7.1.3 Basic requirements for this scenario

This scenario requires the following components:

- A functioning Microsoft Windows 2000 or 2003 Active Directory Domain including:

  - A domain controller
  - A client workstation

  Users must be able to log in to the domain controller.

  A working domain controller and at least one client computer in that domain are required. Using SPNEGO from the domain controller does not work.

  For more information, you can find the tutorial *How do I install Active Directory on my Windows Server 2003 server* at:

  `http://www.petri.co.il/how_to_install_active_directory_on_windows_2003.htm`

- A functioning WebSphere Application Server environment with application security enabled

- Access for users from the Microsoft Active Directory to WebSphere Application Server's protected resources using a native authentication mechanism, such as basic authentication (BA) or forms authentication
- Windows 200x Support Tools on the Microsoft Windows Server, including the **`setspn`** and **`ktpass`** tools that this scenario uses

  These support tools are available on the Windows Server installation CD or from the following Microsoft Web site:

  `http://www.microsoft.com/downloads/details.aspx?familyid=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D`

We also recommend that you install the Windows Server 2003 Resource Kit Tools, which provide the **`kerbtray.exe`** utility to see Kerberos tickets on the user workstation. These tools are available at:

`http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd`

7.2 Configuring the Microsoft Active Directory server

In this section, we highlight the main items that you need to configure in the Microsoft Active Directory server.

Perform the following steps on the Microsoft Active Directory server:

1. Create a Microsoft Active Directory user account that maps to the Kerberos service principal name (SPN) for WebSphere Application Server. This account identifies the WebSphere Application Server installation as a Kerberized service. To create this SPN for WebSphere, follow these steps:
   a. Log on as administrator to the Windows 2003 Server system that serves as the Microsoft domain controller.
   b. Invoke the Microsoft Active Directory users and computers snap-in. Click **Start** → **Run**. Then type the following command, and click **OK**:
   ```
   dsa.msc /server=localhost
   ```
   c. Create a user account for WebSphere Application Server V7. In this scenario, we created the *waskerb* user account.
2. Create the Kerberos keytab file. You can use the **`ktpass`** tool to set up the Kerberos keytab file (`krb5.keytab`) for the previously created SPN. With the **`ktpass`** command, you can configure a non-Windows Server 2003 Kerberos service (such as WebSphere applications on a z/OS platform) as a security principal in the Windows Server 2003 Active Directory. This tool generates a

keytab file that contains the shared key of the service. In our scenario, we use the command shown in Example 7-1.

Example 7-1 Output from the ktpass utility for Windows host

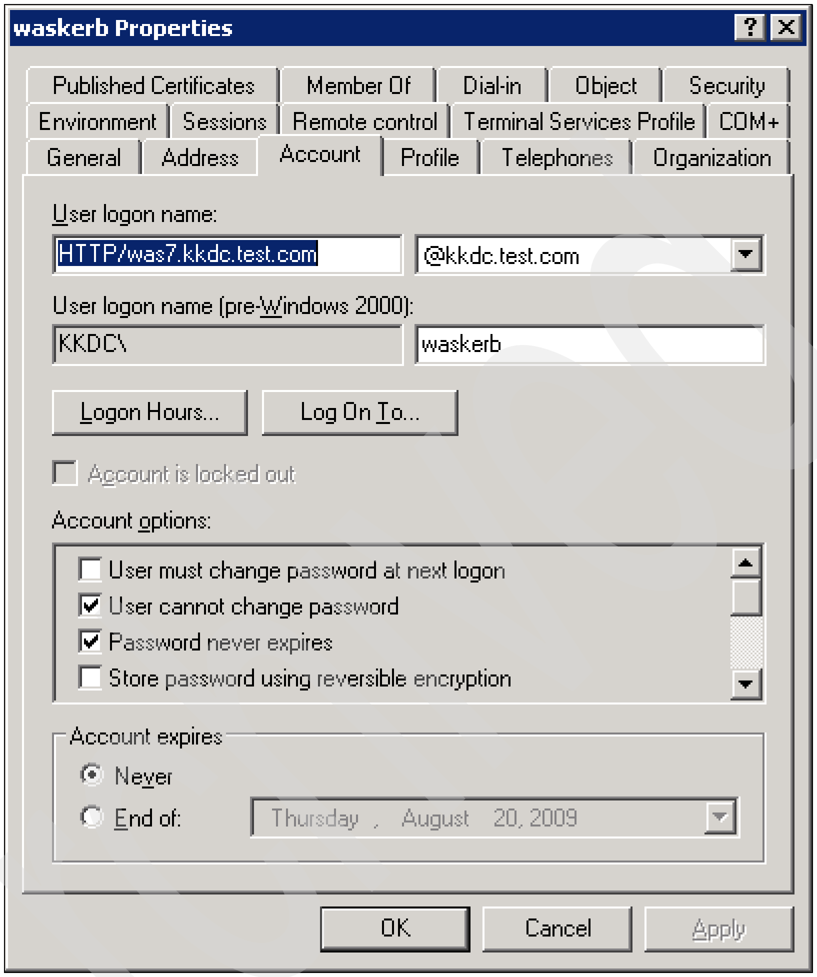
```
ktpass -out c:\waskerb.keytab -princ
HTTP/was7.kkdc.test.com@KKDC.TEST.COM -mapUser waskerb -mapOp set
-pass * -crypto RC4-HMAC-NT
Targeting domain controller: sys7.kkdc.test.com
Using legacy password setting method
Successfully mapped HTTP/was7.kkdc.test.com to waskerb.
Type the password for HTTP/was7.kkdc.test.com:
Type the password again to confirm:
WARNING: pType and account type do not match. This might cause
problems.
Key created.
Output keytab to c:\waskerb.keytab:
Keytab version: 0x502
keysize 72 HTTP/was7.kkdc.test.com@KKDC.TEST.COM ptype 0
(KRB5_NT_UNKNOWN) vno 5
 etype 0x17 (RC4-HMAC) keylength 16
(0x25cf2dc91be14a3509cbada1743bef84)
```

3. Use the **setspn** command to verify the new SPN list, as shown in Example 7-2.

Example 7-2 Verifying SPNs using the -l flag

```
C:\>setspn.exe -l waskerb
Registered ServicePrincipalNames for
CN=waskerb,CN=Users,DC=kkdc,DC=test,DC=com:

HTTP/was7.kkdc.test.com
```

4. Verify the properties of the waskerb user account, as shown in Figure 7-2.


waskerb Properties

User logon name: HTTP/was7.kkdc.test.com @kkdc.test.com

User logon name (pre-Windows 2000): KKDC\ waskerb

☐ 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: Thursday , August 20, 2009

Figure 7-2 WebSphere user waskerb properties

7.3 Configuring WebSphere Application Server

Before you configure SSO, you need to consider the user repository that the WebSphere security domain uses. The simplest option is to use the domain's Microsoft Active Directory registry as the user registry for the server. If you decide to use another registry, you need to ensure that user names in the Microsoft Active Directory are also recognized by the user registry that WebSphere uses.

You need to implement a process to ensure that user mapping exists between registries. This mapping can be one-to-one or many-to-one, depending upon the environments architecture.

7.3.1 Configuring the Kerberos configuration file

To create a Kerberos configuration file for WebSphere's use, follow these steps:

1. Copy the keytab file to WebSphere Application Server system. In this scenario, we copied the file to `c:\windows`.

   Note: A Kerberos keytab configuration file contains a list of keys that are analogous to user passwords. Thus, it is important for hosts to protect their Kerberos keytab files.

2. Use **`wsadmin`** to create the associated Kerberos configuration as shown in Example 7-3. The user ID that is used to connect to the **`wsadmin`** utility must have the Administrator role.

   Example 7-3 Generating the krb5.ini file using wsadmin

```
wsadmin>$AdminTask createKrbConfigFile {-krbPath c:/WINDOWS/krb5.ini
-realm KKDC.TEST.COM -kdcHost sys7.kkdc.test.com -dns kkdc.test.com
-keytabPath c:/WINDOWS/kerbuser.keytab}
c:/WINDOWS/krb5.ini has been
created.
```

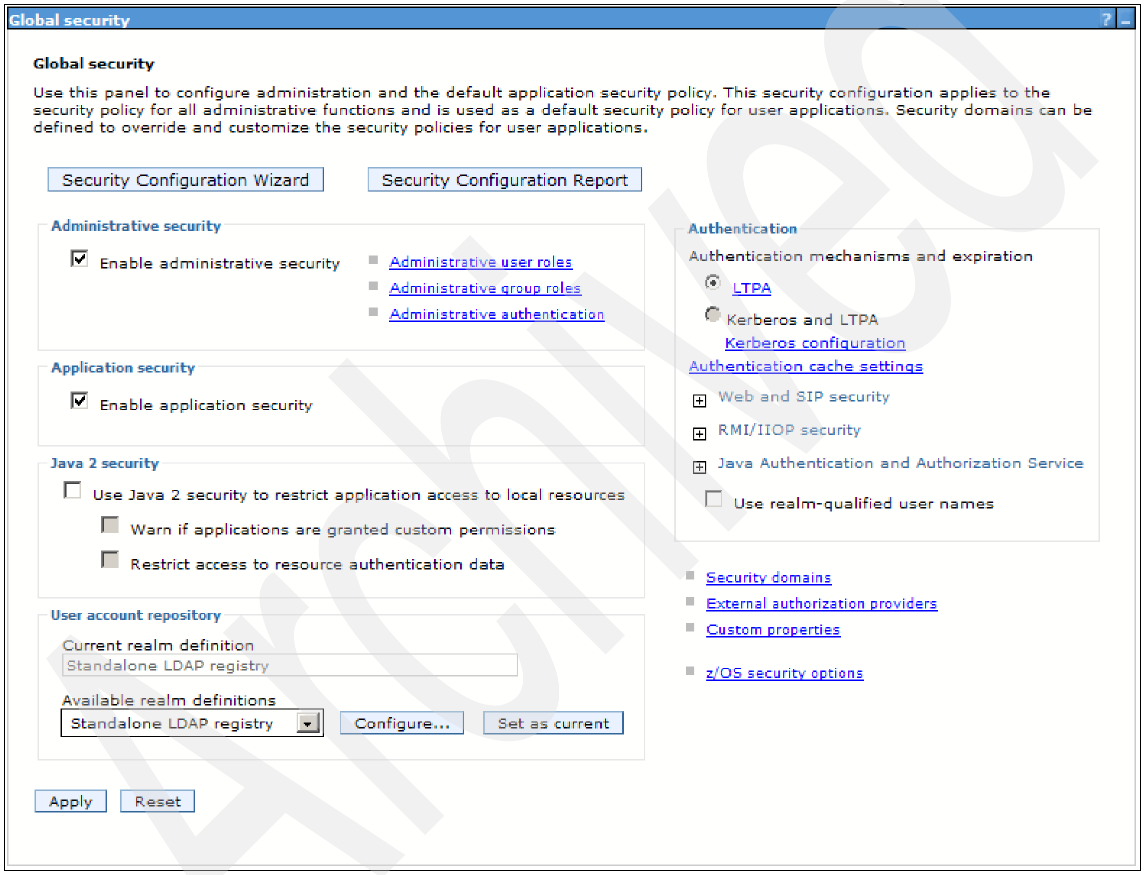
Example 7-4 shows the resulting `krb5.conf` file.

Example 7-4 krb5.conf file content

```
[libdefaults]
   default_realm = KKDC.TEST.COM
   default_keytab_name = FILE:///c:/WINDOWS/waskerb.keytab
   default_tkt_enctypes = rc4-hmac
   default_tgs_enctypes = rc4-hmac
   forwardable  = false
   renewable  = false
   clockskew  = 300
[realms]
   KKDC.TEST.COM = {
      kdc = sys7.kkdc.test.com:88
      default_domain = kkdc.test.com
   }
[domain_realm]
   .kkdc.test.com = KKDC.TEST.COM
```

7.3.2 Configuring the Microsoft Active Directory as the WebSphere Application Server user repository

Next, you need to define the Microsoft Active Directory as an LDAP user registry in WebSphere Application Server and make it the current registry. Log on to the WebSphere Application Server administrative console, and navigate to **Security** → **Global security**. The Global security panel opens, as shown in Figure 7-3.



Global security

Use this panel to configure administration and the default application security policy. This security configuration applies to the security policy for all administrative functions and is used as a default security policy for user applications. Security domains can be defined to override and customize the security policies for user applications.

Administrative security

[✓] Enable administrative security

- Administrative user roles
- Administrative group roles
- Administrative authentication

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

Authentication

Authentication mechanisms and expiration

[✓ LTPA] Kerberos and LTPA

Kerberos configuration

Authentication cache settings

- Web and SIP security
- RMI/IIOP security
- Java Authentication and Authorization Service

[ ] Use realm-qualified user names

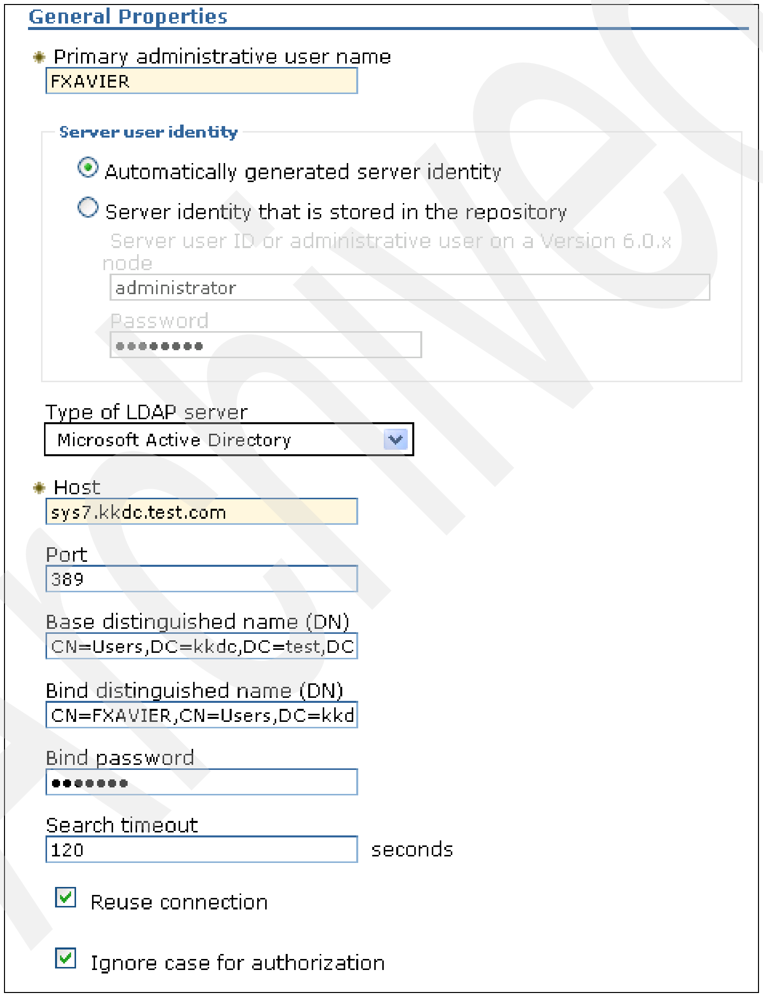
- Security domains
- External authorization providers
- Custom properties
- z/OS security options

Figure 7-3 Global security panel

In this panel, complete the following steps:

1. In the "Authentication" section, verify that the LTPA option is selected. (This option is selected by default.)

2. In the “Web and SIP security” section, click **Single sign-on (SSO)**. Ensure that **Enabled** is selected and that the Domain name is set. In this scenario, the Domain name field is set to `kkdc.test.com`. Return to the Global security panel.
3. Click **External authorization providers**. Then, select **Built-in authorization**, and click **OK**.
4. In the Available realm drop-down menu, select **Standalone LDAP registry**, and click **Configure**. Configure the Microsoft Active Directory as the user registry (as shown in Figure 7-4). Click **OK**.


General Properties

* Primary administrative user name: FXAVIER

Server user identity

[✓ Automatically generated server identity]

Server identity that is stored in the repository

Server user ID or administrative user on a Version 6.0.x node: administrator

Password: ••••••••

Type of LDAP server: Microsoft Active Directory

* Host: sys7.kkdc.test.com

Port: 389

Base distinguished name (DN): CN=Users,DC=kkdc,DC=test,DC

Bind distinguished name (DN): CN=FXAVIER,CN=Users,DC=kkd

Bind password: •••••••

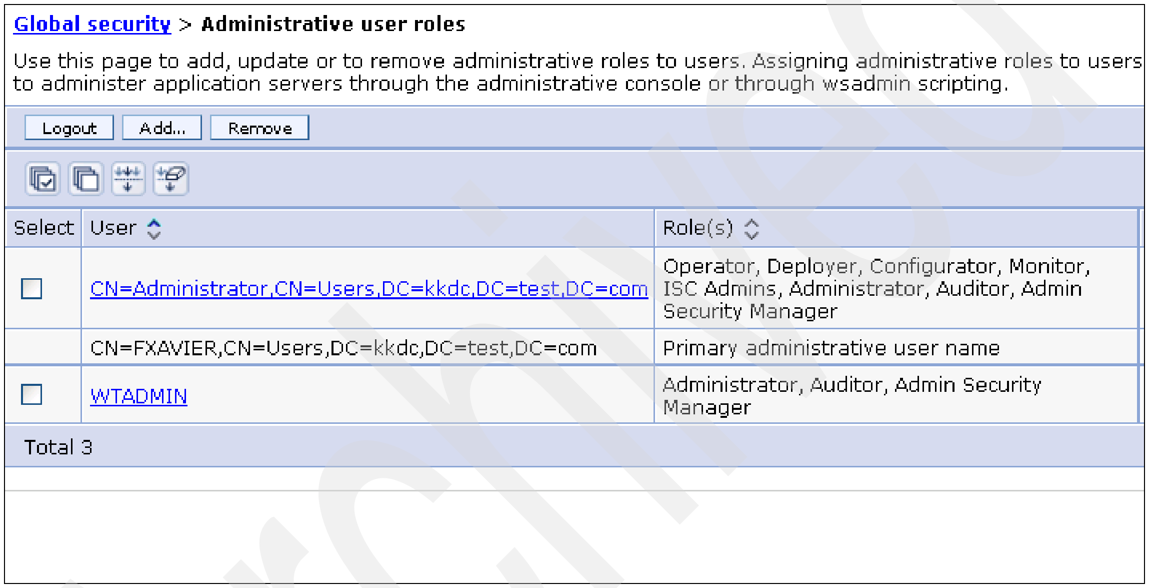
Search timeout: 120 seconds

[✓] Reuse connection

[✓] Ignore case for authorization

Figure 7-4 Microsoft Active Directory as user registry

5. Back in the Global security panel (refer to Figure 7-3), in the Available realms drop-down menu, select **Standalone LDAP registry**, and click **Set as current**.
6. Verify that you have selected the enabled options for both Administrative security and Application security.
7. Click **Apply** to apply all the settings in the Global security panel.
8. In the Administrative security section, click **Administrative user roles**. and assign the Administrator role to at least one user in the Microsoft Active Directory.


Global security > Administrative user roles

Use this page to add, update or to remove administrative roles to users. Assigning administrative roles to users to administer application servers through the administrative console or through wsadmin scripting.

Select	User	Role(s)
☐	CN=Administrator,CN=Users,DC=kkdc,DC=test,DC=com	Operator, Deployer, Configurator, Monitor, ISC Admins, Administrator, Auditor, Admin Security Manager
	CN=FXAVIER,CN=Users,DC=kkdc,DC=test,DC=com	Primary administrative user name
☐	WTADMIN	Administrator, Auditor, Admin Security Manager

Total 3

Figure 7-5 Administrative user roles of Administrator user from Microsoft Active directory

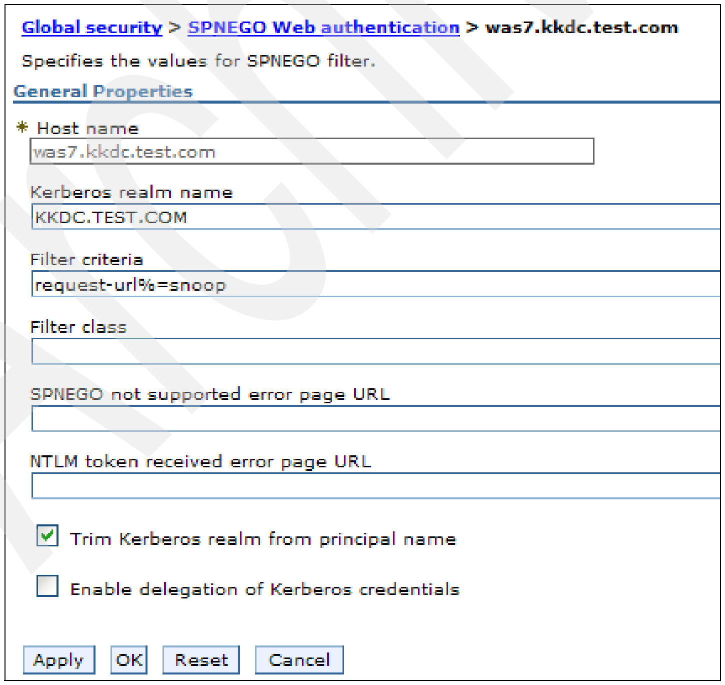
9. Save the configuration and restart WebSphere Application Server V7.

Because this example changed the user registry in the global security configuration, the next time you log on to WebSphere, you need to use one of the IDs defined in step 8.

7.3.3 Configuring the SPNEGO Web authentication in WebSphere Application Server

To configure the SPNEGO Web authentication in WebSphere Application Server using the administrative console, follow these steps:

1. Navigate to **Security** → **Global security**. The Global security panel opens, as shown in Figure 7-3 on page 162. Expand **Web and SIP security**, and then click **SPNEGO Web authentication**.
2. You need to define a filter for each application that will participate in SSO. On the SPNEGO Filters list, click **New**. In the General Properties panel, shown in Figure 7-6, complete the following steps:
   a. Enter the fully qualified host name of the WebSphere Application Server in the Host name field and enter the realm name in the Kerberos realm name field.
   b. Enter `request-url%=snoop` in the Filter criteria field so that requests can be verified by SPNEGO Web authentication.
   c. Select **Trim Kerberos realm from principal name** to remove the suffix of the principal user name.
   d. Click **Apply**.


Global security > SPNEGO Web authentication > was7.kkdc.test.com

Specifies the values for SPNEGO filter.

General Properties

* Host name: was7.kkdc.test.com

Kerberos realm name: KKDC.TEST.COM

Filter criteria: request-url%=snoop

Filter class:

SPNEGO not supported error page URL:

NTLM token received error page URL:

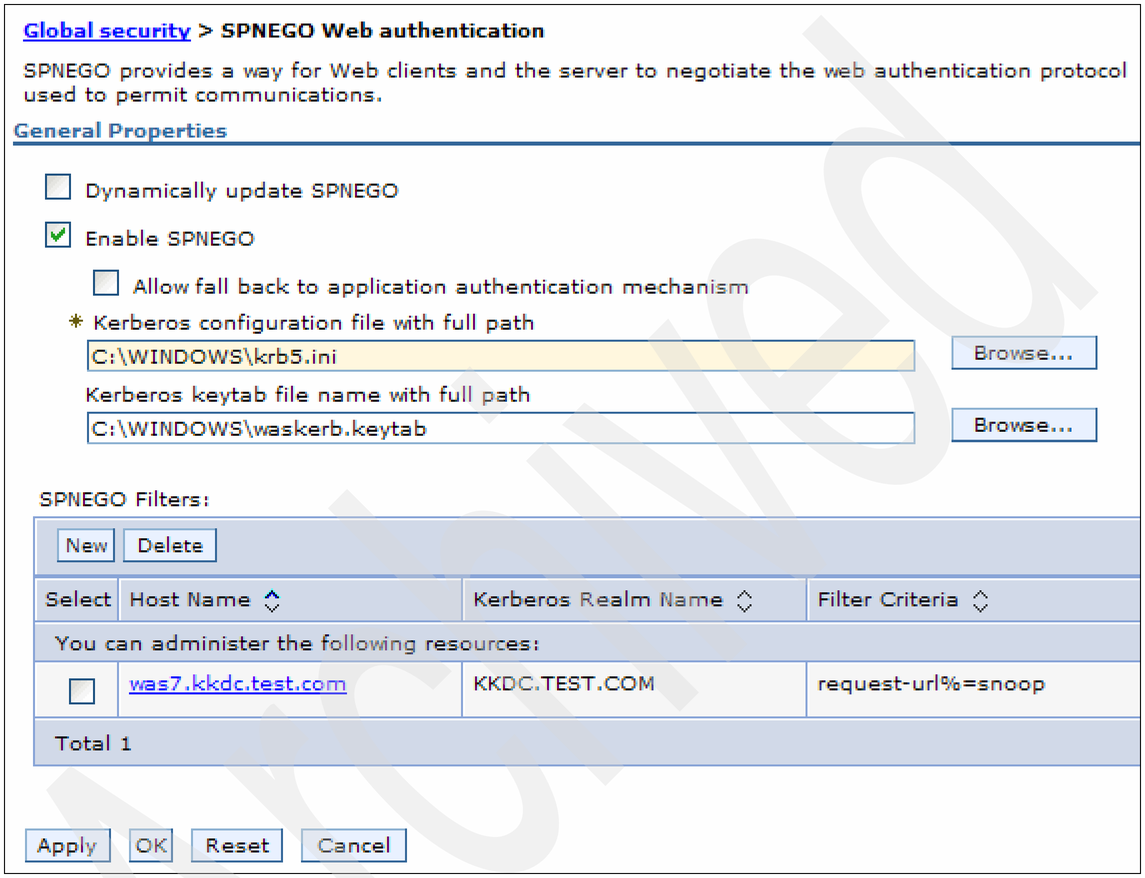
[✓] Trim Kerberos realm from principal name

[ ] Enable delegation of Kerberos credentials

Figure 7-6 Specify SPNEGO filter

The realm name and filter criteria are validated, and you are returned to the SPNEGO Web authentication page.

3. Click **Enable SPNEGO,** and select the Kerberos configuration and keytab file, as shown in Figure 7-7. Then, click **Apply**.


Global security > SPNEGO Web authentication

SPNEGO provides a way for Web clients and the server to negotiate the web authentication protocol used to permit communications.

General Properties

☐ Dynamically update SPNEGO

☑ Enable SPNEGO

☐ Allow fall back to application authentication mechanism

* Kerberos configuration file with full path: C:\WINDOWS\krb5.ini

Kerberos keytab file name with full path: C:\WINDOWS\waskerb.keytab

SPNEGO Filters:

Select	Host Name	Kerberos Realm Name	Filter Criteria
You can administer the following resources:			
☐	was7.kkdc.test.com	KKDC.TEST.COM	request-url%=snoop
Total 1			

Figure 7-7 Enable SPNEGO Web authentication window

Note the “Allow fall back to application authentication mechanism” option. This option specifies that SPNEGO is used to log in to WebSphere Application Server first. If SPNEGO authentication fails, then the authentication mechanism that is defined during application assembly time is used.

Click **OK**.

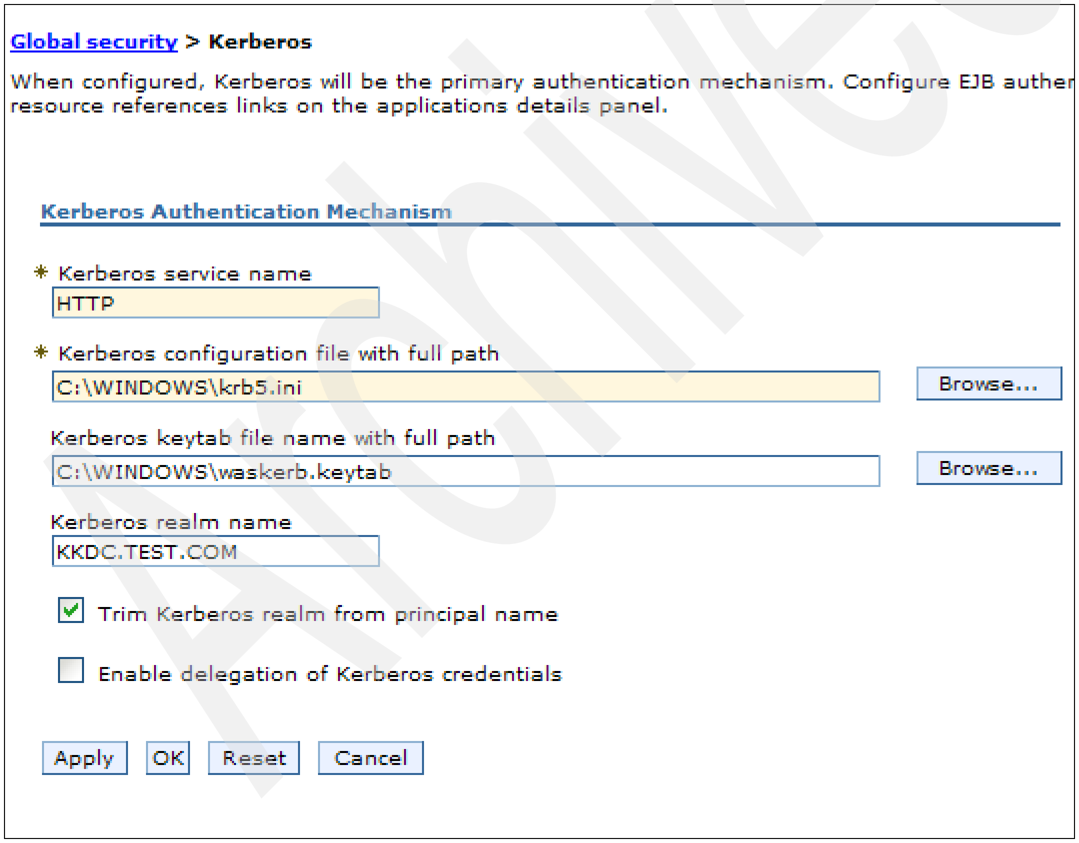
Note: If the “Dynamically update SPNEGO” option is enabled, then when the server recycles, any changes to the SPNEGO configuration are loaded automatically.

4. On the Global security panel, shown in Figure 7-3 on page 162, in the Authentication section, click **Kerberos configuration**. Then, in the Kerberos Authentication Mechanism panel, shown in Figure 7-8, complete the following steps:

   a. Change the Kerberos service name to `HTTP`.

   b. Enter `KKDC.TEST.COM` as the Kerberos realm name.

   c. Select **Trim Kerberos realm from principal name**, and clear **Enable delegation of Kerberos credentials**.



Global security > Kerberos

When configured, Kerberos will be the primary authentication mechanism. Configure EJB auther
resource references links on the applications details panel.

Kerberos Authentication Mechanism

* Kerberos service name
HTTP

* Kerberos configuration file with full path
C:\WINDOWS\krb5.ini

Kerberos keytab file name with full path
C:\WINDOWS\waskerb.keytab

Kerberos realm name
KKDC.TEST.COM

✓ Trim Kerberos realm from principal name

☐ Enable delegation of Kerberos credentials

Figure 7-8 Kerberos configuration panel

5. Save the configuration changes, and in a network deployment cell synchronize the changes to the nodes.
6. Now, stop and start WebSphere Application Server to complete the SPNEGO Web Authentication configuration.

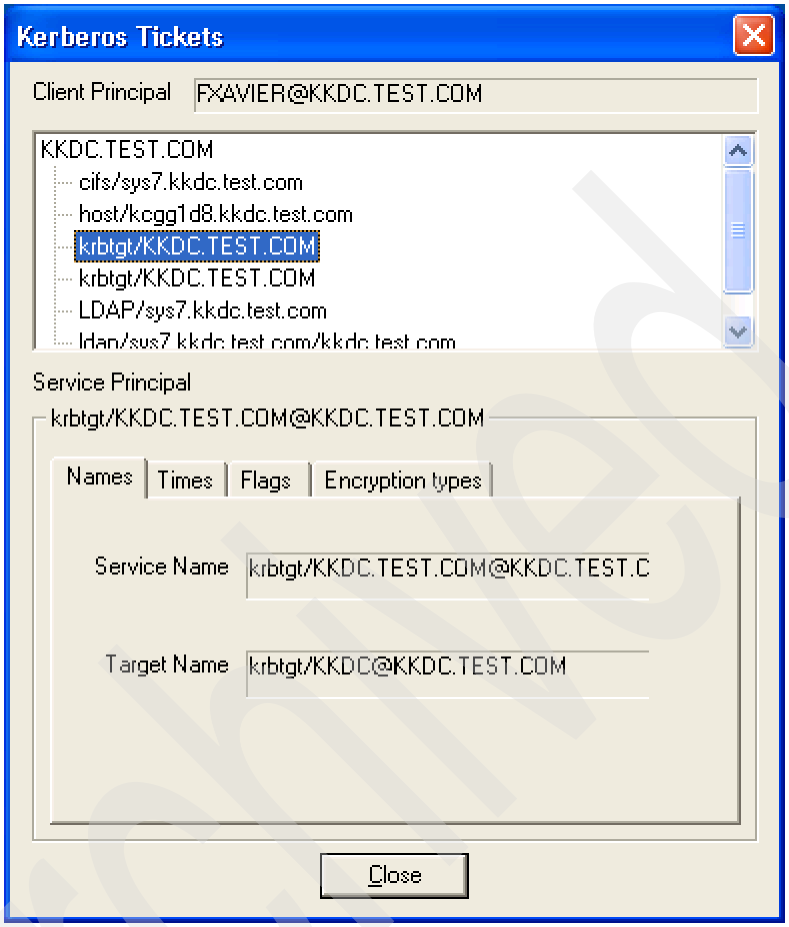
7.4 Configuring Web browsers for SPNEGO

The Web browser is a common client tool used in SSO scenarios and is used to verify the SPNEGO setup for this scenario. First, make sure the client workstation (*kcgg1d8*) is a member of the Microsoft domain controller. Then, configure the Web browser on the client workstation for SPNEGO using the instructions described in Appendix B, “Configuring Web browsers for SPNEGO” on page 499. This scenario can use either Microsoft Internet Explorer or Mozilla Firefox to test SPNEGO authentication with WebSphere Application Server.

7.5 Validating SSO with SPNEGO Web authentication

To validate the SSO scenario between a Windows workstation and the application server, follow these steps:

1. In the user workstation, log on to the Microsoft Active Directory domain. The user ID used in this example is `FXAVIER`.
2. Use the **`kerbtray.exe`** utility to see the Kerberos tickets that are acquired by the user. The **`kerbtray.exe`** utility is available from the Windows Server 2003 Resource Kit Tools. Immediately after logon, you can verify that the workstation obtained the TGT from the Windows KDC. This TGT is named `krbtgt/KKDC.TEST.COM` in our example, as shown in Figure 7-9.


Kerberos Tickets

Client Principal FXAVIER@KKDC.TEST.COM

KKDC.TEST.COM
- cifs/sys7.kkdc.test.com
- host/kcgg1d8.kkdc.test.com
- krbtgt/KKDC.TEST.COM
- krbtgt/KKDC.TEST.COM
- LDAP/sys7.kkdc.test.com
- ldap/sys7.kkdc.test.com/kkdc.test.com

Service Principal

krbtgt/KKDC.TEST.COM@KKDC.TEST.COM

Names | Times | Flags | Encryption types

Service Name krbtgt/KKDC.TEST.COM@KKDC.TEST.C

Target Name krbtgt/KKDC@KKDC.TEST.COM

Figure 7-9 User Kerberos tickets after logon

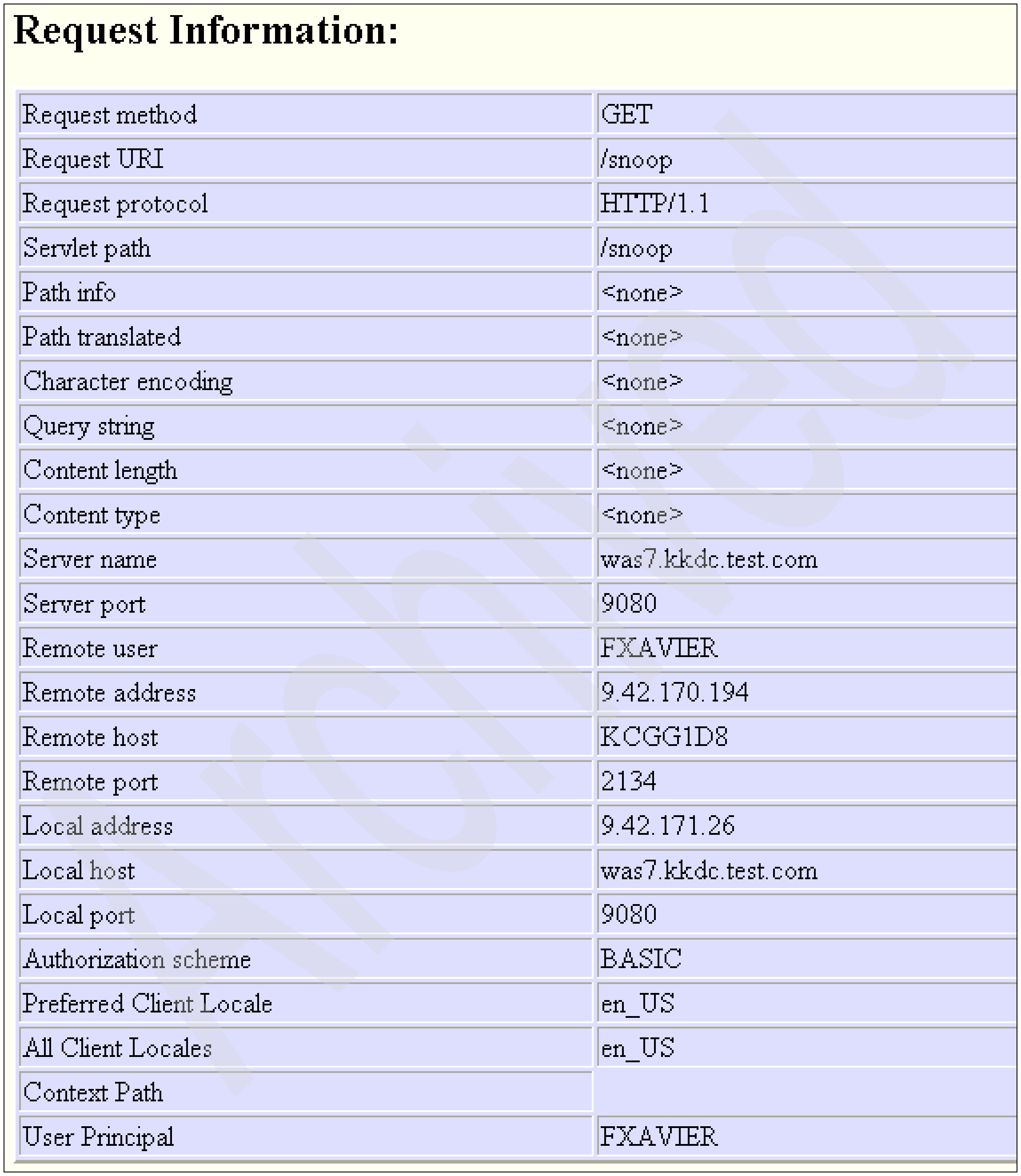
3. Launch a Web browser and enter the URL of the application that you want to access using SSO. In this example, the snoop servlet in the default application is used:

   ```
   http://was7.kkdc.test.com:9080/snoop/
   ```

   When application security is enabled, the snoop servlet asks for authentication.

Because SPNEGO is configured, the browser asks the KDC for a service ticket to use the HTTP service with `was7.kkdc.test.com`. The browser then sends the HTTP request, including a SPNEGO token with the user identity (`FXAVIER`) to the application server. (These steps are described in Figure 7-1 on page 155.)

Figure 7-10 shows the output of the snoop servlet.



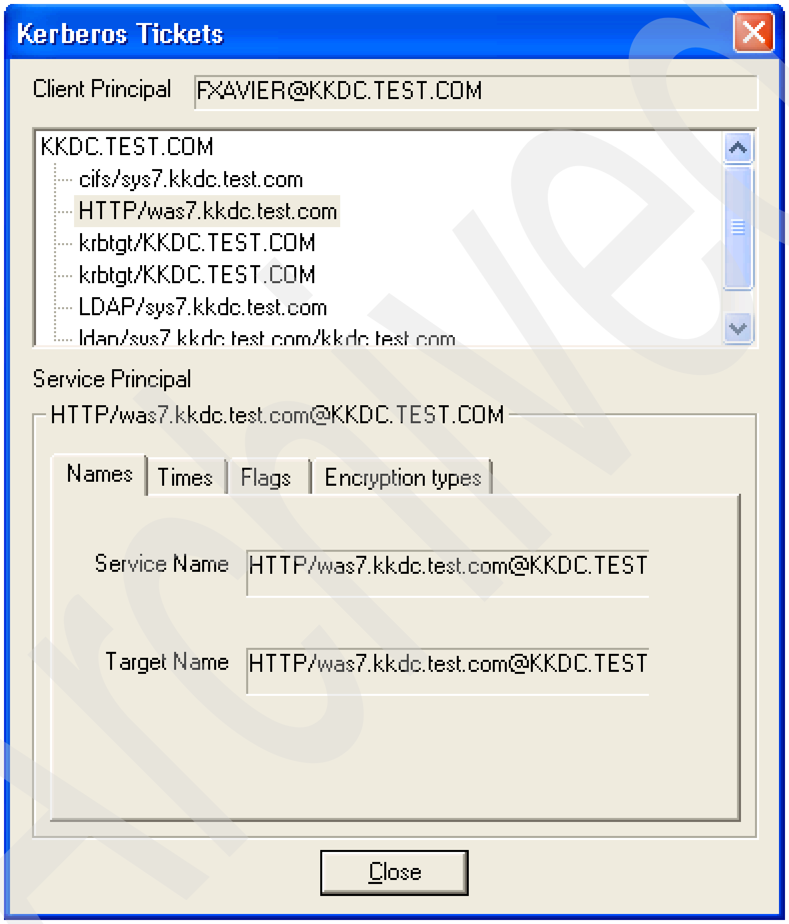
Request Information:

Request method	GET
Request URI	/snoop
Request protocol	HTTP/1.1
Servlet path	/snoop
Path info	<none>
Path translated	<none>
Character encoding	<none>
Query string	<none>
Content length	<none>
Content type	<none>
Server name	was7.kkdc.test.com
Server port	9080
Remote user	FXAVIER
Remote address	9.42.170.194
Remote host	KCGG1D8
Remote port	2134
Local address	9.42.171.26
Local host	was7.kkdc.test.com
Local port	9080
Authorization scheme	BASIC
Preferred Client Locale	en_US
All Client Locales	en_US
Context Path	
User Principal	FXAVIER

Figure 7-10 Internet Explorer Snoop servlet display

The output shows the authenticated user (`FXAVIER`) from the Windows domain, indicating that SSO occurred between the Windows domain and WebSphere Application Server using the SPNEGO.

Using the **`kerbtray.exe`** utility again, you can see the Kerberos tickets that are acquired by the user for the request (Figure 7-11). In this example, the utility shows that the user obtained a service ticket to access the WebSphere Application Server Kerberized service (`HTTP/was7.kkdc.test.com`).



Client Principal FXAVIER@KKDC.TEST.COM

```
KKDC.TEST.COM
  cifs/sys7.kkdc.test.com
  HTTP/was7.kkdc.test.com
  krbtgt/KKDC.TEST.COM
  krbtgt/KKDC.TEST.COM
  LDAP/sys7.kkdc.test.com
  ldap/sys7.kkdc.test.com/kkdc.test.com
```

Service Principal HTTP/was7.kkdc.test.com@KKDC.TEST.COM

Service Name HTTP/was7.kkdc.test.com@KKDC.TEST

Target Name HTTP/was7.kkdc.test.com@KKDC.TEST

Figure 7-11 Kerberos ticket HTTP/was7.kkdc.test.com assigned to FXAVIER

These steps validate the scenario shown in Figure 7-1 on page 155 and confirm SSO between the Windows domain and WebSphere Application Server V7.

8

Single sign-on to WebSphere Application Server for z/OS using SPNEGO

This chapter provides information about the use of single sign-on (SSO) using Simple and Protected GSSAPI Negotiation (SPNEGO) Web authentication with WebSphere Application Server V7. It includes a complete scenario that illustrates how to configure the three main components of the solution:

- Microsoft Windows key distribution center (referred to as *Active Directory*)
- WebSphere Application Server V7 for z/OS
- The client

This scenario is identical to the scenario described in Chapter 7, “Single sign-on to WebSphere Application Server using SPNEGO” on page 153, with the exception of the WebSphere Application Server. This chapter is targeted toward the users of WebSphere Application Server on z/OS platforms. It includes the following topics:

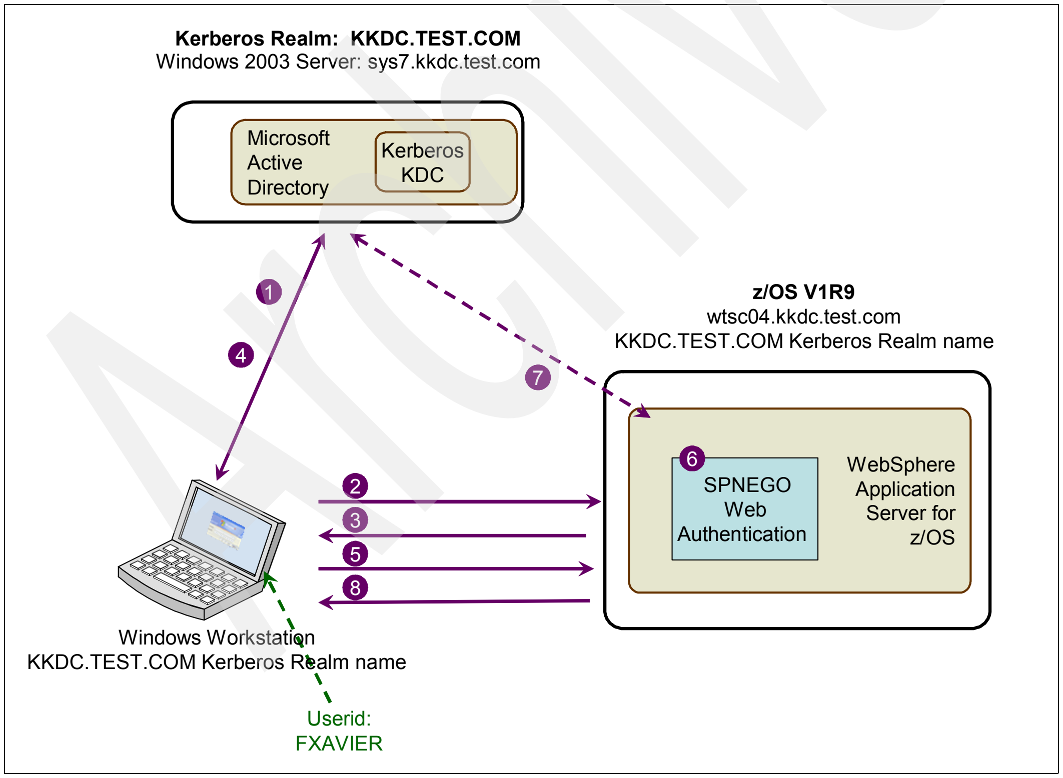
- Scenario overview
- Configuring the Microsoft Active Directory server
- Configuring WebSphere Application Server
- Configuring Web browsers for SPNEGO
- Validating SSO with SPNEGO Web authentication

8.1 Scenario overview

The scenario that we use in this chapter shows how to enable SPNEGO Web authentication in WebSphere Application Server, so that SPNEGO-enabled clients that are logged in to an Microsoft Active Directory domain can access secure WebSphere applications without entering a second password. This scenario illustrates SSO capability between a Windows domain and WebSphere Application Server V7 for z/OS. In this scenario, the user logs on to the Microsoft domain controller with an identity known to Microsoft Active Directory (`FXAVIER`). The user then accesses a Web application from a browser, and the user's identity is sent to WebSphere Application Server. The goal allows `FXAVIER` to access secured applications successfully on WebSphere Application Server without being prompted for user ID and password.

8.1.1 Step-by-step flow of the scenario

Figure 8-1 illustrates the basic flow of this scenario.



Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: sys7.kkdc.test.com
Microsoft Active Directory
Kerberos KDC
1
4
7
z/OS V1R9
wtsc04.kkdc.test.com
KKDC.TEST.COM Kerberos Realm name
6
SPNEGO Web Authentication
WebSphere Application Server for z/OS
2
3
5
8
Windows Workstation
KKDC.TEST.COM Kerberos Realm name
Userid: FXAVIER

Figure 8-1 SPNEGO and Windows SSO scenario

The numbers in the figure correspond to the following steps in this process:

1. To begin, the user logs on to the Windows domain from the workstation. After the user enters a user name and password, Windows logon sends the information to the workstation local security authority, which passes the request to the Kerberos authentication package.

   The client sends an initial authentication request (AS_REQ), which includes the user's credentials and an encrypted time stamp to the key distribution center (KDC). This request is a request for authentication and a ticket-granting ticket (TGT).

   The first Microsoft domain controller in the domain generates the `krbtgt/`*`domain_name`* ticket. The KDC uses the secret key to decrypt the time stamp and issues a TGT to the client. The AS_REP is encrypted with the user's key and returned to the user.
2. Next, the user attempts to access the Web application. The user requests a secured Web resource using a browser, which sends an HTTP get request to WebSphere Application Server.
3. WebSphere Application Server and SPNEGO answer the client browser with an HTTP challenge header 401 that contains the `Authenticate: Negotiate` status.
4. The client browser recognizes the negotiate header because the header is configured to support integrated Windows authentication. The client parses the requested URL for the host name. The client uses the host name to form the target Kerberos service principal name (SPN), `HTTP/<`*`fully qualified host name`*`>`, to request a Kerberos service ticket from the Kerberos ticket-granting service (TGS) in the Microsoft Kerberos KDC (TGS_REQ). The TGS then issues a service ticket (TGS_REP) to the client.
5. The client gets the SPNEGO token from the previous step. The Kerberos service ticket proves both the user's identity and permissions to the service, and the service's identity to the user. The client responds to the WebSphere Application Server `Authenticate: Negotiate` challenge with the SPNEGO token in the request HTTP header.
6. WebSphere Application Server and SPNEGO see the HTTP header with the SPNEGO token. WebSphere Application Server validates the SPNEGO token and gets the identity (principal) of the user.
7. After WebSphere gets the identity of the user, it validates the user in Microsoft Active Directory, `FXAVIER` in our example. When the identification process is executed, WebSphere executes the related Java code (servlets, JSPs, EJBs, and so on) and checks authorizations.
8. If all access is granted, WebSphere Application Server sends the response with an HTTP 200. WebSphere also includes in the response an LTPA token in the form of a cookie.

For all subsequent requests, by default, the LTPA token is used to access resources securely. Thus, the SPNEGO protocol is executed once for the first request.

The key point to notice here is that the user is not prompted to authenticate to WebSphere Application Server. The user's existing Microsoft Active Directory credentials are sent to WebSphere Application Server. The user's access uses the Windows Kerberos credentials.

8.1.2 Scenario components

This scenario includes the following main components:

- The application server

  WebSphere Application Server for z/OS V7.0.0.5. The operating system is z/OS V1r9.

- The Kerberos KDC

  Microsoft Windows Server 2003 SP2 with Active Directory. The Windows 2003 Support Tools are installed.

- The client workstation

  Microsoft Windows XP Professional 2002 SP2. The Windows Server 2003 Resource Kit Tools are installed. It runs Microsoft Internet Explorer v6.0.2900 and Mozilla Firefox v3.0.6.

Note: The client workstation must be separate machine. It cannot be same as the system where WebSphere Application Server is running.

8.1.3 Basic requirements for this scenario

This scenario requires the following components:

- A functioning Microsoft Windows 2000 or 2003 Active Directory Domain including:
  - A domain controller
  - A client workstation

  Users must be able to log in to the domain controller.

  A working domain controller and at least one client computer in that domain are required. Using SPNEGO from the domain controller does not work.

For more information, you can find the tutorial *How do I install Active Directory on my Windows Server 2003 server* at:

http://www.petri.co.il/how_to_install_active_directory_on_windows_2003.htm

- A functioning WebSphere Application Server for z/OS running V7.0.0.5 with global security configured to the Microsoft Active Directory server
- Access for users from the Microsoft Active Directory to WebSphere Application Server's protected resources using a native authentication mechanism, such as basic authentication (BA) or forms authentication
- Windows 200x Support Tools on the Microsoft Windows Server, including the **`setspn`** and **`ktpass`** tools that we use in this scenario

  These support tools are available on the Windows Server installation CD or from the Microsoft Web site:

  http://www.microsoft.com/downloads/details.aspx?familyid=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D

We also recommend that you install the Windows Server 2003 Resource Kit Tools, which provide the **`kerbtray.exe`** utility to see Kerberos tickets on the user workstation. You can install this package on the user workstation. These tools are available at:

http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd

8.1.4 Summary of the implementation steps

Implementing SSO for a WebSphere Application Server environment using SPNEGO requires the following high-level steps:

1. Configure a Windows Server with Microsoft Active Directory domain and associated Kerberos.
2. Configure the client applications (.NET, Web service, and browser) to support the SPNEGO Web authentication mechanism.
3. Configure SPNEGO Web authentication for WebSphere Application Server.

8.2 Configuring the Microsoft Active Directory server

To configure the Microsoft Active Directory server, follow these steps:

1. Create the WebSphere Application Server SPNEGO Service Principal Name as follows:
   a. Create a user in Microsoft Active Directory.
   b. Map the SPNEGO SPN to this user, for example:

      ```
      HTTP/<fully qualified hostname>
      ```

      Refer to 4.6.1, "Creating a Kerberos service principal name" on page 83 for more details about how to complete this step in Microsoft KDC.
2. Create the Kerberos Keytab file for the SPN using one of the following options:
   - On the Microsoft KDC, copy the `keytab` file to the WebSphere Application Server workstation as described in 4.6.2, "Creating the Kerberos keytab file" on page 86.
   - Use the Java **`ktab`** tool on the WebSphere Application Server workstation as described in 2.3.2, "Creating a Kerberos keytab file" on page 19.

Attention: Map the HTTP service name to only one host name. If there is more than one entry, remove the user and recreate the `keytab` file again.

8.3 Configuring WebSphere Application Server

To configure WebSphere Application Server for SPNEGO Web authentication, you need to complete the following tasks:

- Configure the Kerberos configuration file
- Configure Microsoft Active Directory as the WebSphere user repository
- Configure the SPNEGO Web authentication in WebSphere

We describe how to complete these tasks in this section. You can find more information about SPNEGO configuration requirements and settings at:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tsec_SPNEGO_config.html

8.3.1 Configure the Kerberos configuration file

You need to update the Kerberos configuration properties with the relevant values for the KDC (host name and port), the location of the actual Kerberos `keytab` file that was copied from the Windows 2003 server, as well as the Kerberos realm name that is the related Microsoft domain controller.

Use the **`wsadmin`** utility to generate the Kerberos configuration file as follows:

1. Transfer, in binary format, the previously generated `keytab` file from the Windows server to the z/OS partition in UNIX System Services. For example, in this scenario, we put the `keytab` file in `/wasconfig/wtcell/wtdmnode`.

   Note: A Kerberos keytab configuration file contains a list of keys that are analogous to user passwords. It is important for hosts to protect their Kerberos keytab files.

2. Connect to WebSphere Application Server. Use the **`createKrbConfigFile`** command to create a Kerberos configuration file as shown in Example 8-1. The user ID that is used to connect to the **`wsadmin`** utility must have the Administrator role.

   Example 8-1 The wsadmin command execution output

   ```
   wsadmin>$AdminTask createKrbConfigFile {-krbPath
   /wasconfig/wtcell/wtdmnode/krb5.conf -realm KKDC.TEST.COM -kdcHost
   sys7.kkdc.test.com -dns kkdc.test.com -keytabPath
   /wasconfig/wtcell/wtdmnode/wtsc04.keytab}

   /wasconfig/wtcell/wtdmnode/krb5.conf has been created.
   ```

3. Make sure that WebSphere user IDs for the controller region and servant region have sufficient read access to the folder where you put the configuration files.

Example 8-2 shows the resulting `krb5.conf` file.

Example 8-2 The krb5.conf file content

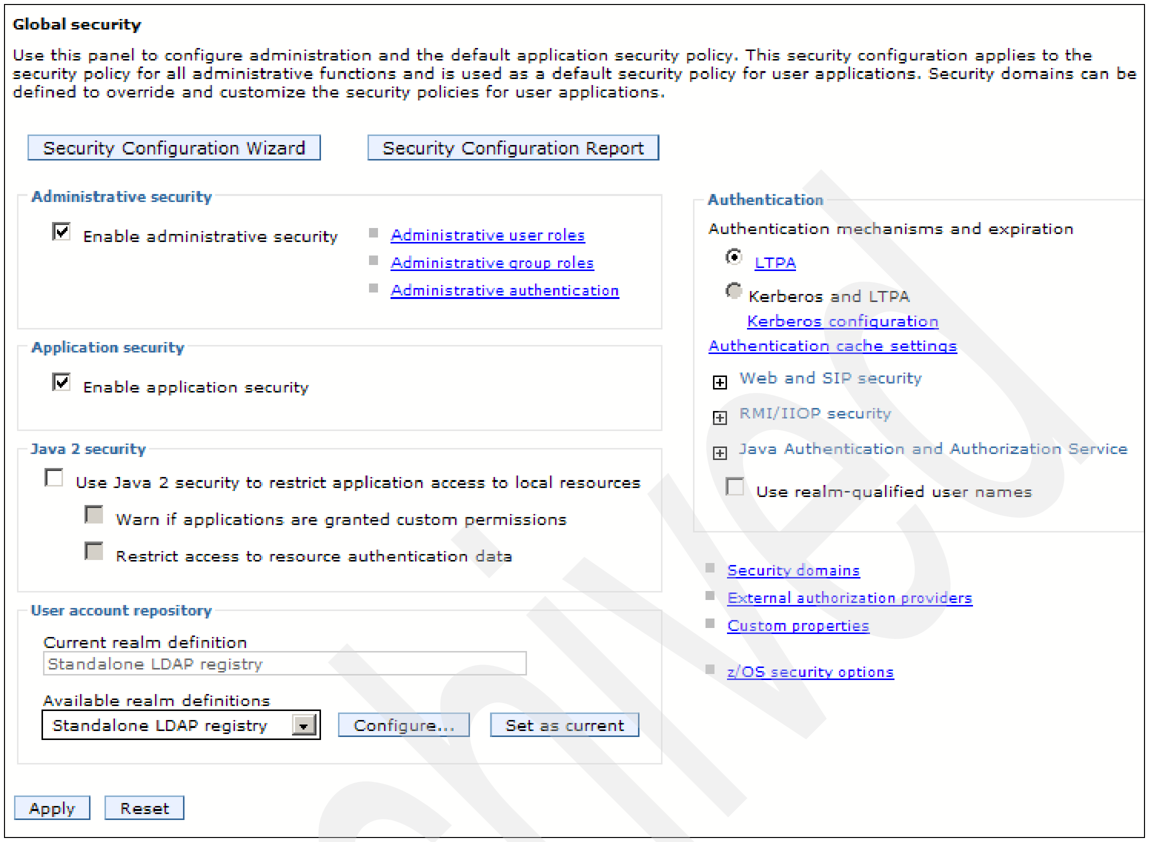
```
[libdefaults]
        default_realm = KKDC.TEST.COM
        default_keytab_name =
FILE:/wasconfig/wtcell/wtdmnode/wtsc04.keytab
        default_tkt_enctypes = rc4-hmac des-cbc-md5
        default_tgs_enctypes = rc4-hmac des-cbc-md5
        forwardable  = true
        renewable  = true
```

```
        noaddresses = true
        clockskew  = 300
[realms]
        KKDC.TEST.COM = {
                kdc = sys7.kkdc.test.com:88
                default_domain = kkdc.test.com
        }
[domain_realm]
        .kkdc.test.com = KKDC.TEST.COM
```

8.3.2 Configure Microsoft Active Directory as the WebSphere user repository

Although Microsoft Active Directory might already be configured in your environment, you need to verify the settings using the steps that we describe in this section.

Log on to the WebSphere Application Server administrative console, and navigate to **Security** → **Global security**. The Global security panel opens as shown in Figure 8-2.


Global security

Use this panel to configure administration and the default application security policy. This security configuration applies to the security policy for all administrative functions and is used as a default security policy for user applications. Security domains can be defined to override and customize the security policies for user applications.

Security Configuration Wizard | Security Configuration Report

Administrative security

[✓] Enable administrative security

- Administrative user roles
- Administrative group roles
- Administrative authentication

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

Authentication mechanisms and expiration

[✓ LTPA]

Kerberos and LTPA

Kerberos configuration

Authentication cache settings

⊞ Web and SIP security

⊞ RMI/IIOP security

⊞ Java Authentication and Authorization Service

[ ] Use realm-qualified user names

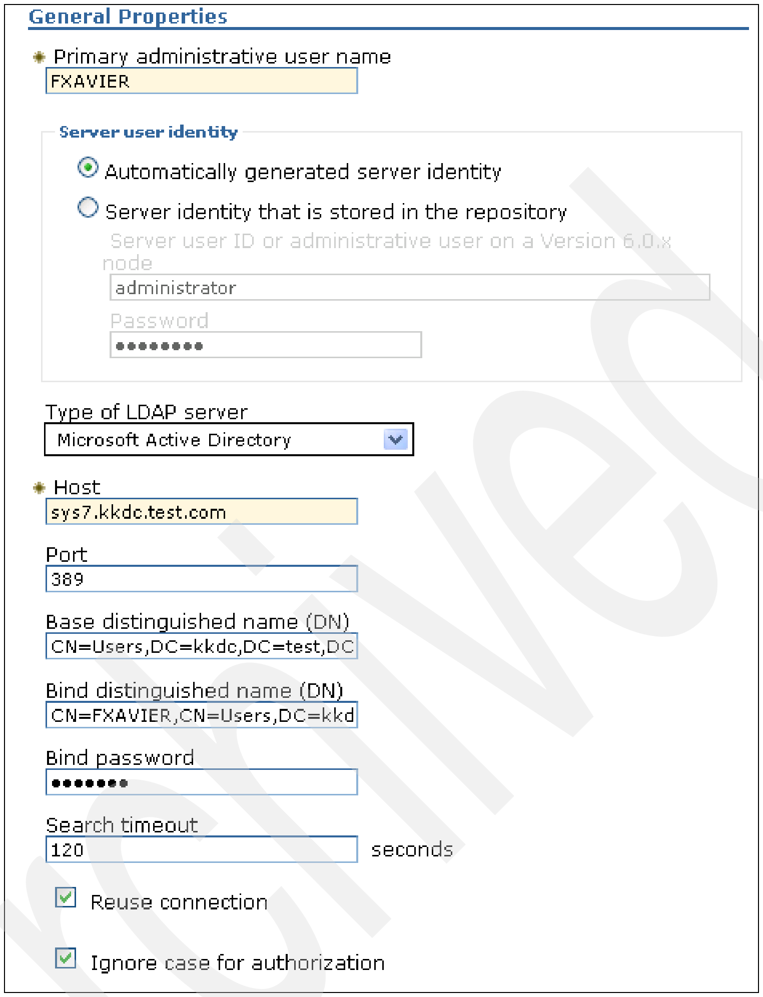
- Security domains
- External authorization providers
- Custom properties
- z/OS security options

Apply | Reset

Figure 8-2 Global security panel

In the Global security panel, complete the following steps:

1. In the Authentication mechanisms and expiration section, verify that the LTPA option is select by default.
2. In the Web and SIP security section, click **Single sign-on (SSO)**. Ensure that **Enabled** is selected and that the Domain name is set. In this scenario, the Domain name field is set to `kkdc.test.com` value. Return to the Global security panel.
3. Click **External authorization providers**. Select **Built-in authorization**, and click **OK**.
4. Select **Standalone LDAP registry** in the Available realm definitions field, and click **Configure**. Configure the Microsoft Active Directory as the user registry as shown in Figure 8-3. Then, click **OK** to return to the Global security panel.


General Properties

* Primary administrative user name
FXAVIER

Server user identity

[✓ Automatically generated server identity]
Server identity that is stored in the repository
Server user ID or administrative user on a Version 6.0.x node
administrator
Password
●●●●●●●●

Type of LDAP server
Microsoft Active Directory

* Host
sys7.kkdc.test.com

Port
389

Base distinguished name (DN)
CN=Users,DC=kkdc,DC=test,DC

Bind distinguished name (DN)
CN=FXAVIER,CN=Users,DC=kkd

Bind password
●●●●●●●

Search timeout
120 seconds

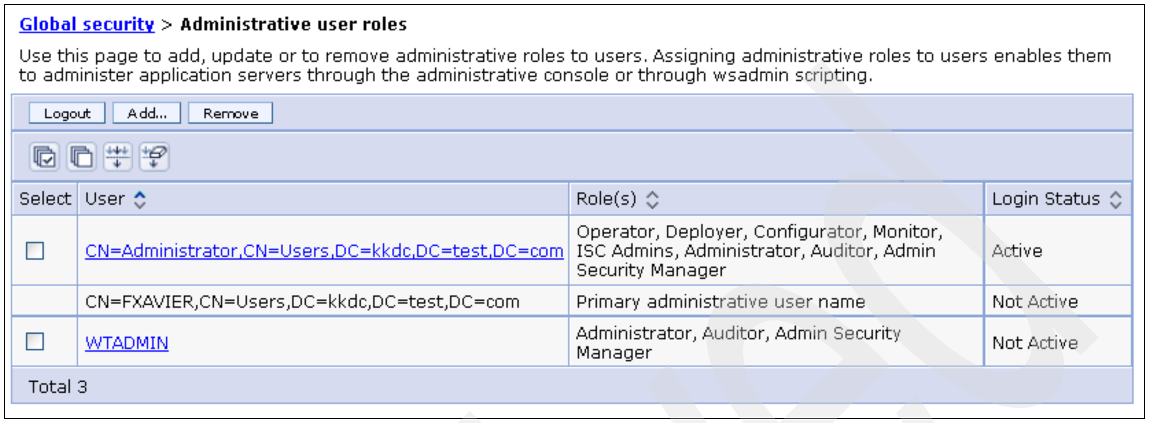
✓ Reuse connection

✓ Ignore case for authorization

Figure 8-3 Microsoft Active Directory as user registry

5. Select **Standalone LDAP registry** in the Available realm definitions field, and click **Set as current**.
6. Verify that **Enabled** is selected for both administrative and application security.
7. Click **Apply** to apply all the settings in the Global security panel.

8. In the Administrative security section, click **Administrative user roles**, and assign the Administrator role to at least one user in the Microsoft Active Directory, as shown in Figure 8-4.


Global security > **Administrative user roles**

Use this page to add, update or to remove administrative roles to users. Assigning administrative roles to users enables them to administer application servers through the administrative console or through wsadmin scripting.

Select	User	Role(s)	Login Status
☐	CN=Administrator,CN=Users,DC=kkdc,DC=test,DC=com	Operator, Deployer, Configurator, Monitor, ISC Admins, Administrator, Auditor, Admin Security Manager	Active
	CN=FXAVIER,CN=Users,DC=kkdc,DC=test,DC=com	Primary administrative user name	Not Active
☐	WTADMIN	Administrator, Auditor, Admin Security Manager	Not Active
Total 3			

Figure 8-4 Administrative user roles of Administrator user from Microsoft Active Directory

9. Save the configuration and restart WebSphere Application Server V7.

Because this example changed the user registry in the global security configuration, the next time that you log on to WebSphere, you need to use one of the IDs defined in the step 8.

8.3.3 Configuring the SPNEGO Web authentication in WebSphere

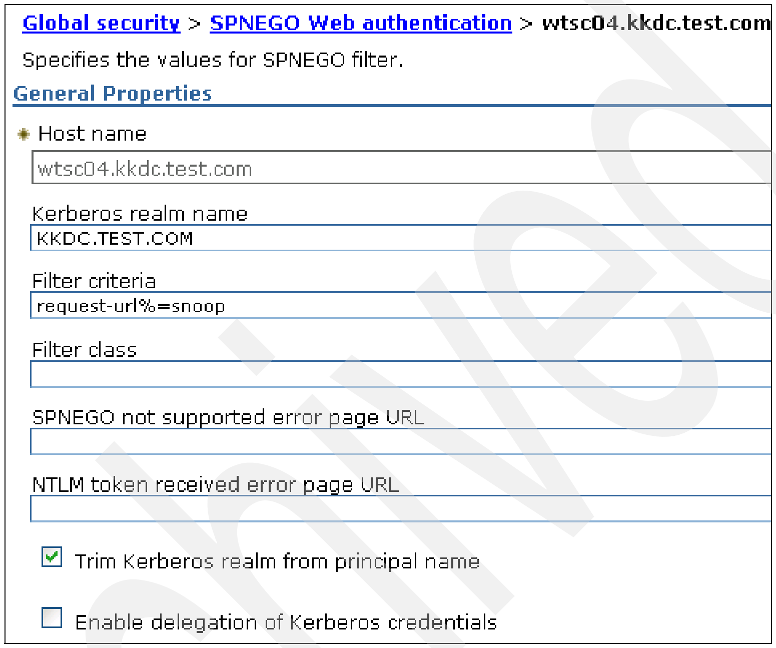
To configure the SPNEGO Web authentication in WebSphere Application Server using the administrative console, follow these steps:

1. Log on to the WebSphere Application Server administrative console and navigate to **Security** → **Global security**. The Global security panel opens as shown in Figure 8-2 on page 181. Expand **Web and SIP security**, and then click **SPNEGO Web authentication**.
2. You need to define a filter each application that participates in SSO. On the SPNEGO Filters list, click **New**. In the General Properties panel, shown in Figure 8-5 on page 184, complete the following steps:
   a. Enter the fully qualified host name of WebSphere Application Server in the Host name field and enter the realm name in the Kerberos realm name field.
   b. Enter `request-url%=snoop` in the Filter criteria field so that requests can be verified by SPNEGO Web authentication.

c. Select **Trim Kerberos realm from principal name** to remove the suffix of the principal user name.

d. Click **Apply**.

e. The realm name and filter criteria are validated, and you are returned to the SPNEGO Web authentication page.



Global security > SPNEGO Web authentication > wtsc04.kkdc.test.com

Specifies the values for SPNEGO filter.

General Properties

* Host name: wtsc04.kkdc.test.com

Kerberos realm name: KKDC.TEST.COM

Filter criteria: request-url%=snoop

Filter class:

SPNEGO not supported error page URL:

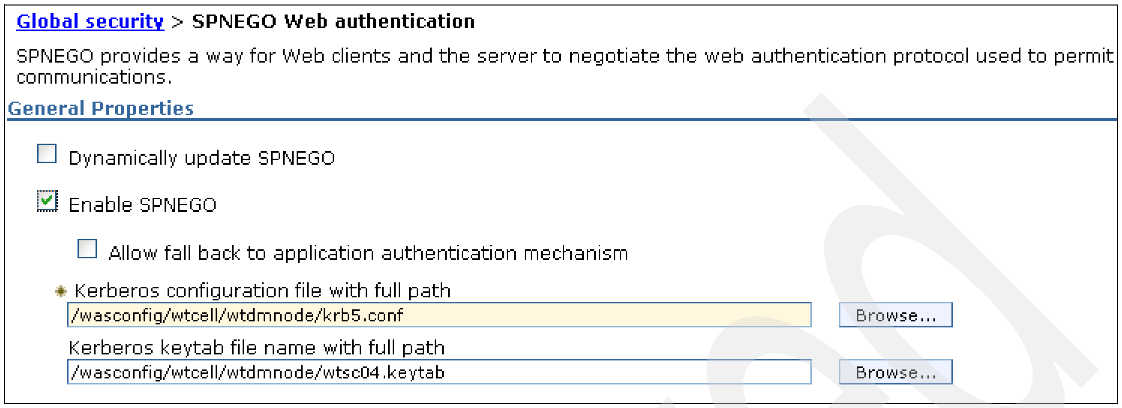
NTLM token received error page URL:

[✓] Trim Kerberos realm from principal name

[ ] Enable delegation of Kerberos credentials

Figure 8-5 SPNEGO Web authentication filter

3. Click **Enable SPNEGO**, and select the Kerberos configuration and keytab file as shown in Figure 8-6. Then, click **Apply**.


Global security > SPNEGO Web authentication

SPNEGO provides a way for Web clients and the server to negotiate the web authentication protocol used to permit communications.

General Properties

☐ Dynamically update SPNEGO

☑ Enable SPNEGO

☐ Allow fall back to application authentication mechanism

* Kerberos configuration file with full path: /wasconfig/wtcell/wtdmnode/krb5.conf

Kerberos keytab file name with full path: /wasconfig/wtcell/wtdmnode/wtsc04.keytab

Figure 8-6 Enable SPNEGO Web authentication window

Note the "Allow fall back to application authentication mechanism" option. This option specifies that SPNEGO is used to log in to WebSphere Application Server first. If SPNEGO authentication fails, then the authentication mechanism that is defined during application assembly time is used.

Click **OK**.

Note: If the "Dynamically update SPNEGO" option is enabled, then when the server recycles, any changes to the SPNEGO configuration are loaded automatically.

4. Save the configuration changes, and in a network deployment cell synchronize the changes to the nodes.
5. Recycle the server to load the SPNEGO modifications to global security.

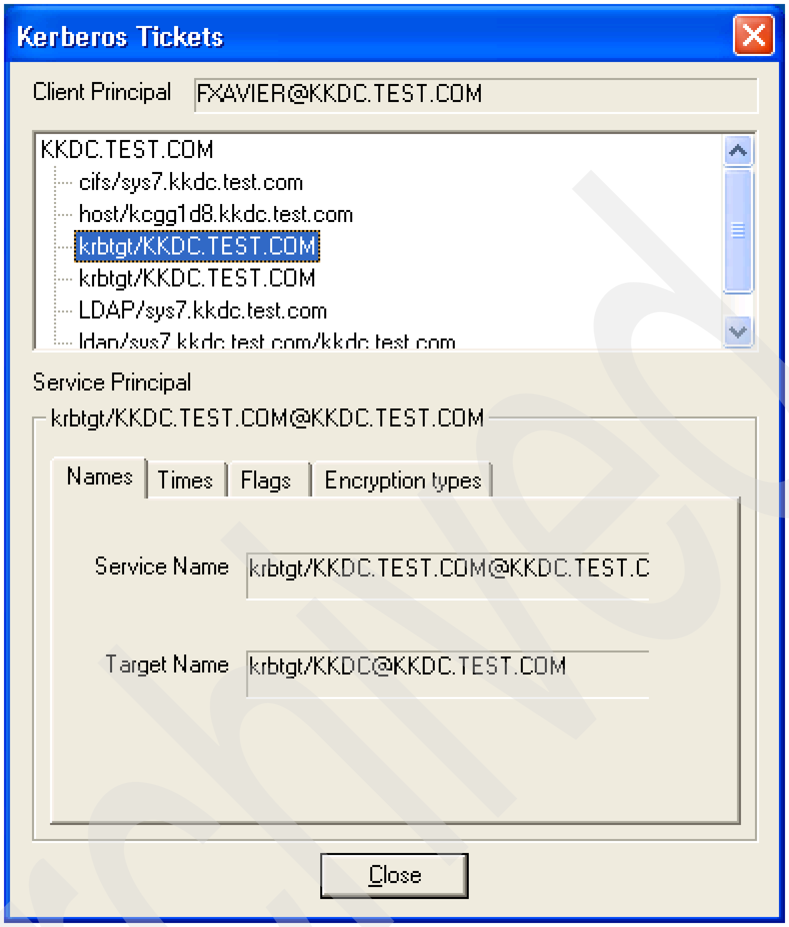
8.4 Configuring Web browsers for SPNEGO

The Web browser is a common client tool used in SSO scenarios and is used to verify the SPNEGO setup for this scenario. First, make sure that the client workstation is a member of the Microsoft domain controller. Then, configure the Web browser on the client workstation for SPNEGO using the instructions described in Appendix B, “Configuring Web browsers for SPNEGO” on page 499. This scenario can use either Microsoft Internet Explorer or Mozilla Firefox to test SPNEGO authentication with WebSphere Application Server.

8.5 Validating SSO with SPNEGO Web authentication

To validate the SSO scenario between a Windows workstation and the application server, follow these steps:

1. In the user workstation, log on to the Windows Active Directory domain. The user ID used in the example is `FXAVIER`.
2. Use the **`kerbtray.exe`** utility to see the Kerberos tickets that are acquired by the user. The **`kerbtray.exe`** utility is available from the Windows Server 2003 Resource Kit Tools. Immediately after logon, you can verify that the workstation obtained the TGT from the Windows KDC. This TGT is named `krbtgt/KKDC.TEST.COM` in our example, as shown in Figure 8-7.


Kerberos Tickets

Client Principal: FXAVIER@KKDC.TEST.COM

```
KKDC.TEST.COM
  cifs/sys7.kkdc.test.com
  host/kcgg1d8.kkdc.test.com
  krbtgt/KKDC.TEST.COM
  krbtgt/KKDC.TEST.COM
  LDAP/sys7.kkdc.test.com
  ldap/sys7.kkdc.test.com/kkdc.test.com
```

Service Principal: krbtgt/KKDC.TEST.COM@KKDC.TEST.COM

Service Name: krbtgt/KKDC.TEST.COM@KKDC.TEST.C

Target Name: krbtgt/KKDC@KKDC.TEST.COM

Figure 8-7 User Kerberos tickets after logon

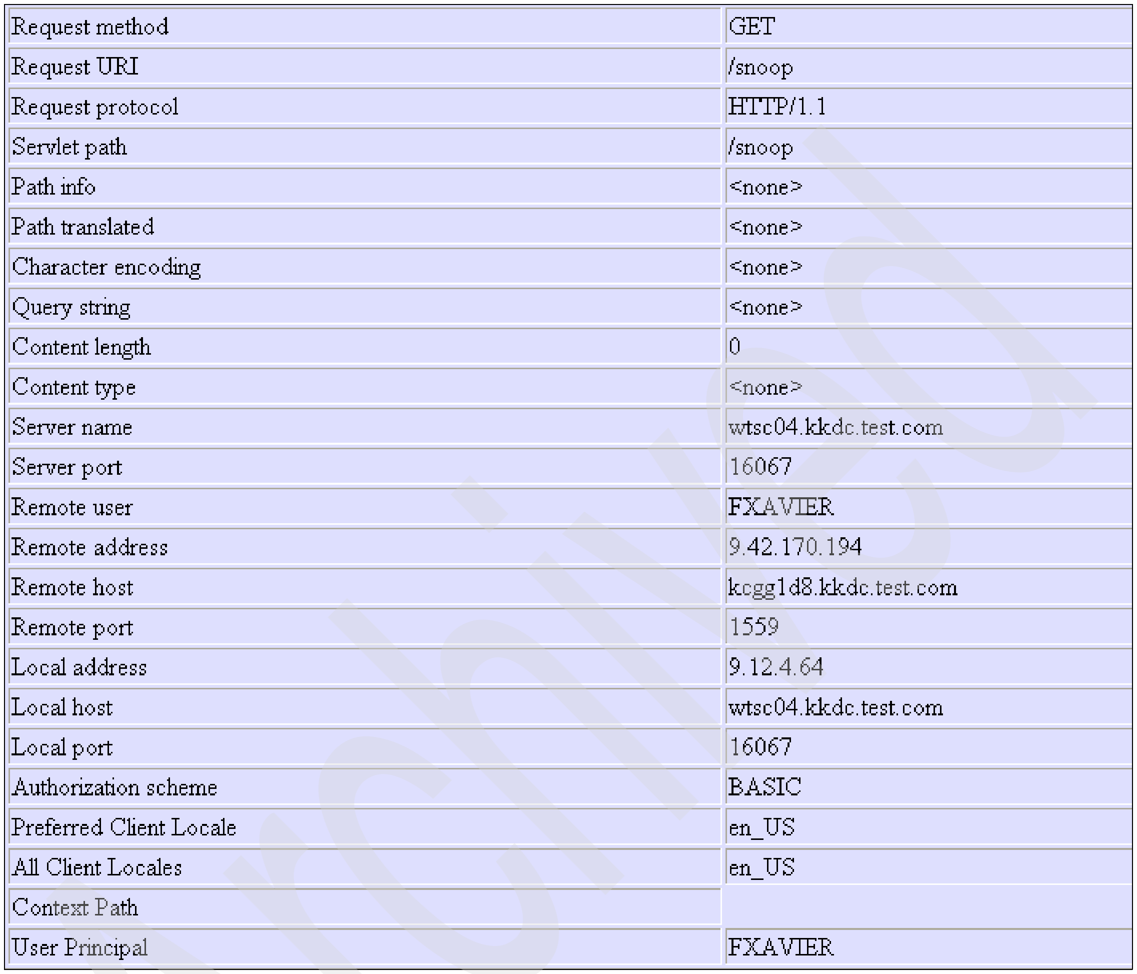
3. Launch a Web browser and enter the URL of the application that you want to access using SSO. In this example, we use the snoop servlet in the default application:

   ```
   http://wtsc04.kkdc.test.com:16067/snoop/
   ```

   When application security is enabled, the snoop servlet asks for authentication.

Because SPNEGO is configured, the browser asks the KDC for a service ticket to use the HTTP service with `wtsc04.kkdc.test.com`. The browser then sends the HTTP request, including a SPNEGO token with the user identity (`FXAVIER`) to the application server. (These steps are described in Figure 8-1 on page 174.)

Figure 8-8 shows the output of the snoop servlet.



Request method	GET
Request URI	/snoop
Request protocol	HTTP/1.1
Servlet path	/snoop
Path info	<none>
Path translated	<none>
Character encoding	<none>
Query string	<none>
Content length	0
Content type	<none>
Server name	wtsc04.kkdc.test.com
Server port	16067
Remote user	FXAVIER
Remote address	9.42.170.194
Remote host	kcgg1d8.kkdc.test.com
Remote port	1559
Local address	9.12.4.64
Local host	wtsc04.kkdc.test.com
Local port	16067
Authorization scheme	BASIC
Preferred Client Locale	en_US
All Client Locales	en_US
Context Path	
User Principal	FXAVIER

Figure 8-8 Internet Explorer Snoop servlet display

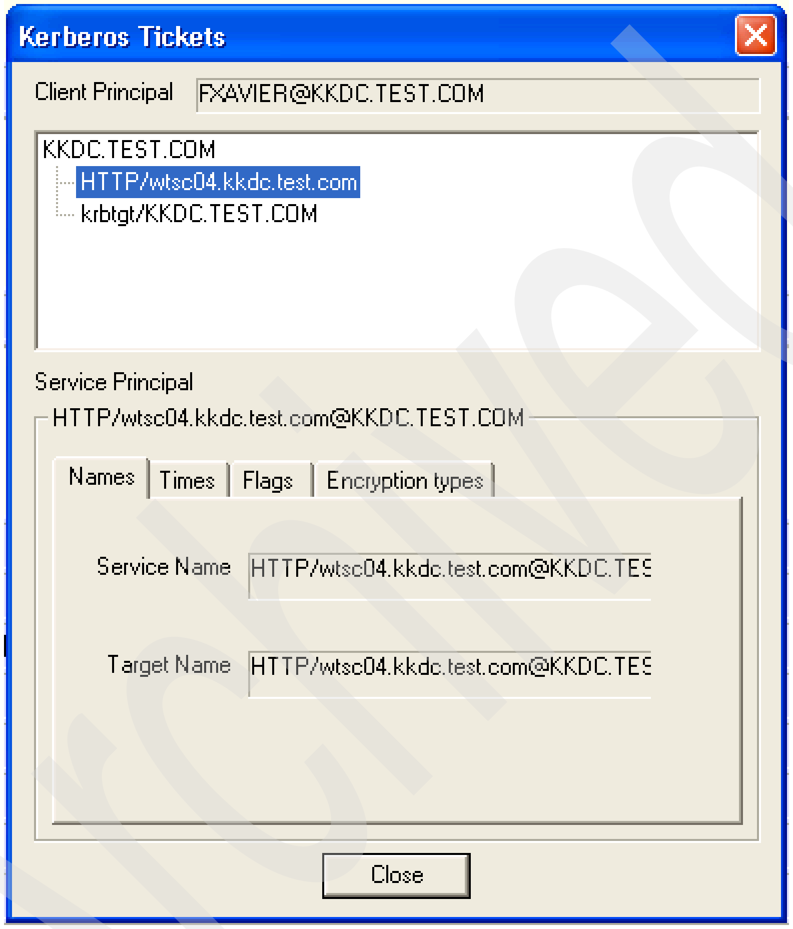
The output shows the authenticated user (`FXAVIER`) from the Windows domain, indicating that SSO occurred between the Windows domain and WebSphere Application Server using the SPNEGO.

The WebSphere Application Server V7 for z/OS servant region log also confirms this scenario, as shown in Example 8-3. The log shows that the SPNEGO token is processed and highlights the authenticated user.

Example 8-3 WebSphere Application Server z/OS servant region log with traces on

```
Trace: 2009/03/05 18:17:35.428 01 t=7C15D0 c=UNK key=P8 (13007002)
  ThreadId: 0000001c
  FunctionName: com.ibm.ws.security.spnego.SpnegoHandler.handleRequest
  SourceId: com.ibm.ws.security.spnego.SpnegoHandler
  Category: FINER
  ExtendedMessage: SPNEGO request token successfully processed.
Trace: 2009/03/05 18:17:35.428 01 t=7C15D0 c=UNK key=P8 (13007002)
  ThreadId: 0000001c
  FunctionName: com.ibm.ws.security.spnego.Context.getPrincipalName
  SourceId: com.ibm.ws.security.spnego.Context
  Category: FINER
  ExtendedMessage: ENTRY
Trace: 2009/03/05 18:17:35.428 01 t=7C15D0 c=UNK key=P8 (13007002)
  ThreadId: 0000001c
  FunctionName: com.ibm.ws.security.spnego.Context.getPrincipalName
  SourceId: com.ibm.ws.security.spnego.Context
  Category: FINER
  ExtendedMessage: RETURN FXAVIER@KKDC.TEST.COM
Trace: 2009/03/05 18:17:35.429 01 t=7C15D0 c=UNK key=P8 (13007002)
  ThreadId: 0000001c
  FunctionName: com.ibm.ws.security.spnego.SpnegoHandler.handleRequest
  SourceId: com.ibm.ws.security.spnego.SpnegoHandler
  Category: FINER
   ExtendedMessage: CWSPN0023I: User name FXAVIER@KKDC.TEST.COM Token
size 1,630
 Trace: 2009/03/05 18:17:35.429 01 t=7C15D0 c=UNK key=P8 (13007002)
   ThreadId: 0000001c
 FunctionName: com.ibm.ws.security.spnego.SpnegoHandler.handleRequest
  SourceId: com.ibm.ws.security.spnego.SpnegoHandler
   Category: FINER
  ExtendedMessage: Kerberos client principal: FXAVIER@KKDC.TEST.COM
 Trace: 2009/03/05 18:17:35.429 01 t=7C15D0 c=UNK key=P8 (13007002)
   ThreadId: 0000001c
 FunctionName: com.ibm.ws.security.spnego.SpnegoHandler.handleRequest
  SourceId: com.ibm.ws.security.spnego.SpnegoHandler
   Category: FINER
    ExtendedMessage: handleRequest: Sending back SPNEGO response token
```

Using the **kerbtray.exe** utility again, you can see the Kerberos tickets that are acquired by the user for the request (Figure 8-9). In this example, the utility shows that the user obtained a service ticket to access the WebSphere for z/OS Kerberized service (`HTTP/wtsc04.kkdc.test.com`).



Client Principal FXAVIER@KKDC.TEST.COM

KKDC.TEST.COM
HTTP/wtsc04.kkdc.test.com
krbtgt/KKDC.TEST.COM

Service Principal
HTTP/wtsc04.kkdc.test.com@KKDC.TEST.COM

Service Name HTTP/wtsc04.kkdc.test.com@KKDC.TES

Target Name HTTP/wtsc04.kkdc.test.com@KKDC.TES

Figure 8-9 User Kerberos tickets after HTTP request

These steps validate the scenario described in Figure 8-1 on page 174 and confirm the SSO between the Windows domain and WebSphere Application Server V7.

9

Single sign-on using SPNEGO in a trusted Microsoft Kerberos KDC environment

This chapter explains single sign-on (SSO) using Simple and Protected GSSAPI Negotiation Mechanism (SPNEGO) Web authentication with WebSphere Application Server V7. In this scenario, the client and server are in two different Kerberos realms. Trust is established between the two Kerberos realms, enabling SSO to occur. The KDCs are implemented with Microsoft Active Directory.

This chapter includes the following topics:

- Scenario overview
- Creating the service principal name and keytab file
- Configuring WebSphere Application Server
- Validating the scenario

9.1 Scenario overview

In a Microsoft Active Directory solution, it is common to have a collection of domains throughout an organization. You can create a *trust relationship* between these domains, which enables clients who are authenticated in other, trusted domains to access the resources that are hosted in one domain (such as applications, printers, and so forth).

As an example, consider a company that has two Microsoft Active Directory administrative domains (`kdc.itso.com` and `kkdc.test.com`) that have a trust relationship established. Clients registered and authenticated in the `kdc.itso.com` domain can access an application running in a WebSphere Application Server V7 that is hosted in the `kkdc.test.com` domain.

In addition, you can configure WebSphere Application Server V7 to combine multiple domains into a federated repository for use as its user registry. A federated repository in WebSphere can consist of multiple file-based repositories, LDAP repositories, or even a sub-tree of an LDAP repository. These repositories are combined virtually into a single domain.

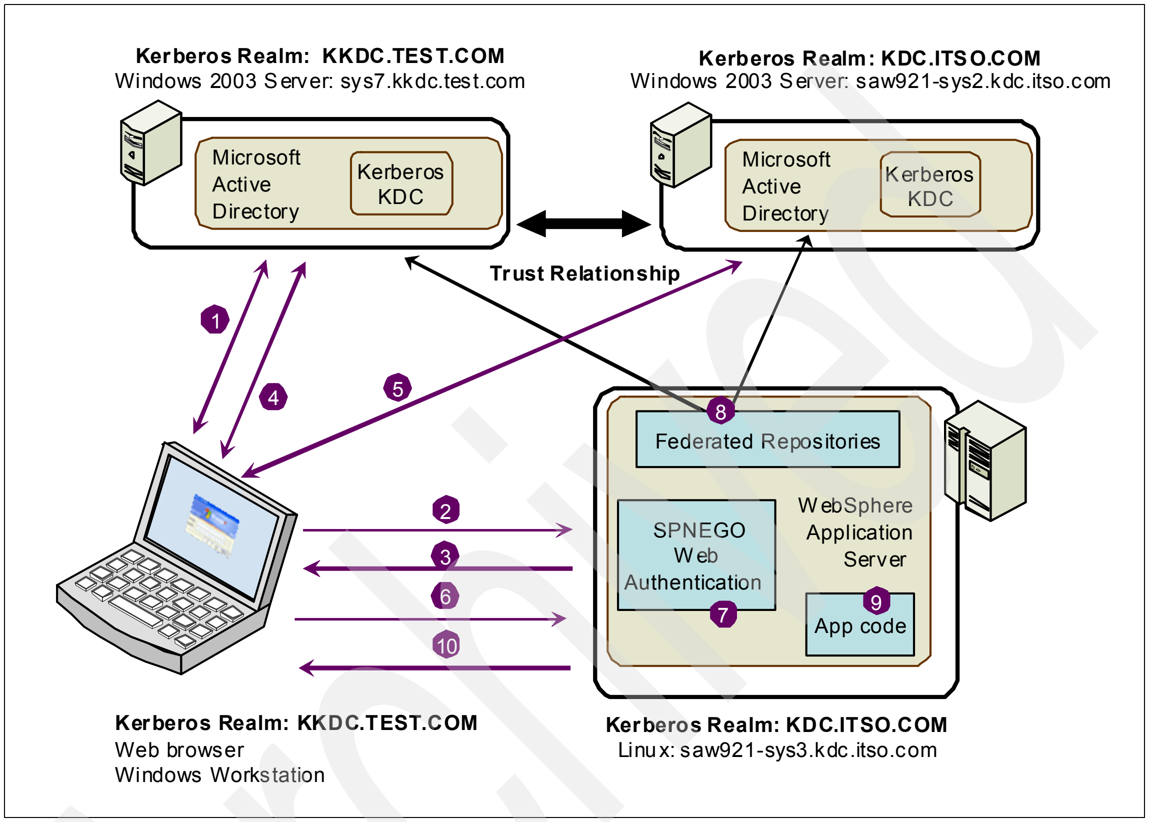
The scenario in this chapter uses two different domains that are federated into a single repository, which allows WebSphere Application Server to access principals that are created on both domains.

Note: This scenario features the following components:

- SSO from a Web browser client in one domain to WebSphere Application Server in another domain
- Two Microsoft Kerberos key distribution centers (KDCs) with trust established between them
- A Web browser client
- The following WebSphere Application Server configuration:
  - User registry: Federated repositories, including two Microsoft Active Directory installations
  - Authentication mechanism: LTPA
  - SPNEGO Web authentication
  - Kerberos configuration and keytab files
  - Sample application: DefaultApplication (snoop servlet)

9.1.1 Step-by-step flow of the scenario

Figure 9-1 illustrates the basic flow of this scenario.



Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: sys7.kkdc.test.com
Microsoft Active Directory
Kerberos KDC
Kerberos Realm: KDC.ITSO.COM
Windows 2003 Server: saw921-sys2.kdc.itso.com
Microsoft Active Directory
Kerberos KDC
Trust Relationship
1
4
5
8
Federated Repositories
2
3
6
10
SPNEGO Web Authentication
7
WebSphere Application Server
9
App code
Kerberos Realm: KKDC.TEST.COM
Web browser
Windows Workstation
Kerberos Realm: KDC.ITSO.COM
Linux: saw921-sys3.kdc.itso.com

Figure 9-1 WebSphere Application Server in a trusted Active Directory domain

A request for a Web application in this scenario takes the following steps:

1. The user logs in to the `kkdc.test.com` domain.
2. From a browser, the user makes a request for a secured Web application that is hosted in a WebSphere Application Server V7 in the second domain, `KDC.itso.com`.
3. WebSphere Application Server SPNEGO answers the client browser with an HTTP challenge header 401 that contains the `Authenticate: Negotiate` status.
4. The browser is configured to support Integrated Windows Authentication. The browser parses the URL using the host name of the workstation that hosts the WebSphere application. The browser uses the host name as an attribute to

request a cross-realm ticket (TGS_REQ) for `KDC.ITSO.COM` from realm `KKDC.TEST.COM`.

5. The client uses the cross-realm ticket from step 4 to request a Service Ticket from realm `KDC.ITSO.COM`.
6. The client puts this Kerberos service ticket (TGS_REP) in a SPNEGO token. The service ticket proves the user's identity and permissions to the service as well as the service's identity to the user. The client responds to the WebSphere Application Server `Authenticate: Negotiate` challenge with the SPNEGO token in the request HTTP header.
7. WebSphere Application Server receives the request and checks the HTTP header with the SPNEGO token. It then extracts the Kerberos service ticket and gets the identity (principal) of the user.
8. WebSphere Application Server verifies that the identity available in the SPNEGO token matches a valid account that belongs to the Kerberos realms that are defined to the federated repositories.
9. After the identification process is executed, WebSphere Application Server executes the related Java code (servlets, JSPs, EJBs, and so on) and checks authorizations.
10. WebSphere Application Server sends the response with an HTTP 200. WebSphere Application Server also includes in the response an LTPA token in the form of a cookie. This LTPA cookie is used for the next requests.

9.1.2 Scenario components

The scenario that we use in this chapter focuses on WebSphere Application Server V7 architecture that uses federated repositories with two Microsoft Active Directory servers. The components for this scenario are as follows:

- The first KDC (host name `saw921-sys2.kdc.itso.com`)

  Microsoft Windows Server 2003 SP2 with a Microsoft Active Directory domain, which is responsible for the `KDC.ITSO.COM` Kerberos realm.

- The second KDC (host name `sys7.kkdc.test.com`)

  Microsoft Windows Server 2003 SP2 with a Microsoft Active Directory domain, which is responsible for the `KKDC.TEST.COM` Kerberos realm.

- The application server

  WebSphere Application Server for Windows V7.0.0.5, Build Level cf050925.25, which is installed on a Linux Server (host name `saw921-sys3.kdc.itso.com`)

- The user workstation

  Microsoft Windows XP Professional 2002 SP2, including the Windows Server 2003 Resource Kit Tools installed. The workstation has Microsoft Internet Explorer V6.0.2900 and Mozilla Firefox V3.0.6. In addition, this workstation is a member of the `kkdc.test.com` domain.

9.1.3 Basic requirements for this scenario

The following servers and configurations are required for the scenario that we use in this chapter:

- Two Microsoft Active Directory domain controllers:
  - One domain controller for the KDC.ITSO.COM Kerberos realm
  - One domain controller for the KKDC.TEST.COM Kerberos realm

  You can find the steps to create the relationship in 4.3, "Creating the Microsoft Kerberos KDC" on page 58.
- At least one client computer that is a member of the `kkdc.test.com` domain
- A trust relationship between the Kerberos realms that is enabled

  You can find the steps to create the relationship in 4.4, "Establishing a trust relationship between Microsoft Active Directory realms" on page 69.

9.1.4 Summary of implementation steps

To implement SSO for a WebSphere environment with Microsoft Kerberos trusted realms requires the following high-level steps:

1. Set up two Windows servers, each with a Microsoft Kerberos KDC and a Kerberos realm.
2. Establish the trust relationship between the Kerberos realms.
3. Configure the client (browser) to support the SPNEGO Web authentication mechanism.
4. Install WebSphere Application Server in the Linux system, `saw921-sys3.kdc.itso.com`, and configure the system to use federated repositories with both of the Kerberos realms (`KKDC.TEST.COM` and `KDC.ITSO.COM`) as registries in the federation.
5. Configure WebSphere Application Server to use SPNEGO Web authentication.
6. Validate the solution that is making a request to a WebSphere application through a Web browser.

9.2 Creating the service principal name and keytab file

In this section, we explain how to create the Kerberos and SPNEGO service principal name (SPN) for WebSphere Application Server and how to create the keytab file for use on the WebSphere Application Server system.

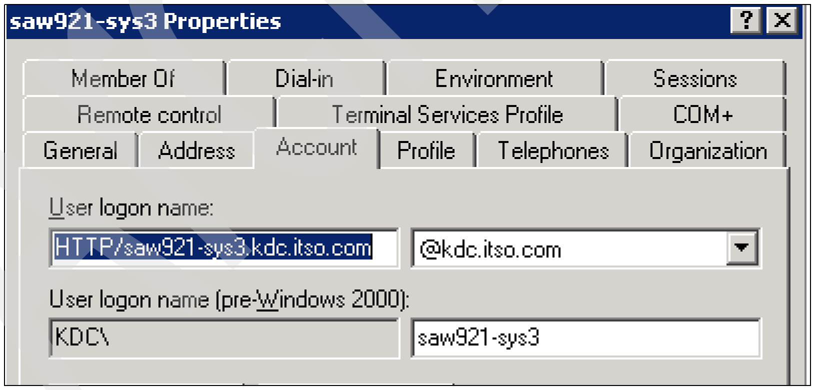
9.2.1 Creating the SPN for SPNEGO and Kerberos

First, you need to create the Kerberos and SPNEGO SPN for WebSphere Application Server in the KDC realm to which it belongs (`KDC.ITSO.COM`). For the SPN, use the host name of the server where WebSphere Application Server is running, which in this example is `saw921-sys3.kdc.itso.com`. To do this, go to the Active Directory Users and Computers console on the KDC system (`saw921-sys2.kdc.itso.com`), and create a new user for SPNEGO with the properties listed in Table 9-1.

Table 9-1 SPNEGO user ID in the kdc.itso.com Kerberos realm

Property	Value
User logon name	`HTTP/saw921-sys3.kdc.itso.com@kdc.itso.com`
User logon name (pre-Windows 2000)	`KDC\saw921-sys3`

Figure 9-2 shows the SPNEGO user properties.



User logon name:
HTTP/saw921-sys3.kdc.itso.com @kdc.itso.com

User logon name (pre-Windows 2000):
KDC\ saw921-sys3

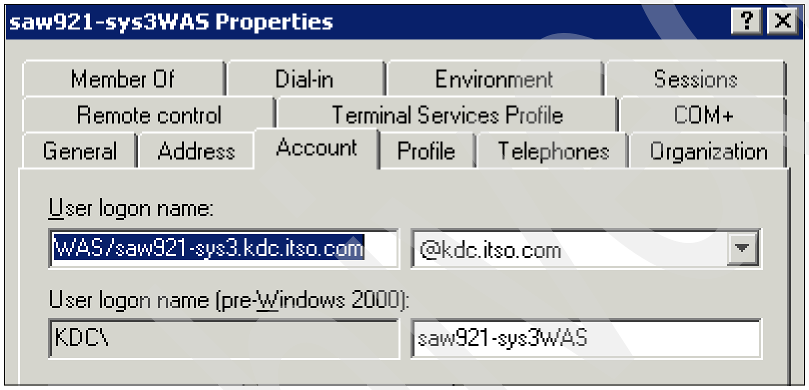
Figure 9-2 SPNEGO user properties

Next, create a new user for Kerberos with the properties listed in Table 9-2.

Table 9-2 Kerberos user ID in the kdc.itso.com Kerberos realm

Property	Value
User logon name	`WAS/saw921-sys3.kdc.itso.com@kdc.itso.com`
User logon name (pre-Windows 2000)	`KDC\saw921-sys3WAS`

Figure 9-3 shows the Kerberos user properties.



User logon name:
WAS/saw921-sys3.kdc.itso.com @kdc.itso.com

User logon name (pre-Windows 2000):
KDC\ saw921-sys3WAS

Figure 9-3 Kerberos user properties

9.2.2 Creating the keytab files in the Kerberos realm

Now, you can use the **ktpass** command to create a keytab file for WebSphere Application Server in the `KDC.ITSO.COM` Kerberos realm. To create the keytab file for SPNEGO, execute the command in the KDC system, `saw921-sys2.kdc.itso.com`, as shown in Example 9-1.

Example 9-1 The ktpass command output for SPNEGO keytab

```
C:\>ktpass -princ HTTP/saw921-sys3.kdc.itso.com@KDC.ITSO.COM -mapuser
saw921-sys3 -pass <password> -out c:\saw921-sys3.keytab
Targeting domain controller: saw921-sys2.kdc.itso.com
Using legacy password setting method
Successfully mapped HTTP/saw921-sys3.kdc.itso.com to saw921-sys3.
WARNING: pType and account type do not match. This might cause
problems.
Key created.
Output keytab to c:\saw921-sys3.keytab:
Keytab version: 0x502
```

```
keysize 77 HTTP/saw921-sys3.kdc.itso.com@KDC.ITSO.COM ptype 0
(KRB5_NT_UNKNOWN)
vno 3 etype 0x17 (RC4-HMAC) keylength 16
(0x31a1f13010baa0c4f9fc8ead6de09db5)
```

To create the keytab for Kerberos, execute the command in the KDC system, `saw921-sys2.kdc.itso.com`, as shown in Example 9-2.

Example 9-2 The ktpass command output for Kerberos keytab

```
C:\>ktpass -princ WAS/saw921-sys3.kdc.itso.com@KDC.ITSO.COM -mapuser
saw921-sys3WAS -pass <password> -out c:\saw921-sys3WAS.keytab
Targeting domain controller: saw921-sys2.kdc.itso.com
Using legacy password setting method
Successfully mapped WAS/saw921-sys3.kdc.itso.com to saw921-sys3.
WARNING: pType and account type do not match. This might cause
problems.
Key created.
Output keytab to c:\saw921-sys3WAS.keytab:
Keytab version: 0x502
keysize 77 WAS/saw921-sys3.kdc.itso.com@KDC.ITSO.COM ptype 0
(KRB5_NT_UNKNOWN)
vno 3 etype 0x17 (RC4-HMAC) keylength 16
(0x31a1f13010baa0c4f9fc8ead6de09db5)
```

9.2.3 Moving the keytab files to the WebSphere Application Server system

Securely transfer the keytab files, `c:\saw921-sys3.keytab` and `c:\saw921-sys3WAS.keytab`, from the KDC system to the WebSphere Application Server system. Put the files in the `/etc` directory. Then, use the **ktab** tool to merge the keytab files. Execute **ktab -m** as shown in Example 9-3 to merge the service principals into the `saw921-sys3.keytab` file.

Example 9-3 Merging the keytab files

```
[root@saw921-sys3 etc]# /opt/WebSphere/AppServer/java/jre/bin/ktab -m
saw921-sys3WAS.keytab saw921-sys3.keytab
Merging keytab files: source=saw921-sys3WAS.keytab
destination=saw921-sys3.keytab
Done !
[root@saw921-sys3 etc]#
```

You can use the **ktab** tool to list the Kerberos SPN in the keytab file. Execute **ktab -l** to verify that the files were copied successfully, as shown in Example 9-4.

Example 9-4 List of entries in the keytab file

```
[root@saw921-sys3 etc]# /opt/WebSphere/AppServer/java/jre/bin/ktab -l
-k /etc/saw921-sys3.keytab
2 entries in keytab, name: /etc/saw921-sys3.keytab
        KVNO    Principal
        ----    ---------
        3       HTTP/saw921-sys3.kdc.itso.com@KDC.ITSO.COM
        3       WAS/saw921-sys3.kdc.itso.com@KDC.ITSO.COM
You have new mail in /var/spool/mail/root
[root@saw921-sys3 etc]#
```

Note: A Kerberos keytab configuration file contains a list of keys that are analogous to user passwords. It is important for hosts to protect their Kerberos keytab files by storing them on the local disk, which makes them readable only by authorized users.

9.3 Configuring WebSphere Application Server

In this section, we explain the steps needed to configure WebSphere Application Server V7 for SSO with trusted Kerberos realms.

9.3.1 Configuring the federated repositories

Note: Although WebSphere Application Server V7 supports multiple security domains, Kerberos configuration is possible only at the cell level. Thus, all WebSphere Application Server servers must use the same Kerberos realm.

To configure the WebSphere Application Server federated repositories to include both the `kdc.itso.com` and `kkdc.test.com` domains, follow these steps:

1. Access the WebSphere administrative console and select **Security** → **Global Security**.
2. From the drop-down menu for Available realm definitions, select **Federated repositories**, and then click **Configure**.

3. In the Federated Repository window, select **Add Base entry to Realm**.
4. In the Repository reference window, under General properties, click **Add Repository** to add a new LDAP repository.
5. Add the first Microsoft Active Directory domain and server configuration, `kdc.itso.com`, with the following properties, as shown in Figure 9-4:
   a. Enter the name of the first realm for the repository identifier:

      ```
      KDC.ITSO.COM
      ```

   b. Select **Microsoft Windows Active Directory** as the directory type.

   c. Provide the host name where the KDC is running:

      ```
      saw921-sys2.kdc.itso.com
      ```

   d. Enter the port number. In this scenario, the port is `389` because the LDAP connection is not SSL enabled.

   e. Provide a bind distinguished name in the Security section for the ID:

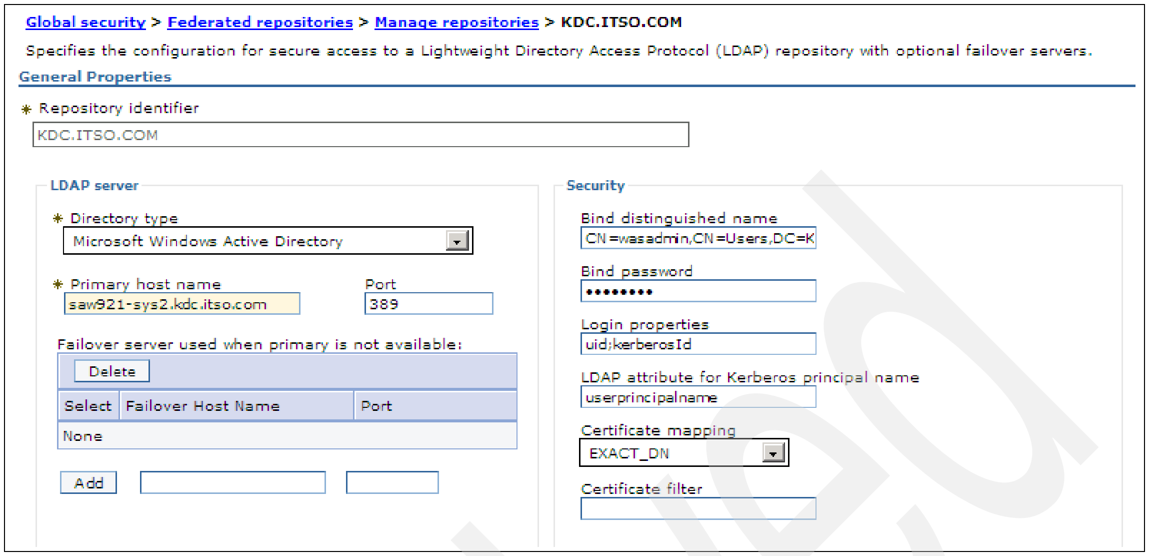
      ```
      CN=wasadmin,CN=Users,DC=KDC,DC=ITSO,DC=COM
      ```

      WebSphere uses this name when binding to the LDAP repository.

   f. Enter the password for the application server to use when binding to the LDAP repository.

   g. Click **OK.**

Note: The “Login properties” and “LDAP attribute” fields are set automatically when Kerberos is enabled (as described in 9.3.4, “Configuring Kerberos” on page 210). After Kerberos is enabled, the “Login properties” field is set to `uid;kerberosId` and the “LDAP attribute” field is set to `userprincipalname`.


Global security > Federated repositories > Manage repositories > KDC.ITSO.COM

Specifies the configuration for secure access to a Lightweight Directory Access Protocol (LDAP) repository with optional failover servers.

General Properties

* Repository identifier: KDC.ITSO.COM

LDAP server

* Directory type: Microsoft Windows Active Directory

* Primary host name: saw921-sys2.kdc.itso.com

Port: 389

Failover server used when primary is not available:

Select	Failover Host Name	Port
None		

Security

Bind distinguished name: CN=wasadmin,CN=Users,DC=K

Bind password: ••••••••

Login properties: uid;kerberosId

LDAP attribute for Kerberos principal name: userprincipalname

Certificate mapping: EXACT_DN

Certificate filter:

Figure 9-4 Configuring LDAP properties

6. In the Federated repositories General Properties page (shown in Figure 9-5), complete the following fields:

   a. Enter the distinguished name of a base entry that uniquely identifies this set of entries in the realm:

      ```
      CN=Users,DC=KDC,DC=ITSO,DC=COM
      ```

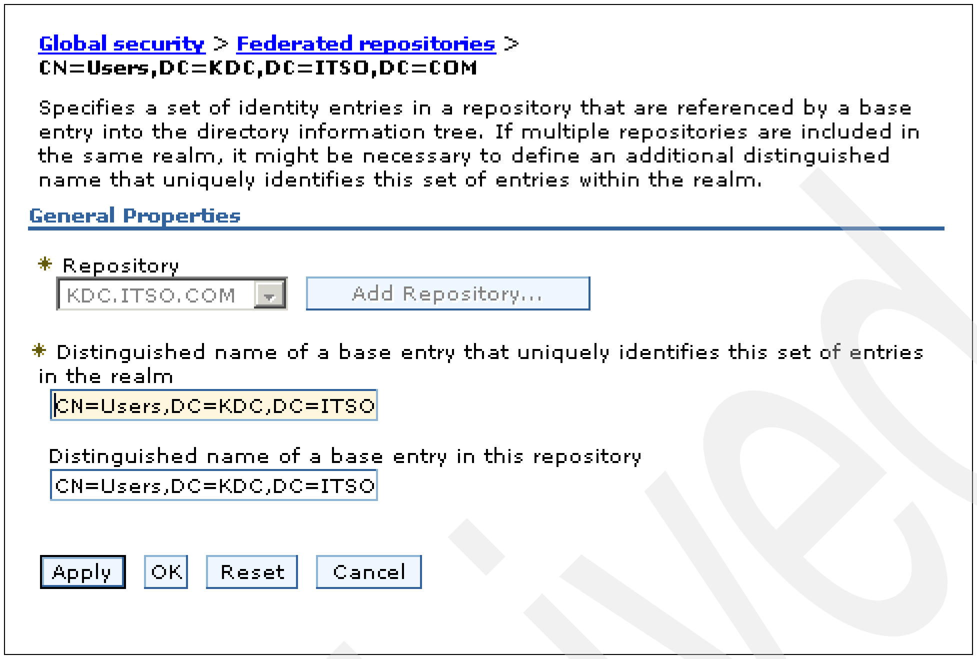
      If multiple repositories are included in the realm, define an additional distinguished name that uniquely identifies this set of entries within the realm. Overlapping base entries are not supported. Do not define two base entries where one is `c=us` and the other is `o=myorg,c=us` in the same realm. Otherwise, a search returns duplicate results.

      Because we use multiple repositories in this scenario, we define an additional distinguished name that uniquely identifies this set of entries within the realm.

   b. Enter a distinguished name of a base entry in this repository:

      ```
      CN=Users,DC=KDC,DC=ITSO,DC=COM
      ```

   c. Click **OK**. The first realm is added to federated repositories configuration.


Global security > Federated repositories >
CN=Users,DC=KDC,DC=ITSO,DC=COM

Specifies a set of identity entries in a repository that are referenced by a base entry into the directory information tree. If multiple repositories are included in the same realm, it might be necessary to define an additional distinguished name that uniquely identifies this set of entries within the realm.

General Properties

* Repository
KDC.ITSO.COM

* Distinguished name of a base entry that uniquely identifies this set of entries in the realm
CN=Users,DC=KDC,DC=ITSO

Distinguished name of a base entry in this repository
CN=Users,DC=KDC,DC=ITSO

Figure 9-5 Configuring Repository Reference

7. Now, to add the second realm, select **Global security** → **Federated repositories**. Then, click **Add Base entry to Realm**, and repeat steps 4 through 6 to add the second KDC realm (`kkdc.test.com`).

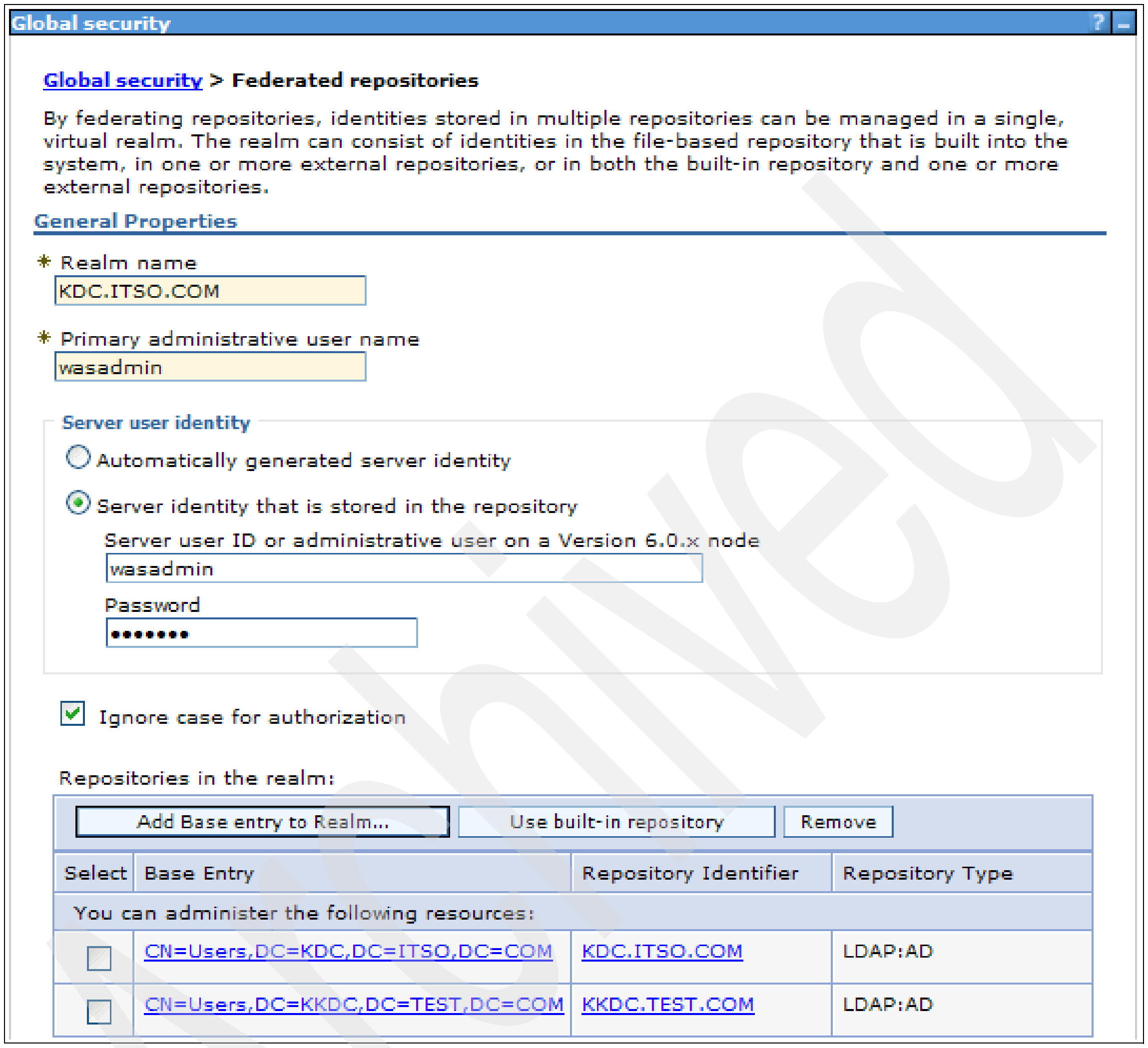
   Table 9-3 lists parameters that require values and the values that we entered for this scenario.

Table 9-3 Parameter values

Property	**Value**
Repository identifier	`KKDC.TEST.COM`
Directory Type	Microsoft Windows Active Directory
hostname	`sys7.kkdc.test.com`
port	`389`

Property	Value
Bind distinguished name	`CN=wasadmin1,CN=Users,DC=KKDC,DC=TEST,DC=COM`
password	Provide the user password
Distinguished name of a base entry that uniquely identifies this set of entries in the realm	`CN=Users,DC=KKDC,DC=TEST,DC=COM`
Distinguished name of a base entry in this repository	`CN=Users,DC=KKDC,DC=TEST,DC=COM`

8. Now that both KDC realms are added to the federated repositories, change the “Realm name” property to the first realm name, `KDC.ITSO.COM`. Also, set the Primary administrative user name and Server user identity fields, as shown in Figure 9-6. Click **OK**.



Global security

Global security > Federated repositories

By federating repositories, identities stored in multiple repositories can be managed in a single, virtual realm. The realm can consist of identities in the file-based repository that is built into the system, in one or more external repositories, or in both the built-in repository and one or more external repositories.

General Properties

* Realm name: KDC.ITSO.COM

* Primary administrative user name: wasadmin

Server user identity

○ Automatically generated server identity

● Server identity that is stored in the repository

Server user ID or administrative user on a Version 6.0.x node: wasadmin

Password: ••••••••

✓ Ignore case for authorization

Repositories in the realm:

Select	Base Entry	Repository Identifier	Repository Type
You can administer the following resources:			
☐	CN=Users,DC=KDC,DC=ITSO,DC=COM	KDC.ITSO.COM	LDAP:AD
☐	CN=Users,DC=KKDC,DC=TEST,DC=COM	KKDC.TEST.COM	LDAP:AD

Figure 9-6 Configuring KDC realms into federated repositories

9. To create the federated repository realm name for the second KDC, `KKDC.TEST.COM`, use the **`wsadmin`** command tool. Use the **`createIdMgrRealm`** and **`addIdMgrRealmBaseEntry`** commands as shown in Example 9-6. For the `name` parameter, specify the second KDC realm. For the `baseEntry` parameter, specify the base entry distinguished name for the second KDC.

Example 9-5 Adding the second realm using wsadmin

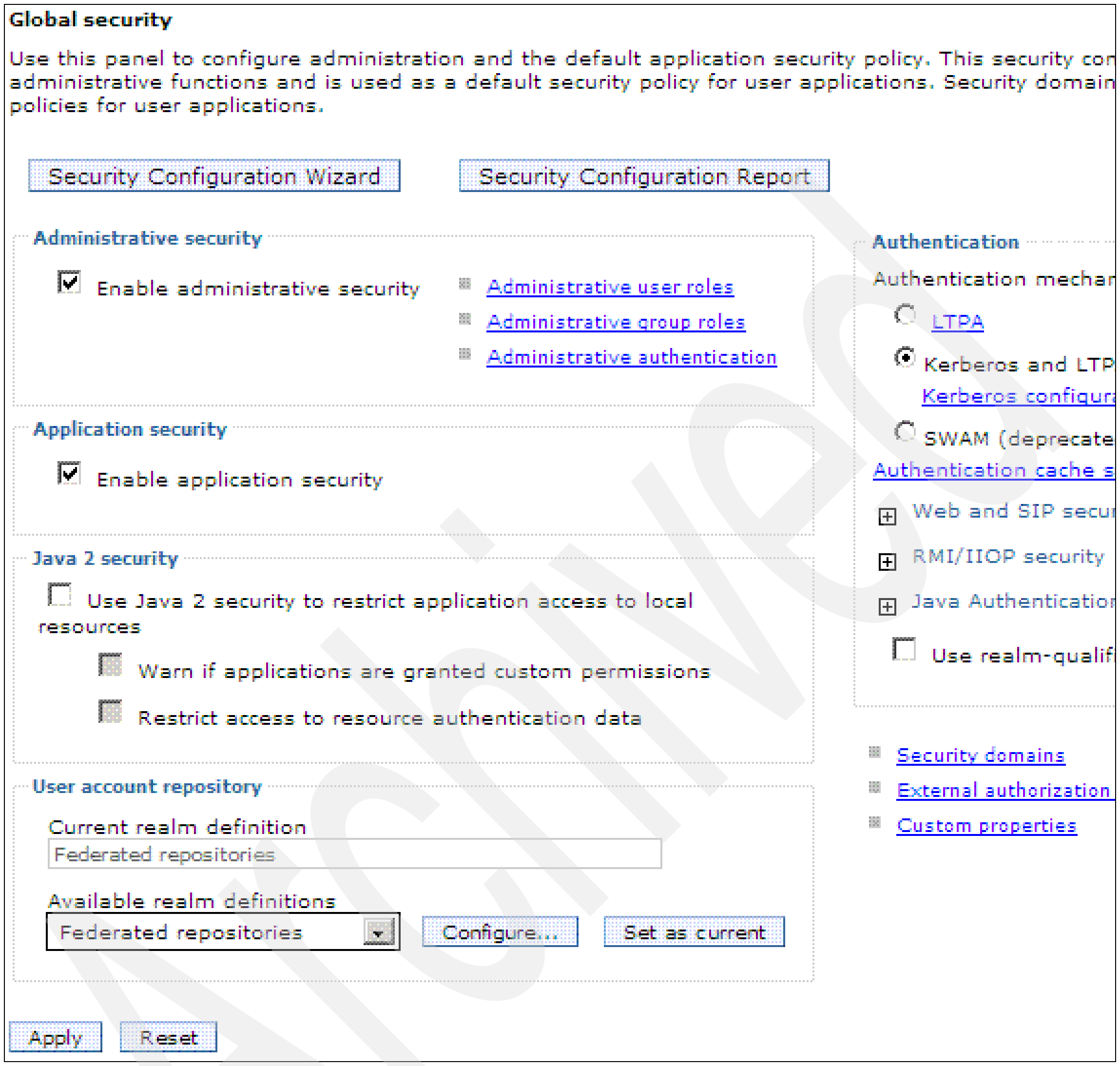
```
wsadmin>$AdminTask createIdMgrRealm {-name KKDC.TEST.COM}
CWWIM5047W Each configured realm must contain at least one
participating base entry. Add a base entry before saving the
configuration.
wsadmin>$AdminTask addIdMgrRealmBaseEntry {-name KKDC.TEST.COM
-baseEntry CN=Users,DC=KKDC,DC=TEST,DC=COM}
CWWIM5028I The configuration is saved in a temporary workspace. You
must use the "$AdminConfig save" command to save it in the master
repository.
```

10. Then, select **Security** → **Global Security**, and complete the following steps, as shown in Figure 9-7:

    a. Select **Federated repositories** from the Available realm definitions drop-down menu, and then click **Set as current**.

    b. Select the “Enable administrative security” and “Enable application security” options.

    c. Click **Apply**.


Global security

Use this panel to configure administration and the default application security policy. This security con
administrative functions and is used as a default security policy for user applications. Security domain
policies for user applications.

Security Configuration Wizard | Security Configuration Report

Administrative security

[✓] Enable administrative security

- Administrative user roles
- Administrative group roles
- Administrative authentication

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

Authentication

Authentication mechar

( ) LTPA

(●) Kerberos and LTP
Kerberos configura

( ) SWAM (deprecate

Authentication cache s

⊞ Web and SIP secu

⊞ RMI/IIOP security

⊞ Java Authenticatio

[ ] Use realm-qualif

- Security domains
- External authorization
- Custom properties

Apply | Reset

Figure 9-7 Enabling federated repositories

11. Save the configuration. Restart WebSphere Application Server if you are using a stand-alone architecture, or restart the components of your cell if you are using a network deployment architecture.

You should now be able to connect into the WebSphere administrative console using the administrative ID that you specified in the federated repositories configuration for this scenario. We set `wasadmin` as the administrative ID. The `wasadmin` user ID belongs to the `kdc.itso.com` realm.

9.3.2 Creating the Kerberos configuration file

Use **wsadmin** to create a Kerberos configuration file that WebSphere Application Server will use. Use the **wsadmin** tool located in the *profile_home*/bin directory to create the Kerberos file, as shown in Example 9-6. (We describe the **createKrbConfigFile** command and its parameters in 7.3.1, "Configuring the Kerberos configuration file" on page 161.)

Example 9-6 The wsadmin createKrbConfigFile command output

```
wsadmin>$AdminTask createKrbConfigFile {-krbPath /etc/krb5.conf -realm
KDC.ITSO.COM -kdcHost saw921-sys2.kdc.itso.com -dns kdc.itso.com
-keytabPath /etc/saw921-sys3.keytab}
/etc/krb5.conf has been created.
```

Note: You can also use the **createKrbConfigFile** command in interactive mode. To execute this command in interactive mode, use the following command:

```
$AdminTask createKrbConfigFile -interactive
```

The Kerberos configuration file is created in the /etc directory. You must manually add the information about the foreign realm (KKDC.TEST.COM) to the [realms] and [domain_realm] sections of the Kerberos configuration file. Open the /etc/krb5.conf file with any text editor, and add the information specified in Example 9-7.

Example 9-7 Editing /etc/krb5.conf configuration file

```
[libdefaults]
   default_realm = KDC.ITSO.COM
   default_keytab_name = FILE:/etc/saw921-sys3.keytab
   default_tkt_enctypes = rc4-hmac des-cbc-md5
   default_tgs_enctypes = rc4-hmac des-cbc-md5
   forwardable  = true
   renewable  = true
   noaddresses = true
   clockskew  = 300
[realms]
   KDC.ITSO.COM = {
      kdc = saw921-sys2.kdc.itso.com:88
      default_domain = kdc.itso.com
   }
   KKDC.TEST.COM = {
      kdc = sys7.kkdc.test.com:88
      default_domain = kkdc.test.com
   }
[domain_realm]
   .kdc.itso.com = KDC.ITSO.COM
   .kkdc.test.com = KKDC.TEST.COM
```

If your network environment has a *transitive* trust between realms, you have to make an additional configuration for the Kerberos configuration file. For more information about the steps to configure the `krb5.conf` file for a transitive trust between realms, see the WebSphere Application Server Information Center:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.express.doc/info/exp/ae/tsec_kerb_create_conf.html

To validate your Kerberos configuration file, use the **`wsadmin validateKrbConfig`** command. This command validates the Kerberos realm against the default Kerberos realm in the Kerberos configuration file (`/etc/krb5.conf`). If everything is OK, the output for the command will be `true` as shown in Example 9-8.

Example 9-8 Validating kerberos configuration in wsadmin

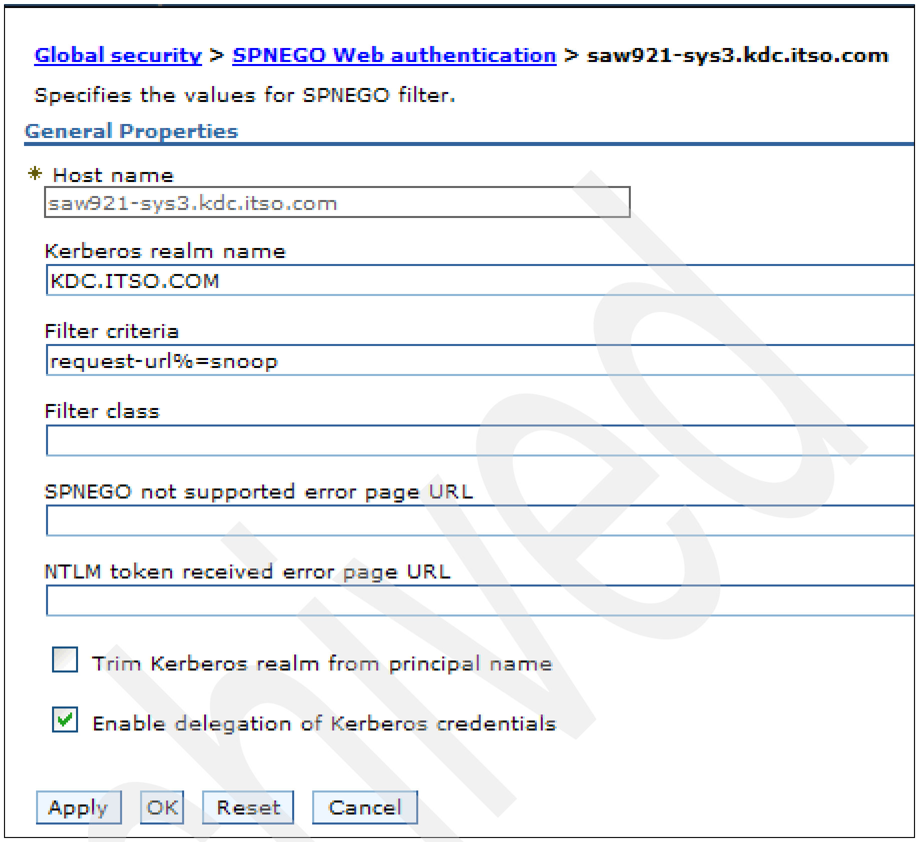
```
wsadmin>$AdminTask validateKrbConfig
true
```

9.3.3 Configuring SPNEGO Web authentication

Next, access the administrative console to configure SPNEGO and Kerberos as follows:

1. Select **Security** → **Global Security**. Expand **Web and SIP security**, and click **SPNEGO Web authentication**.
2. Click **New** to create a new filter for your sample application. In the General Properties window, shown in Figure 9-8, complete the following steps:
   a. Enter the fully qualified host name of the WebSphere server in the “Host name” field.
   b. Enter the realm name in the “Kerberos realm name” field.
   c. In the “Filter criteria” field, enter `request-url%=snoop`. For more information about filter criteria, see:

      http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.express.doc/info/exp/ae/usec_kerb_SPNEGO_config.html
   d. Leave the “Trim Kerberos realm from principal name” option cleared to leave the Kerberos realm on the principal user name.
   e. Click **Apply**.


Global security > SPNEGO Web authentication > saw921-sys3.kdc.itso.com

Specifies the values for SPNEGO filter.

General Properties

* Host name: saw921-sys3.kdc.itso.com

Kerberos realm name: KDC.ITSO.COM

Filter criteria: request-url%=snoop

Filter class:

SPNEGO not supported error page URL:

NTLM token received error page URL:

[ ] Trim Kerberos realm from principal name

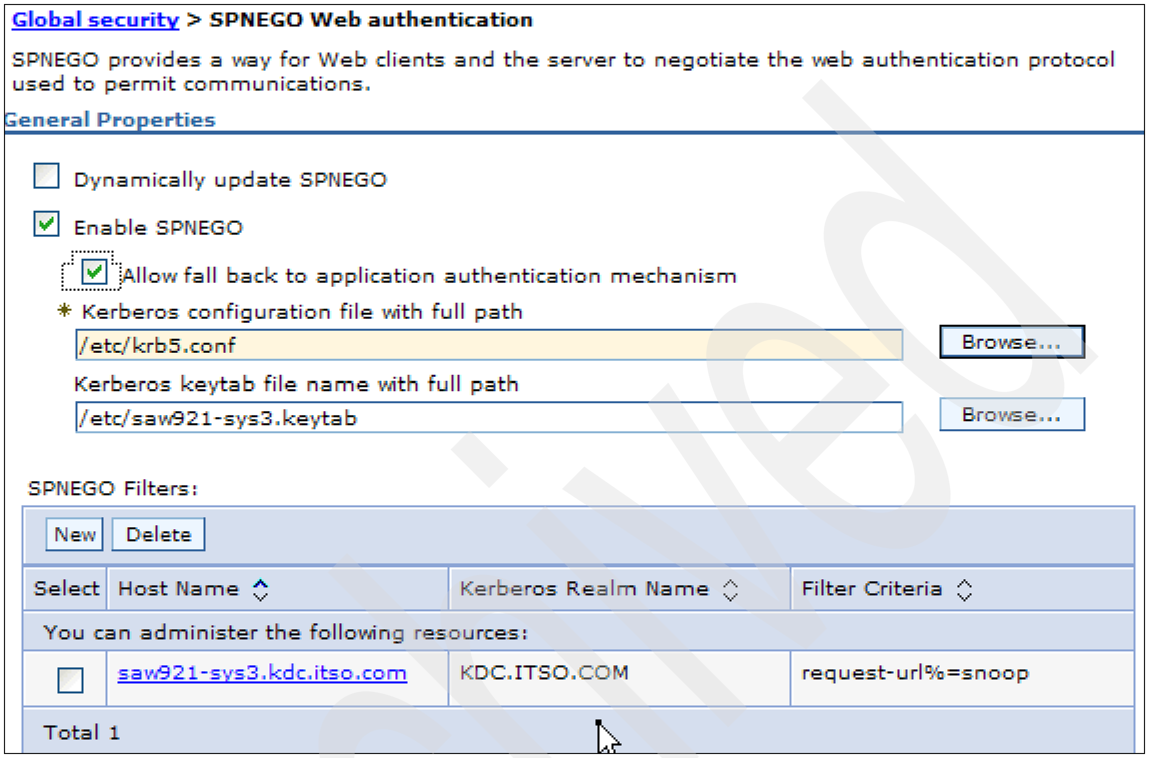
[✓] Enable delegation of Kerberos credentials

Figure 9-8 SPNEGO Web Authentication General Properties

3. Back in the SPNEGO Web authentication page, shown in Figure 9-9 on page 210, complete the following information:
   a. Select the “Dynamically update SPNEGO” option if you want WebSphere Application Server to dynamically update the SPNEGO runtime when SPNEGO changes occur without restarting the WebSphere Application Server.
   b. Select **Enable SPNEGO**.
   c. Select the “Allow fall back to application authentication mechanism” option to specify that SPNEGO is used as a Web authenticator to log in to WebSphere Application Server first. If the login fails, the application authentication mechanism is then used to log in to WebSphere Application Server.

d. Enter the values for the Kerberos configuration and keytab files.

e. Click **Apply**.



Global security > SPNEGO Web authentication

SPNEGO provides a way for Web clients and the server to negotiate the web authentication protocol used to permit communications.

General Properties

[ ] Dynamically update SPNEGO

[✓] Enable SPNEGO

[✓] Allow fall back to application authentication mechanism

* Kerberos configuration file with full path

/etc/krb5.conf

Kerberos keytab file name with full path

/etc/saw921-sys3.keytab

SPNEGO Filters:

Select	Host Name	Kerberos Realm Name	Filter Criteria
You can administer the following resources:			
[ ]	saw921-sys3.kdc.itso.com	KDC.ITSO.COM	request-url%=snoop
Total 1			

Figure 9-9 Configuring SPNEGO Web authentication

9.3.4 Configuring Kerberos

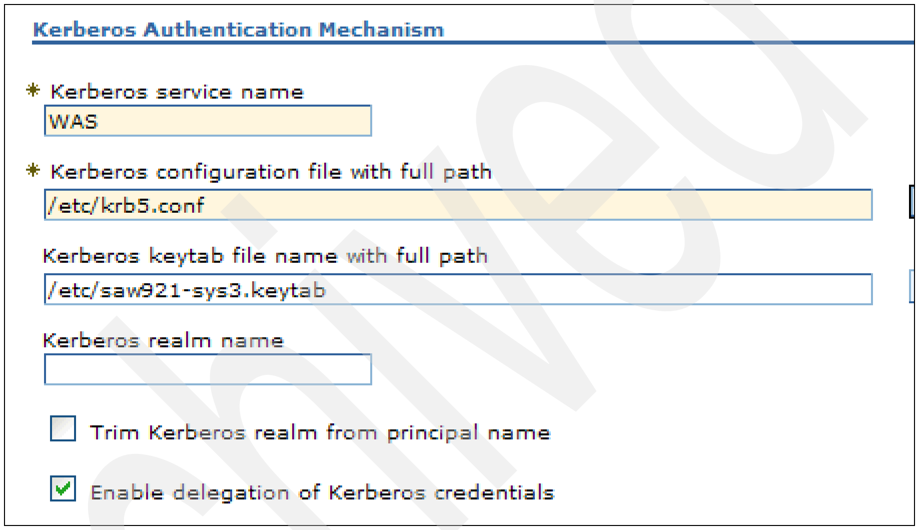
Update the Kerberos configuration settings for WebSphere Application Server as follows:

1. In the administrative console, click **Security** → **Global Security**, and select **Kerberos configuration**.
2. Complete the following fields with the information about the location of the Kerberos file configuration and keytab file, as shown in Figure 9-10:
   a. Enter `WAS` for the Kerberos service name.
   b. Enter the realm name `KDC.ITSO.COM` (in uppercase letters because the `krb5.conf` file was written using uppercase letters for realm name).

c. Leave the “Trim Kerberos realm from principal name” option cleared to leave the Kerberos realm on the principal user name.

d. Select the “Enable delegation of Kerberos credentials” option to specify whether the Kerberos delegated credentials are stored in the subject by the Kerberos authentication. This option also enables an application to retrieve the stored credentials and to propagate them to other applications downstream for additional Kerberos authentication with the credentials from the Kerberos client.

e. Click **Apply**.


Kerberos Authentication Mechanism

* Kerberos service name: WAS

* Kerberos configuration file with full path: /etc/krb5.conf

Kerberos keytab file name with full path: /etc/saw921-sys3.keytab

Kerberos realm name:

☐ Trim Kerberos realm from principal name

✓ Enable delegation of Kerberos credentials

Figure 9-10 Configuring Kerberos

3. Save the configuration and restart the WebSphere Application Server.

9.3.5 Configuring alias support

In WebSphere Application Server V7.0.0.5, when WebSphere receives an HTTP request with a SPNEGO token, it verifies whether the target host name is configured for SPNEGO Web authentication. If the host name is *not* configured but it resolves to an actual host name that is configured for SPNEGO Web authentication by DNS lookup, WebSphere continues to process the HTTP request using the actual host name. With this change, you do not need to configure alias host names for SPNEGO Web authentication as long as the real host name is configured. You can add an alias host name dynamically without having to restart WebSphere to change the configuration for SPNEGO Web authentication to work.

In addition to the basic SPNEGO Web authentication configuration, you need to complete the following steps for WebSphere to perform a DNS lookup for SPNEGO Web authentication:

1. Configure the actual host name to which the alias host name resolves. In the administrative console, select **Security** → **Global Security** → **SPNEGO Web authentication** → **New**. Then, enter the real host name for the system in the **Host name** property.
2. Turn on Canonical support by selecting **Security** → **Global Security** → **Custom properties**. Then, add the new property `com.ibm.websphere.security.krb.canonical_host`. Set the value of this property to `true`.
3. Add the alias host name as a trusted URL in the client browser.

9.4 Validating the scenario

You can validate this scenario by creating a request from a workstation to access an application on WebSphere Application Server (the snoop servlet). The workstation, `ws01.kkdc.test.com`, is a member of the `KKDC.TEST.COM` realm. The user of the workstation is logged in to the `kkdc.test.com` domain controller with the `skarolyn` user ID. The request is made using the HTTP protocol. The target system is the Linux system where WebSphere Application Server is hosted. Figure 9-11 illustrates the flow for the request.

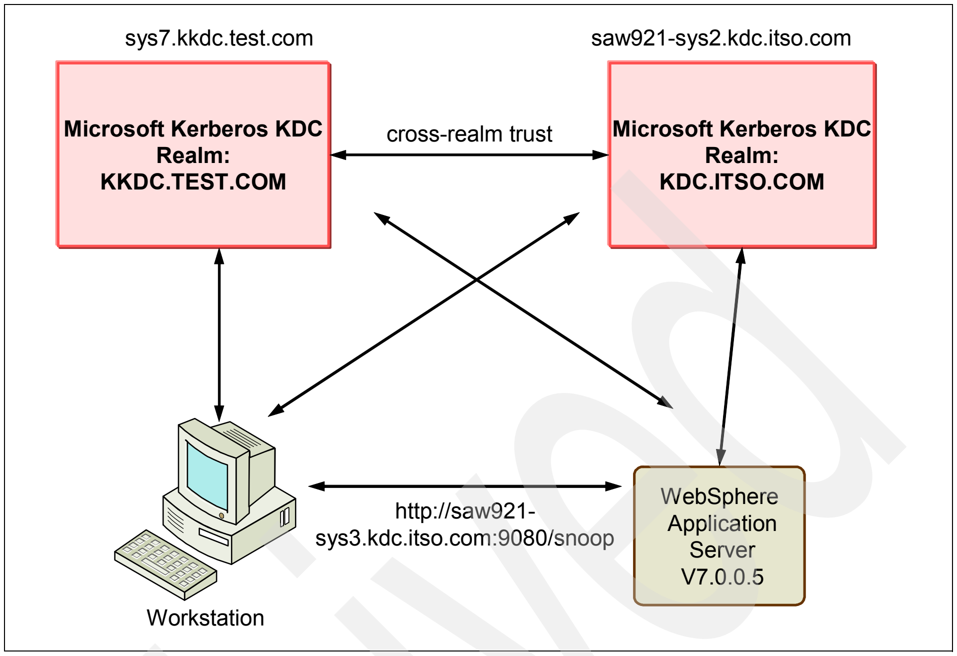
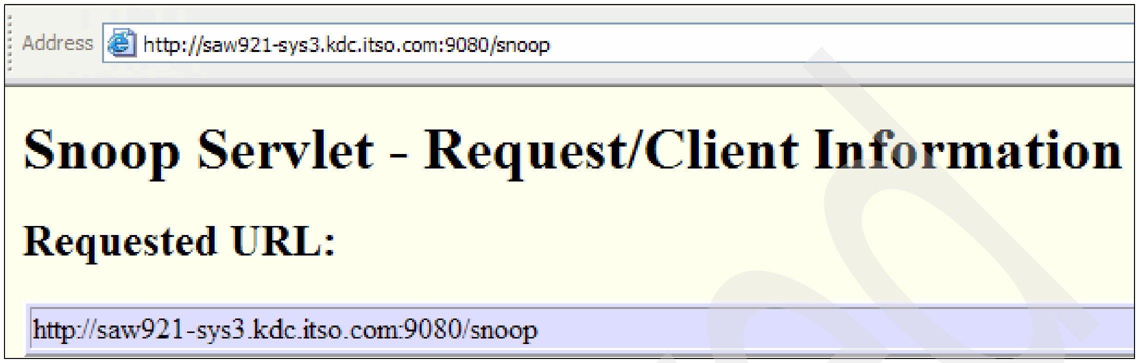

sys7.kkdc.test.com
saw921-sys2.kdc.itso.com
Microsoft Kerberos KDC Realm: KKDC.TEST.COM
cross-realm trust
Microsoft Kerberos KDC Realm: KDC.ITSO.COM
http://saw921-sys3.kdc.itso.com:9080/snoop
WebSphere Application Server V7.0.0.5
Workstation

Figure 9-11   Request flow for this scenario

Before attempting to access the application, verify that the Web browser is enabled to use SPNEGO (as described in Appendix B, “Configuring Web browsers for SPNEGO” on page 499). Also, verify that the address of the application server is added under the specific configuration area for the Web browser.

To validate the scenario, follow these steps:

1. Open the Web browser, and access the snoop application through the Web container port of the application server, as shown in Figure 9-12.


Address http://saw921-sys3.kdc.itso.com:9080/snoop

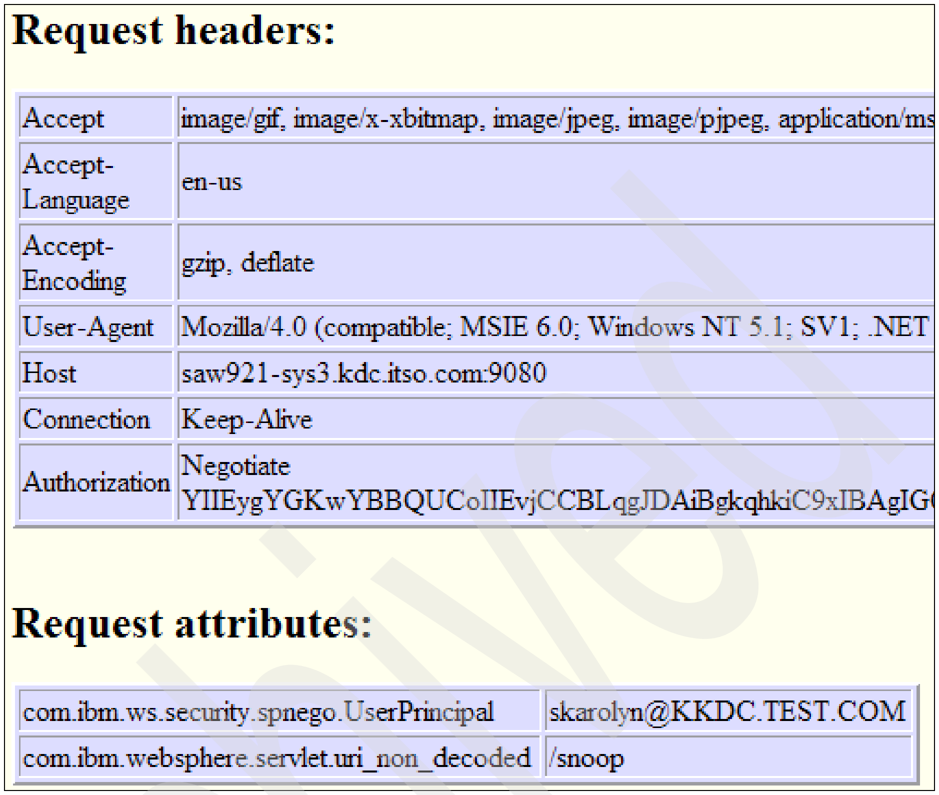
Snoop Servlet - Request/Client Information

Requested URL:

http://saw921-sys3.kdc.itso.com:9080/snoop

Figure 9-12 Requesting snoop Web page

2. If everything is configured correctly, the snoop welcome page displays without your having to re-enter the user ID and password. You can see the request attributes by scrolling down the page. Figure 9-13 shows information about *UserPrincipal*.



Request headers:

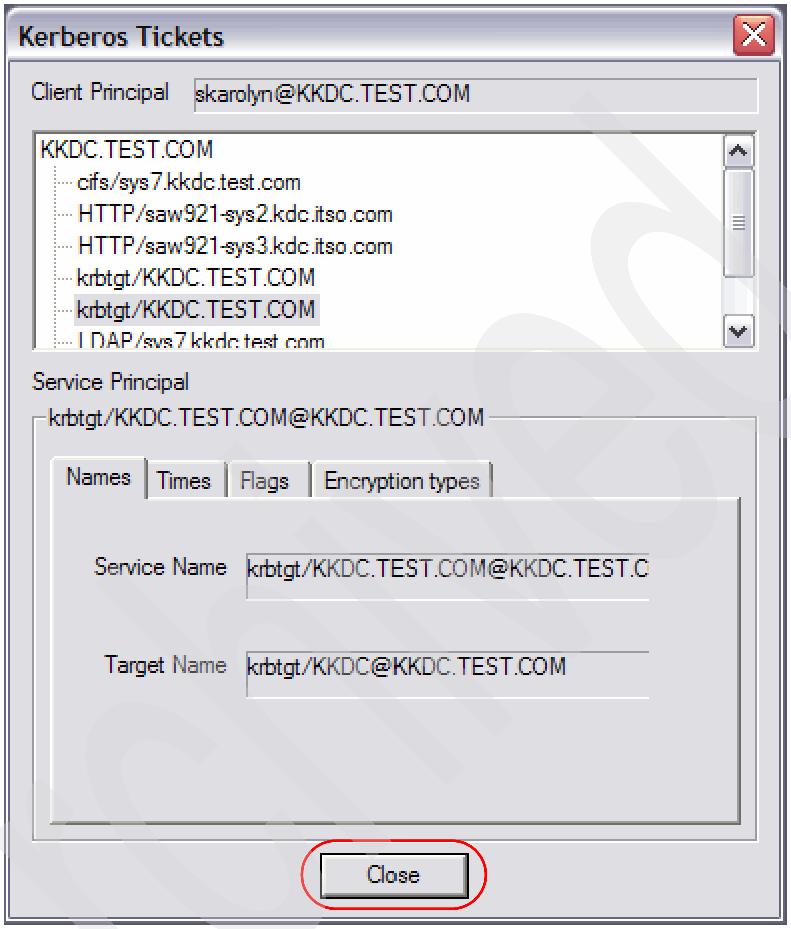
Accept	image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/ms
Accept-Language	en-us
Accept-Encoding	gzip, deflate
User-Agent	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET
Host	saw921-sys3.kdc.itso.com:9080
Connection	Keep-Alive
Authorization	Negotiate YIIEygYGKwYBBQUCoIIEvjCCBLqgJDAiBgkqhkiC9xIBAgIG

Request attributes:

com.ibm.ws.security.spnego.UserPrincipal	skarolyn@KKDC.TEST.COM
com.ibm.websphere.servlet.uri_non_decoded	/snoop

Figure 9-13 Request attributes from snoop application

3. Use **`kerbtray.exe`** to verify information about the Kerberos tickets that is provided by `KKDC.TEST.COM`, as shown in Figure 9-14.



```
Kerberos Tickets

Client Principal   skarolyn@KKDC.TEST.COM

KKDC.TEST.COM
   cifs/sys7.kkdc.test.com
   HTTP/saw921-sys2.kdc.itso.com
   HTTP/saw921-sys3.kdc.itso.com
   krbtgt/KKDC.TEST.COM
   krbtgt/KKDC.TEST.COM
   LDAP/sys7.kkdc.test.com

Service Principal
krbtgt/KKDC.TEST.COM@KKDC.TEST.COM

Names | Times | Flags | Encryption types

Service Name   krbtgt/KKDC.TEST.COM@KKDC.TEST.C
Target Name    krbtgt/KKDC@KKDC.TEST.COM

Close
```

Figure 9-14 Output for kerbtray

10

WS-SecurityKerberos with a J2EE Web services client

This chapter discusses the OASIS Web Services Security (WSS) Kerberos Token Profile 1.1 specification and its support in WebSphere Application Server V7. It also provides an example of how to apply the Kerberos token to secure Web services.

This chapter includes the following topics:

- Scenario overview
- Configuring the KDC and client system
- Configuring WebSphere Application Server
- Validating the scenario by monitoring the SOAP traffic

10.1 Scenario overview

Before you can use Kerberos with Web services security (WSS), you must configure Kerberos in WebSphere Application Server. You do not need to enable Kerberos as the authentication mechanism. However, the Kerberos configuration file, `krb5.conf` or `krb5.ini`, and the Kerberos keytab file, `krb5.keytab`, are required.

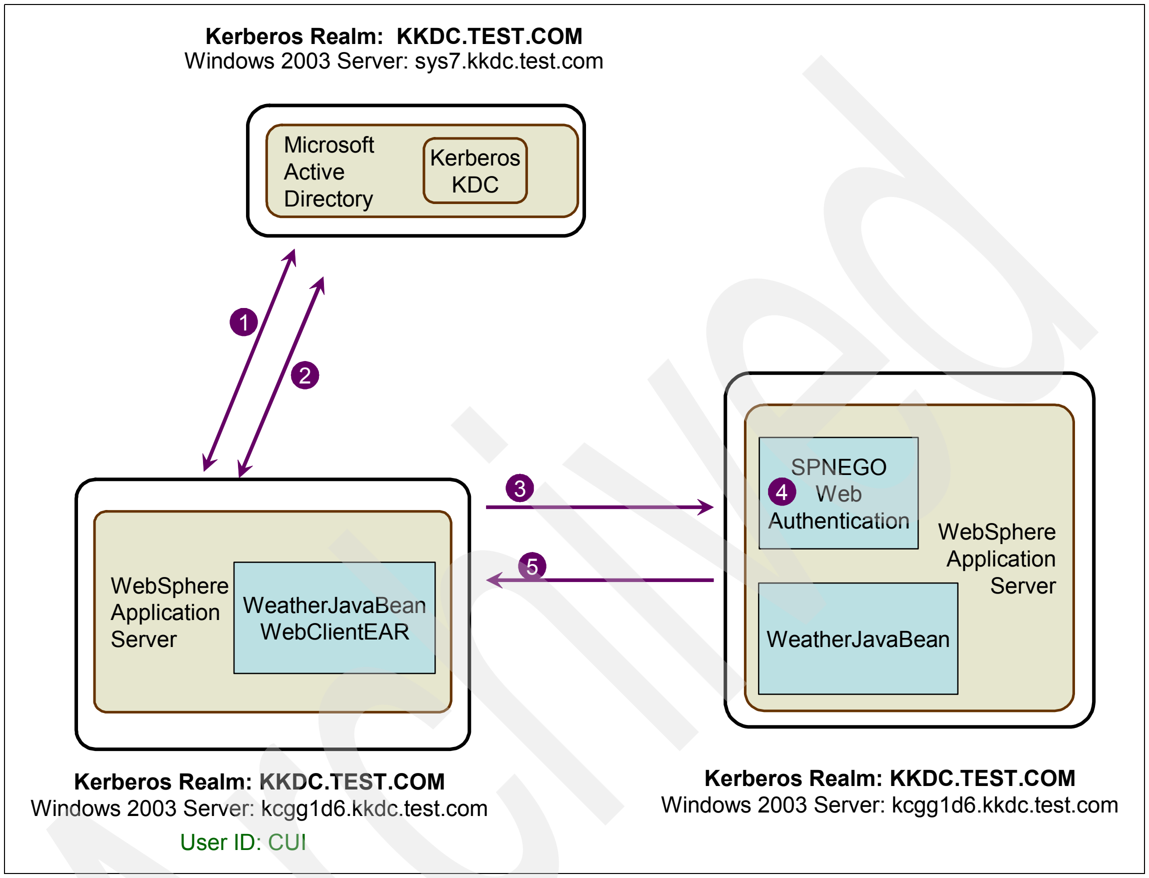
The initial setup and configuration processes to use Kerberos with WSS are identical to the configuration processes for using Kerberos or SPNEGO Web with the security function. Therefore, you must set up and configure Kerberos before continuing with the steps in this section.

Note: This scenario features the following components:

- WS-SecurityKerberos to secure Web services traffic at the message level
- Policy sets to specify Kerberos authentication
- Microsoft Kerberos key distribution center (KDC)
- Web service client application: WeatherJavaBeanWebClientEAR
- WebSphere Application Server configuration:
  - User registry: Microsoft Active Directory
  - Authentication mechanism: LTPA
  - Kerberos authentication and configuration file
  - Web service provider application: WeatherJavaBean

10.1.1 Step-by-step flow of this scenario

Figure 10-1 illustrates the basic flow of this scenario.



Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: sys7.kkdc.test.com
Microsoft Active Directory
Kerberos KDC
1
2
3
4
5
WebSphere Application Server
WeatherJavaBean WebClientEAR
SPNEGO Web Authentication
WebSphere Application Server
WeatherJavaBean
Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: kcgg1d6.kkdc.test.com
User ID: CUI
Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: kcgg1d6.kkdc.test.com

Figure 10-1 J2EE Client using Kerberos authentication for J2EE provider

The numbers in the figure correspond to the following steps in this process:

1. To begin, the user logs on to the Windows domain from the workstation. After the user enters a user name and password, Windows logon sends the information to the workstation local security authority, which passes the request to the Kerberos authentication package.

   The Web service client application, `WeatherJavaBeanWebClientEAR`, sends an initial authentication request (AS_REQ) that includes the user's credentials and an encrypted time stamp to the KDC. This request is a request for authentication and a ticket-granting ticket (TGT).

The first domain controller in the domain generates the `krbtgt/`*`domain_name`* ticket. The KDC uses the secret key to decrypt the time stamp and issues a TGT to the client.

The response AS_REP is encrypted with the user's key and is returned to the user. The ticket is encrypted with the KDC's key and is enclosed in the AS_REP.

2. The Web service client application sends a request (using the TGT) for a service ticket to the ticket-granting server for communication with the Web services. The ticket-granting server then returns a service ticket (TGS_REP) and the session key to the client. The client uses this information to create the Kerberos token.
3. After the Kerberos token is created, the WSS generator encodes and inserts the token into the SOAP message and propagates the token for token consumption or acceptance.
4. The SPNEGO Web authentication mechanism on the provider system receives and extracts the Kerberos token from the SOAP request. The provider then accepts the Kerberos token by validating the token with its own secret key. The secret key of the service is stored in a keytab file. After acceptance, the Web services provider stores the associated request token information into the context Subject.
5. The provider returns a SOAP response.

10.1.2 Scenario components

This scenario shows how to configure and to verify Kerberos as the authentication mechanism for the Web services. In this scenario, we use a Kerberos token to authenticate the Web services client for a Weather forecast application.

The scenario in this chapter uses the following components:

- The application server

  WebSphere Application Server V7.0.0.5 installed on Windows XP Professional 2002 SP2

  This system hosts two applications:

  – The WeatherJavaBeanServer Web service provider
  – The WeatherJavaBeanWebClientEar Web service client

- The Kerberos KDC

  Microsoft Windows Server 2003 SP2 with Active Directory. The Windows 2003 Support Tools are installed.

- The Web services sample application

  The sample used in this scenario is the WeatherJavaBean service. The sample includes Web service provider (`WeatherJavaBeanServer` application) and a Web service client (`WeatherJavaBeanWebClientEar`).

  This scenario creates a new application policy set for Kerberos authentication, called *ITSO Kerberos*, and attaches the policy set to both the Web services client and server. New application-specific bindings for use with the policy set are also created for the client and server.

Download materials: You can import the WeatherJavaBean project that we use in the scenario as a project interchange file from the `WeatherWebService.zip` file included with this book. You also need to install the weather database as described in “Importing project interchange files” on page 510.

After you import the project, be sure to open the WebSphere Application Server Deployment for WeatherJavaBeanServer and update the data source so that it points to the database that you installed.

For more information about how to download the compressed file, see Appendix E, “Additional material” on page 525.

10.1.3 Basic requirements for this scenario

This scenario requires the following basic components:

- A working domain controller and at least one client computer in that domain

  Using SPNEGO from the domain controller does not work.

- A functioning Microsoft Windows 2000 or 2003 Active Directory Domain that includes the following components:
  - Domain controller
  - Client workstation

  Users must be able to log in to the domain controller.

- A functioning WebSphere Application Server for Windows with application security enabled
- Access for users from the Microsoft Active Directory to WebSphere Application Server’s protected resources using a native authentication mechanism, such as basic authentication (BA) or forms authentication

- Windows 200x Support Tools on the Microsoft Windows Server, including the **setspn**, the **ktpass**, and the **adsiedit** tools that this scenario uses

  These support tools are available on the Windows Server installation CD or from the following Microsoft Web site:

  http://www.microsoft.com/downloads/details.aspx?familyid=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D

We also recommend that you install the Windows Server 2003 Resource Kit Tools, which provide the **kerbtray.exe** utility to see Kerberos tickets on the user workstation. These tools are available at:

http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd

10.2 Configuring the KDC and client system

To configure the KDC and client system, complete these steps first:

1. Set up a Windows Server with Microsoft Active Directory domain and associated Kerberos KDC. The Windows administrator needs to complete this step.
2. Join the Windows server and the Microsoft Active Directory domain controller. Right click the **My Computer** icon. Then, click **Properties**. In the System Properties dialog box, go to the Computer Name tab, and click **Change**. Enter the Microsoft Active Directory domain as the system name, and click **OK**. For example, in our scenario, the computer `kcgg1d6` joined the Microsoft Active Directory domain `kkdc.test.com`.
3. Log on to the Microsoft Windows Active Directory domain with an identity that is known in Active Directory. In our scenario, we logged on to the Microsoft Active Directory domain with the user name `cui`.
4. Make sure that you can connect to the Kerberos KDC. Use the **kinit** tool that ships with the JDK to test the connection as follows:
   a. Open a command prompt, and navigate to the *app_server_root*`\java\jre\bin` directory.
   b. Type **kinit** and your user name. Then, enter your password.

   Example 10-1 shows a successful connection to the Kerberos domain.

   Example 10-1 Use kinit to test the connection

```
C:\WAS7\java\jre\bin>kinit cui
Password for cui@KKDC.TEST.COM:
```

```
Done!
New ticket is stored in cache file C:\Documents and
Settings\cui.KKDC.000\krb5cc_cui
```

10.2.1 Creating the SPN for WebSphere Application Server

Next, you need to create a user account for WebSphere Application Server in the Microsoft Active Directory. This account is mapped to the Kerberos service principal name (SPN). The SPN identifies the WebSphere Application Server installation as a Kerberized service. To create this SPN for WebSphere Application Server, follow these steps:

1. Log on as an administrator to the Windows 2003 Server system that serves as the domain controller.
2. Invoke the Microsoft Active Directory users and computers snap-in. Click **Start** → **Run**. Then type the following command, and click **OK**:

   ```
   dsa.msc /server=localhost
   ```

3. Create a user account for WebSphere Application Server V7. This scenario creates the *kcggws* user account.

10.2.2 Creating the keytab file

Now, you can create the Kerberos keytab file. Use the **ktpass** tool to set up the Kerberos keytab file (krb5.keytab) for the previously created SPN. With the **ktpass** command, you can configure a non-Windows Server 2003 Kerberos service (such as WebSphere applications on a z/OS platform) as a security principal in the Windows Server 2003 Active Directory. This tool generates a keytab file that contains the shared key of the service. In our scenario, we use the command shown in Example 10-2.

Example 10-2 Output from the ktpass command

```
ktpass -out c:\kcggws.keytab -princ
HTTP/kcgg1d6.kkdc.test.com@KKDC.TEST.COM -mapUser kcggws -mapOp set
-pass Passw0rd -crypto RC4-HMAC-NT
```

10.3 Configuring WebSphere Application Server

To configure the WebSphere Application Server for the Web service provider and client, you need to complete the following tasks:

- Creating the Kerberos configuration file
- Configuring the user registry
- Configuring the Kerberos token policy set
- Applying the policy set and binding to the Web service provider
- Applying the Kerberos Policy set and binding to the Web service client

10.3.1 Creating the Kerberos configuration file

Create the Kerberos configuration file in the system that hosts WebSphere Application Server (`kcgg1d6.kkdc.test.com`). Use the **`wsadmin`** utility to generate the Kerberos configuration file as follows:

1. Copy the keytab file (`kcggws.keytab`) to the WebSphere Application Server system. This scenario copies the file to the `c:\windows` folder.
2. Start **`wsadmin`** from the *`app_server_root`*`/bin` folder. Use **`wsadmin`** to create the associated Kerberos configuration as shown in Example 10-3.

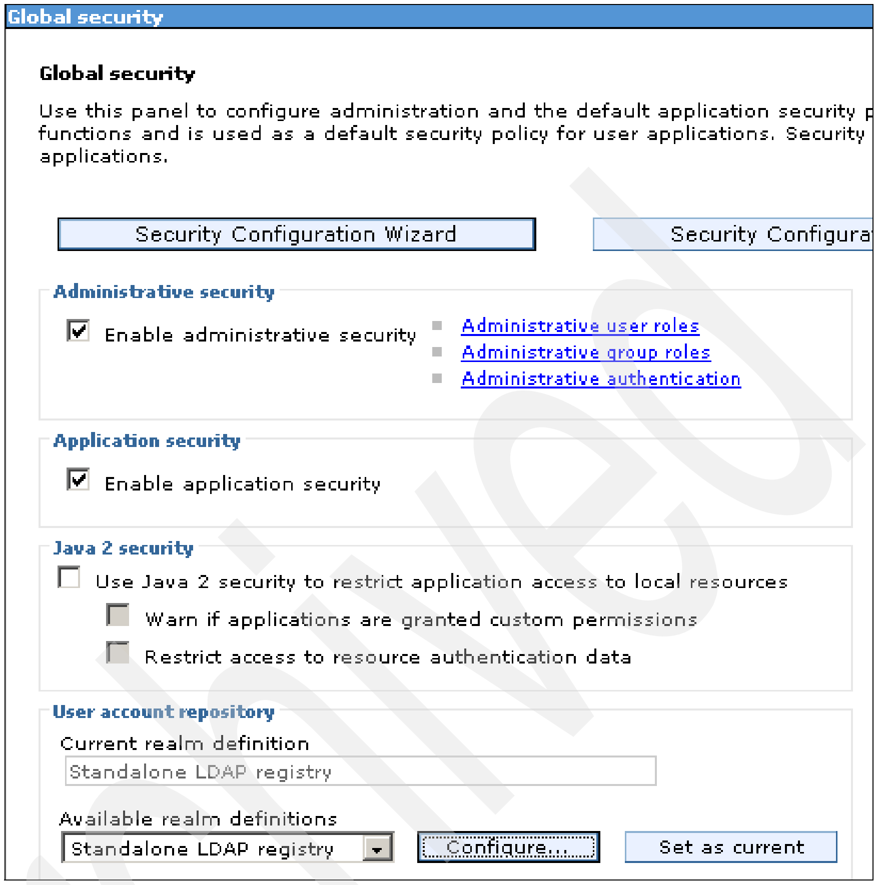
Example 10-3 Generating the krb5.ini file using wsadmin

```
wsadmin>$AdminTask createKrbConfigFile {-krbPath c:/WINDOWS/krb5.ini
-realm KKDC.TEST.COM -kdcHost sys7.kkdc.test.com -dns kkdc.test.com
-keytabPath c:/WINDOWS/kcggws.keytab}
```

10.3.2 Configuring the user registry

Configure WebSphere Application Server global security to connect to the Microsoft Active Directory as follows:

1. Log on to the WebSphere Application Server administrative console and navigate to **Security** → **Global security**. In the Global security panel, shown in Figure 10-2, complete these steps:
   a. Select the “Enable administrative security” and “Enable application security” options.
   b. Clear the “Use Java 2 security to restrict application access to local resources” option.
   c. Select **Standalone LDAP registry** in the Available realm definitions section, and click **Configure**.


Global security

Global security

Use this panel to configure administration and the default application security p
functions and is used as a default security policy for user applications. Security
applications.

Administrative security

☑ Enable administrative security

- Administrative user roles
- Administrative group roles
- Administrative authentication

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

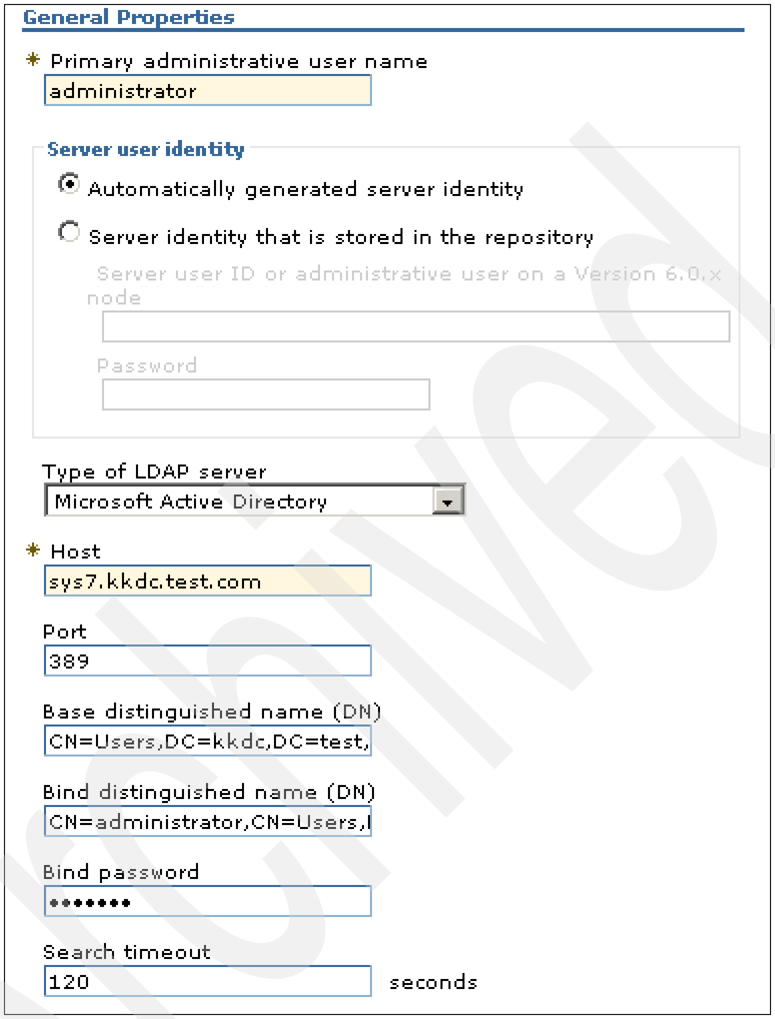
Standalone LDAP registry

Available realm definitions

Standalone LDAP registry

Figure 10-2 Set Global security

2. Enter the active directory information, as shown in Figure 10-3, and click **OK**.



General Properties

* Primary administrative user name: administrator

Server user identity

(●) Automatically generated server identity

( ) Server identity that is stored in the repository

Server user ID or administrative user on a Version 6.0.x node

Password

Type of LDAP server: Microsoft Active Directory

* Host: sys7.kkdc.test.com

Port: 389

Base distinguished name (DN): CN=Users,DC=kkdc,DC=test,

Bind distinguished name (DN): CN=administrator,CN=Users,I

Bind password: ●●●●●●●

Search timeout: 120 seconds

Figure 10-3 Active Directory configuration

3. Click **Set as Current**.
4. Click **Save** to save changes to the master repository.

10.3.3 Configuring the Kerberos token policy set

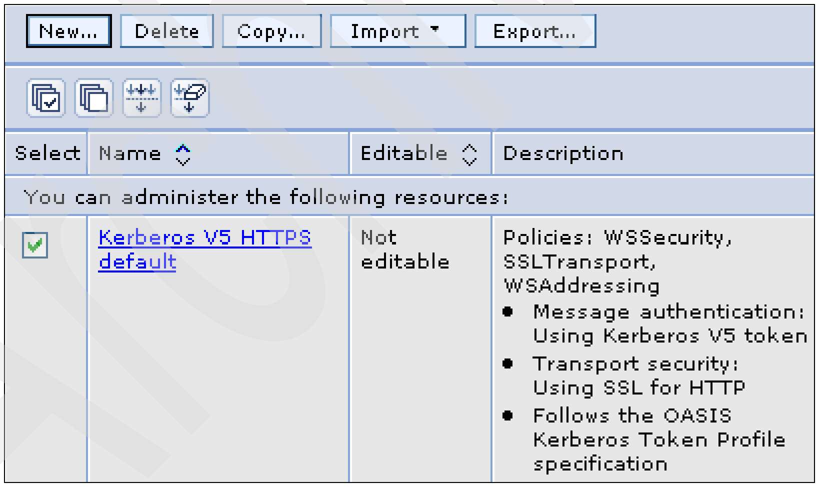
The Kerberos V5 HTTPS default policy ships with WebSphere Application Server V7 and is configured for use. This policy is also available as a template that you can copy and customize to suit your applications.

The policy set provides message authentication with a Kerberos Version 5 token. Message integrity and confidentiality are provided by Secure Sockets Layer (SSL) transport security. This policy set follows the OASIS Web Services Security (WSS) Kerberos Token Profile 1.1 specification.

For our example, we use a Kerberos token for the message authentication only. Message integrity and confidentiality are not needed. So, we use the Kerberos V5 HTTPS default policy set as a template to create our own policy set.

To configure the Kerberos token policy set, follow these steps:

1. Log in to the WebSphere administrative console.
2. In the left pane, select **Services** → **Policy sets** → **Application policy sets**.
3. In the right pane, select **Kerberos V5 HTTPS default**, as shown in Figure 10-4, and click **Copy**.



Select	Name	Editable	Description
You can administer the following resources:			
✓	Kerberos V5 HTTPS default	Not editable	Policies: WSSecurity, SSLTransport, WSAddressing ● Message authentication: Using Kerberos V5 token ● Transport security: Using SSL for HTTP ● Follows the OASIS Kerberos Token Profile specification

Figure 10-4 Copy Kerberos V5 HTTPS default

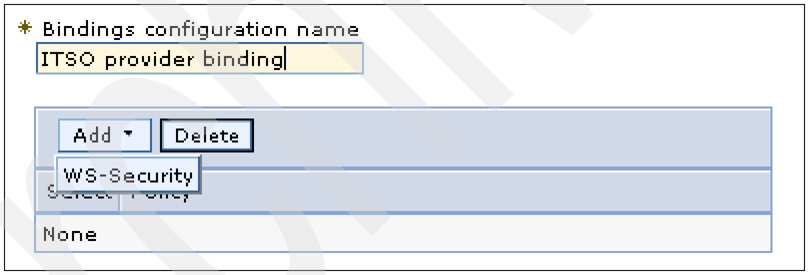
4. In the Name field, enter `ITSO Kerberos`. Then, click **OK**, and then **Save**.
5. Click **ITSO Kerberos**.

6. For our example, we need to use only the Kerberos token for authentication. Select **SSL transport** and **WS-Addressing**, and then click **Delete** (leaving only WSS).
7. Click **Save**.

10.3.4 Applying the policy set and binding to the Web service provider

To assign the policy set to the weather forecast Web service, complete the following tasks in the WebSphere administrative console:

1. In the left pane, click **Services** → **Service providers**.
2. Click **WeatherJavaBeanService**. Select **WeatherJavaBeanService**, and then click **Attach Policy Set**. Then, select **ITSO Kerberos** from the drop-down list.
3. Select **WeatherJavaBeanService** again, and click **Assign Binding**. Select **New Application Specific Binding**.
4. Enter `ITSO provider binding` as the Bindings configuration name.
5. Click **Add,** and select **WS-Security** as shown in Figure 10-5.

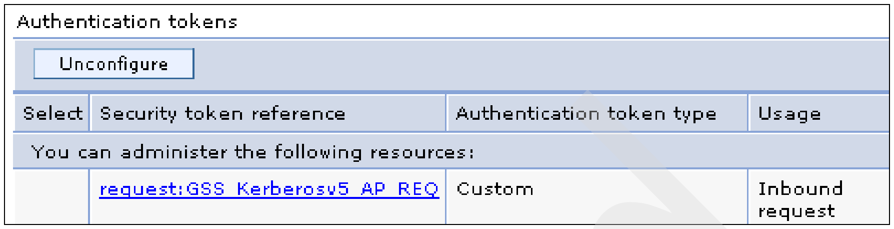

* Bindings configuration name
ITSO provider binding

None

Figure 10-5 Apply the binding

6. Click **Authentication and protection**.

7. In the Authentication tokens section, click **request:GSS_Kerberosv5_AP_REQ** as shown in Figure 10-6.


Authentication tokens

Select	Security token reference	Authentication token type	Usage
You can administer the following resources:			
	request:GSS_Kerberosv5_AP_REQ	Custom	Inbound request

Figure 10-6 Authentication tokens

8. Click **OK**. The status for `request:GSS_Kerberosv5_AP_REQ` is shown as `Configured`.
9. Click **Save**.

10.3.5 Applying the Kerberos Policy set and binding to the Web service client

To attach the custom policy set to the Web service client, follow these steps:

1. In the left pane of the administrative console, click **Services** → **Service clients**.
2. Click **WeatherJavaBeanService**.
3. Select **WeatherJavaBeanService**, and click **Attach Client Policy Set**. Select **ITSO Kerberos** from the drop-down list. Click **Save**.
4. Select **WeatherJavaBeanService** again, and click **Assign Binding**. Select **New Application Specific Binding**.
5. Enter `ITSO client binding` as the Bindings configuration name.
6. Click **Add**, and select **WS-Security**.
7. Click **Authentication and protection**.
8. In the Authentication tokens section, click **request:GSS_Kerberosv5_AP_REQ**. Then, confirm the following information:
   - Make sure that the `wss.generate.KRB5BST` value is selected from the JAAS login menu list.
   - In the Custom properties section, specify the token generator custom properties for the target service name, host, and the optional realm. Table 10-1 lists the target service custom properties.

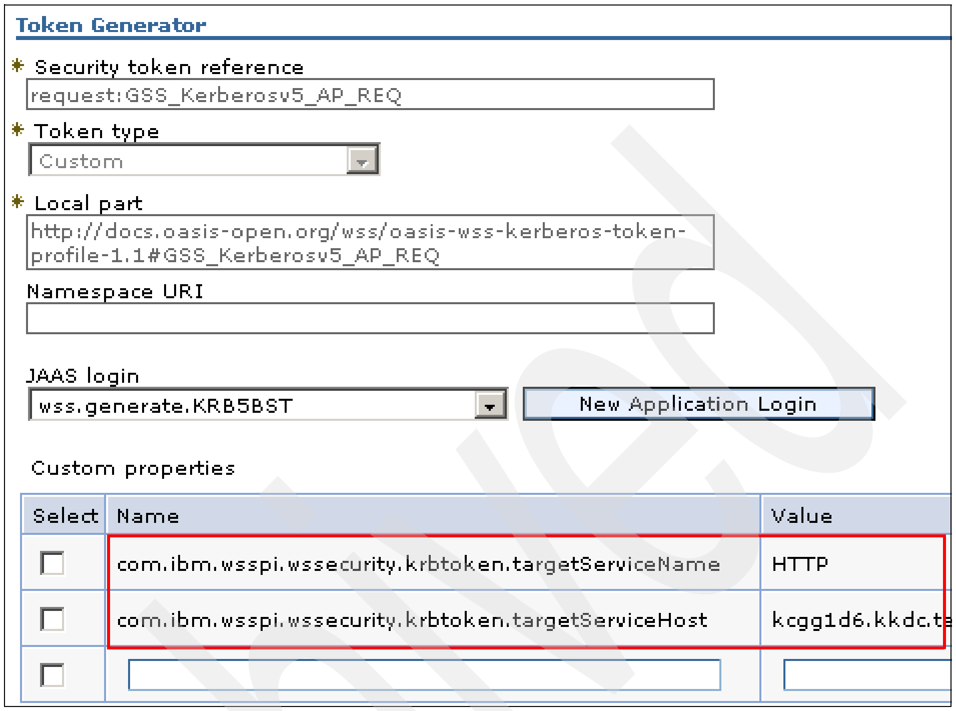
Table 10-1 Target service custom properties

Name	Value	Type
`com.ibm.wsspi.wssecurity.krbtoken.targetServiceName`	Specify the name of the target service	Required
`com.ibm.wsspi.wssecurity.krbtoken.targetServiceHost`	Specify the host name that is associated with the target service in the following format: `myhost.mycompany.com`	Required
`com.ibm.wsspi.wssecurity.krbtoken.targetServiceRealm`	Specify the name of the realm that is associated with the target service	Optional[a]

a. If the `targetServiceRealm` property is not specified, the default realm name from the Kerberos configuration file is used as the realm name. To use Kerberos token security in a cross or trusted realm environment, you must provide a value for the `targetServiceRealm` property.

The combination of the target service name and host values forms the SPN, which represents the target Kerberos SPN. The Kerberos client requests the initial Kerberos AP_REQ token for the SPN.

Enter the values as shown in Figure 10-7, and click **Apply**.



Token Generator

* Security token reference
request:GSS_Kerberosv5_AP_REQ

* Token type
Custom

* Local part
http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-1.1#GSS_Kerberosv5_AP_REQ

Namespace URI

JAAS login
wss.generate.KRB5BST

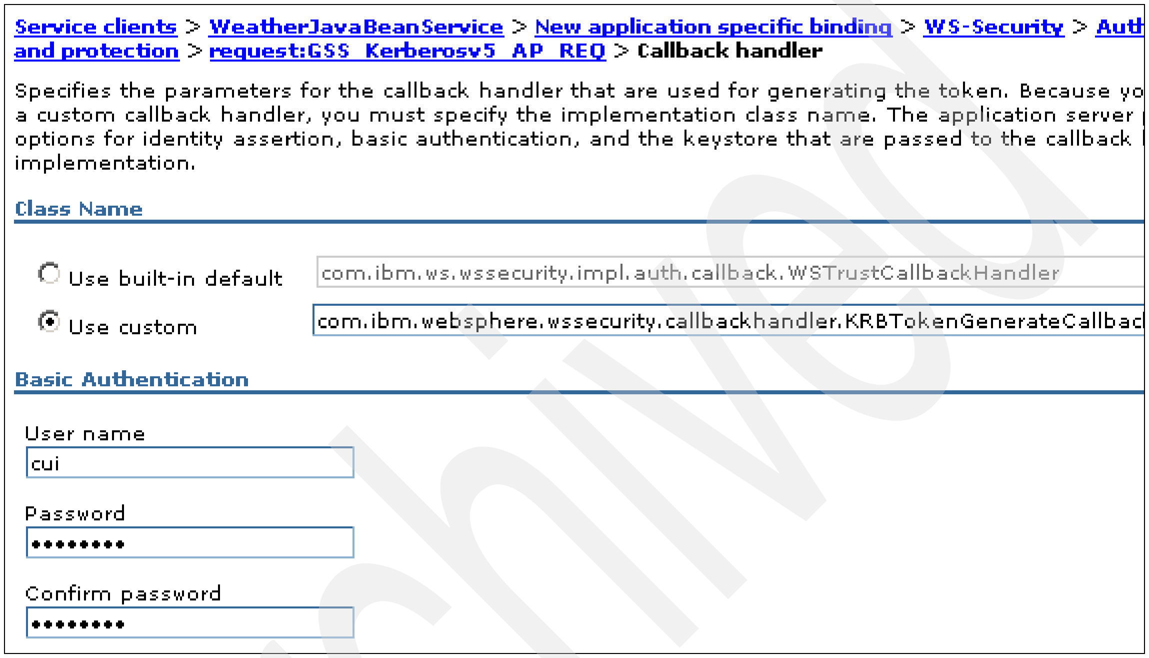
Custom properties

Select	Name	Value
☐	com.ibm.wsspi.wssecurity.krbtoken.targetServiceName	HTTP
☐	com.ibm.wsspi.wssecurity.krbtoken.targetServiceHost	kcgg1d6.kkdc.t
☐		

Figure 10-7 Custom properties

9. Click **Callback handler**. From the Basic Authentication heading, specify the appropriate values for the User name, Password, and Confirm password fields, as shown in Figure 10-8.

   The user name specifies the default user ID that is passed to the constructor of the callback handler. The user name must be a valid user in the Kerberos domain.



Service clients > WeatherJavaBeanService > New application specific binding > WS-Security > Auth and protection > request:GSS_Kerberosv5_AP_REQ > Callback handler

Specifies the parameters for the callback handler that are used for generating the token. Because yo a custom callback handler, you must specify the implementation class name. The application server options for identity assertion, basic authentication, and the keystore that are passed to the callback implementation.

Class Name

○ Use built-in default: com.ibm.ws.wssecurity.impl.auth.callback.WSTrustCallbackHandler

[✓ Use custom]: com.ibm.websphere.wssecurity.callbackhandler.KRBTokenGenerateCallbac

Basic Authentication

User name: cui

Password: ••••••••

Confirm password: ••••••••

Figure 10-8 Callback handler configuration

10. Click **OK**, then click **Save**.

10.4 Validating the scenario by monitoring the SOAP traffic

If you test the application using the instructions in 10.4.1, "Monitoring the SOAP traffic" on page 233. When the configuration completes successfully completed, the Kerberos token is included in the SOAP request, as shown in Example 10-4.

Example 10-4 SOAP request with the Kerberos token

```
<soapenv:Header>
    <wsse:Security soapenv:mustUnderstand="1"
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext
-1.0.xsd">
      <wsse:BinarySecurityToken
ValueType="http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-1.1#GSS_Ke
rberosv5_AP_REQ" Id="krb5_20"
xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility
-1.0.xsd">YIII9wYJKoZIh ... ... lDzuBOymEpF1JWUdw==</wsse:BinarySecurityToken>
    </wsse:Security>
  </soapenv:Header>
  <soapenv:Body>
    <ns2:getDayForecast xmlns:ns2="http://bean.itso/">
      <arg0>2009-05-06T16:57:39</arg0>
    </ns2:getDayForecast>
  </soapenv:Body>
</soapenv:Envelope>
```

10.4.1 Monitoring the SOAP traffic

Rational Application Developer provides a TCP/IP monitor that you use to monitor SOAP traffic over HTTP. The TCP/IP monitor acts as an intermediary between a Web service and its client. The client calls the TCP/IP address of the monitor rather than the SOAP endpoint. The monitor is configured to forward the request to the endpoint, and if it is a two-way request or reply, return the response to the start point, which is the client unless a different endpoint for the reply is used.

To monitor the SOAP traffic:

1. Start Rational Application Developer if it is not already running.
2. Select **Window** → **Preferences** → **Run/Debug** → **TCP/IP Monitor**, and click **Add**. The New Monitor window opens.

Alternatively, on the Servers tab, right-click the server name, and select **Monitoring** → **Properties** → **Add**. Then, the Monitoring Ports window opens.

Although the windows are different, you can achieve the same result with either one. In this example, we use the New Monitor window.

3. In the New Monitor window, shown in Figure 10-9, complete the following steps:

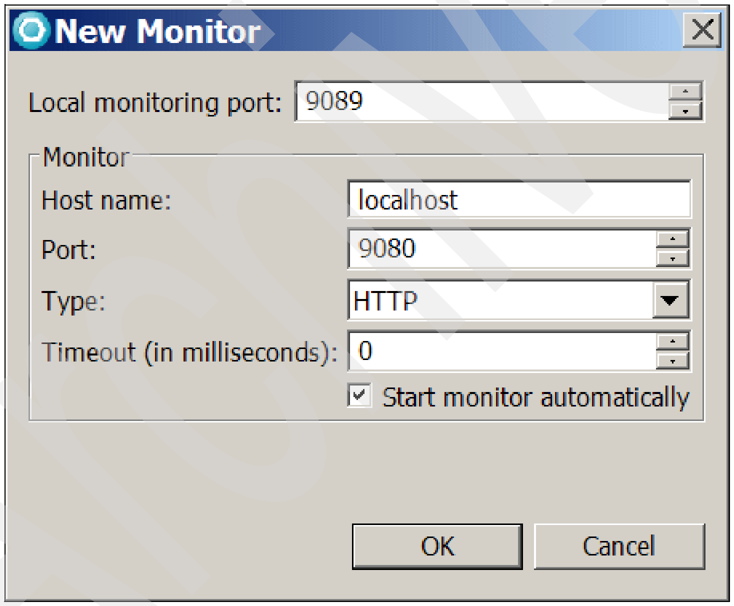
   a. In the “Local monitoring port” field, specify a unique port number on the local system that is not used by any process (for example, `9089`).

   b. In the Host name field, type `localhost`.

   c. For Port, specify the port number of your WebSphere Application Server V7 for Web services provider.

   d. Select **Start monitor automatically**.

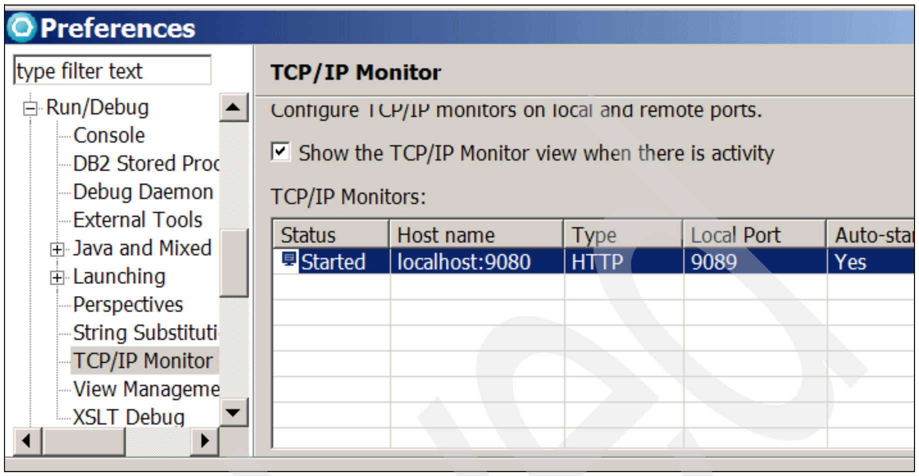
   e. Click **OK**.



Local monitoring port: 9089
Host name: localhost
Port: 9080
Type: HTTP
Timeout (in milliseconds): 0
[✓] Start monitor automatically

Figure 10-9 Creating a new TCP/IP monitor

4. Click **OK** again to close the preferences window. The TCP/IP Monitor starts as shown in Figure 10-10.


Preferences

type filter text

- Run/Debug
  - Console
  - DB2 Stored Proc
  - Debug Daemon
  - External Tools
  - Java and Mixed
  - Launching
  - Perspectives
  - String Substituti
  - TCP/IP Monitor
  - View Manageme
  - XSLT Debug

TCP/IP Monitor

Configure TCP/IP monitors on local and remote ports.

[✓] Show the TCP/IP Monitor view when there is activity

TCP/IP Monitors:

Status	Host name	Type	Local Port	Auto-sta
Started	localhost:9080	HTTP	9089	Yes

Figure 10-10 TCP/IP monitor started

5. Open a Web browser and run the sample JSP client. For example, if your server is running at port 9080, enter the following URL:

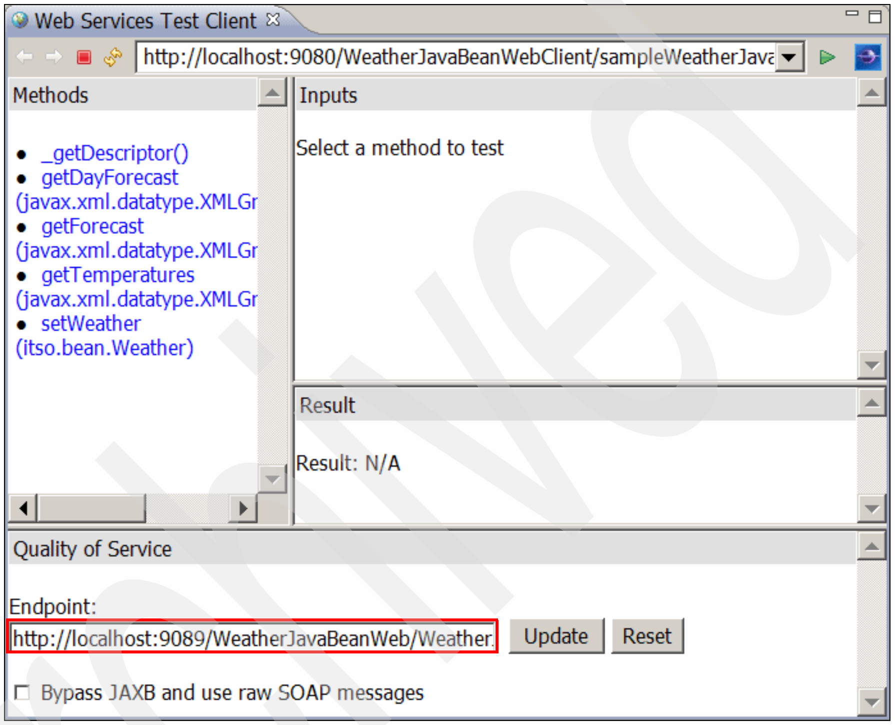
```
http://localhost:9080/WeatherJavaBeanWebClient/sampleWeatherJavaB
eanPortProxy/TestClient.jsp
```

6. When the sample Web service JSP client opens in the browser, in the bottom pane, change the endpoint from `9080` to `9089`. This value points to the port to the TCP/IP monitor that is listening.

As shown on the Web service Test Client page in Figure 10-11, in the Quality of Service pane, the URL is as follows:

```
http://localhost:9089/WeatherJavaBeanWebClient/sampleWeatherJavaB
eanPortProxy/TestClient.jsp
```

Click **Update**.



Methods

- _getDescriptor()
- getDayForecast (javax.xml.datatype.XMLGr
- getForecast (javax.xml.datatype.XMLGr
- getTemperatures (javax.xml.datatype.XMLGr
- setWeather (itso.bean.Weather)

Select a method to test

Result: N/A

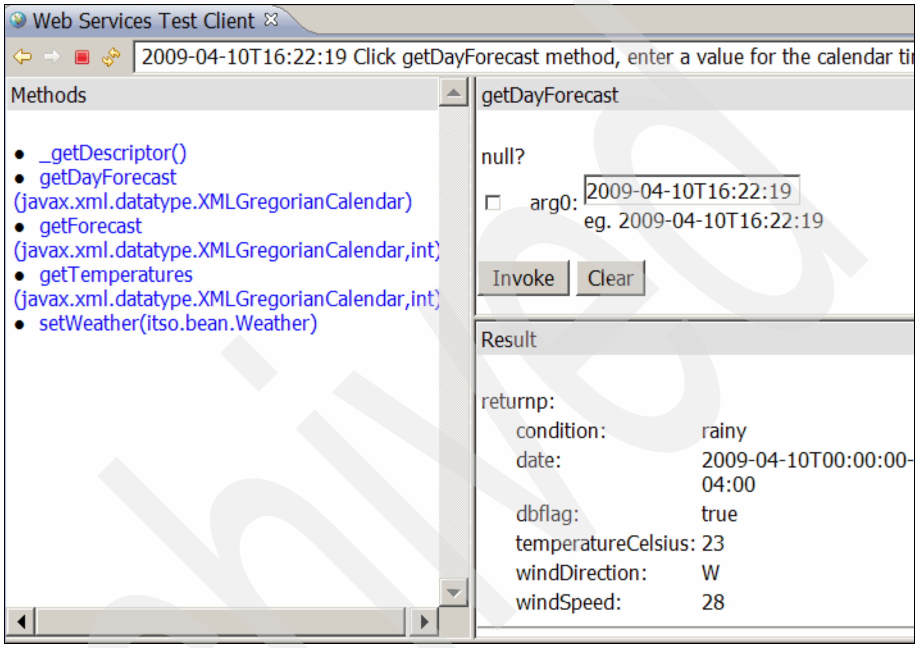
Endpoint: http://localhost:9089/WeatherJavaBeanWeb/Weather

☐ Bypass JAXB and use raw SOAP messages

Figure 10-11 Updating the Web service endpoint

7. On the Web service Test Client page, in the Methods pane, click the **getDayForecast** method. In the upper, right pane, enter a value for arg0. You can copy and paste the example value that is provided by the JSP client. In this case, we use `2009-04-10T16:22:19`. Click **Invoke**.

   Figure 10-12 shows the result.



2009-04-10T16:22:19 Click getDayForecast method, enter a value for the calendar ti

- _getDescriptor()
- getDayForecast (javax.xml.datatype.XMLGregorianCalendar)
- getForecast (javax.xml.datatype.XMLGregorianCalendar,int)
- getTemperatures (javax.xml.datatype.XMLGregorianCalendar,int)
- setWeather(itso.bean.Weather)

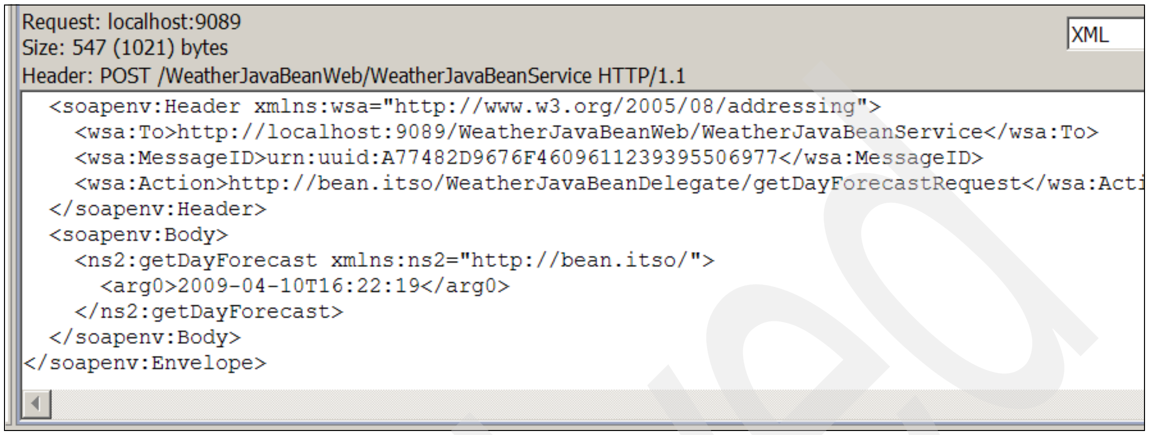
null?

arg0: 2009-04-10T16:22:19
eg. 2009-04-10T16:22:19

```
returnp:
    condition:          rainy
    date:               2009-04-10T00:00:00-
                        04:00
    dbflag:             true
    temperatureCelsius: 23
    windDirection:      W
    windSpeed:          28
```

Figure 10-12 Web services result

8. In the TCP/IP Monitor view, which opens automatically, in the upper-right corner, select **XML** for the request and response to view its contents (Figure 10-13).



```
Request: localhost:9089
Size: 547 (1021) bytes
Header: POST /WeatherJavaBeanWeb/WeatherJavaBeanService HTTP/1.1
  <soapenv:Header xmlns:wsa="http://www.w3.org/2005/08/addressing">
    <wsa:To>http://localhost:9089/WeatherJavaBeanWeb/WeatherJavaBeanService</wsa:To>
    <wsa:MessageID>urn:uuid:A77482D9676F4609611239395506977</wsa:MessageID>
    <wsa:Action>http://bean.itso/WeatherJavaBeanDelegate/getDayForecastRequest</wsa:Acti
  </soapenv:Header>
  <soapenv:Body>
    <ns2:getDayForecast xmlns:ns2="http://bean.itso/">
      <arg0>2009-04-10T16:22:19</arg0>
    </ns2:getDayForecast>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 10-13 Using TCP/IP Monitor to watch the SOAP traffic

Look at the WS-Addressing information in the SOAP request in Figure 10-13:

- The `<wsa:to>` information specifies the destination of the SOAP message.
- The `<wsa:MessageID>` information is the unique identifier for the SOAP message
- The `<wsa:Action>` information is the in-envelope version of the SOAP HTTP Action header.

This information shows that the WS-Addressing policy set is applied successfully to the Weather Web service and its client.

10.4.2 Removing policy sets

In the event that you need to define policy sets, use the following procedures to restore an application to the original state.

If you are using a stand-alone server, to detach the WS-Addressing policy, follow these steps:

1. Detach the service provider's policy set as follows:
   a. In the administrative console, expand **Services** → **Service providers**.
   b. Click **WeatherJavaBeanService**.
   c. Select **WeatherJavaBeanService**.
   d. Click **Detach Policy Set**.
   e. Click **Save**.
2. Detach the service client's policy set as follows:
   a. Expand **Services** → **Service clients**.
   b. Click **WeatherJavaBeanService**.
   c. Select **WeatherJavaBeanService**.
   d. Click **Detach Client Policy Set**.
   e. Click **Save**.
3. Restart the applications:
   a. In the navigation pane, select **Applications** → **Application Types** → **WebSphere Enterprise Applications**.
   b. Select **WeatherJavaBeanServer** and **WeatherJavaBeanWebClientEAR**.
   c. Click **Stop**.
   d. After the applications stop, click **Start**.

If you are using Rational Application Developer and its integrated WebSphere Application Server, follow these steps:

1. Right-click **WebSphere Application Server V7.0**, and select **Add and Remove Projects**.
2. In the Add and Remove Projects window, select **Remove All** to uninstall the Web service and the client.

   Note that after you uninstall the Web service and the client application, the specific policy set and the binding that are attached with the Web service application are also gone.
3. Right-click **WebSphere Application Server V7.0**, and select **Add and Remove Projects**.
4. In the Add and Remove Projects window, select **Add All** to install the Web service and the client.

You now have a fresh Web service and client without the policy set and binding attached.

11

WS-SecurityKerberos with a .NET Web services client

This chapter describes a scenario in which a Microsoft Windows Foundation Framework (WCF) client sends a WS-SecurityKerberos request to a JAX-WS J2EE provider service using Microsoft Active Directory as the user registry. It includes the following topics:

- Scenario overview
- Configuring the KDC and client system
- Configuring WebSphere Application Server
- Configuring the .NET client system
- Configuring the provider application
- Validating the scenario

11.1 Scenario overview

This scenario illustrates how to configure a Kerberos environment to secure SOAP messages that are exchanged between WebSphere Application Server V7.0.0.5 and Windows Communication Foundation (WCF) V3.5.

The scenario included in this chapter provides an example in which a WCF console application (.NET client) communicates with a JAX-WS J2EE provider service using WS-SecurityKerberos. This scenario illustrates the changes that are required for WebSphere Application Server to interoperate with a .NET client. The .NET client and the J2EE provider service share the same Microsoft Active Directory for their user registry.

Before you can use Kerberos with Web services security (WSS), you must configure Kerberos in WebSphere Application Server. You do not need to enable Kerberos as the authentication mechanism. However, the Kerberos configuration file, `krb5.conf` or `krb5.ini`, and the Kerberos keytab file, `krb5.keytab`, are required.

Note: This scenario features the following components:

- Access to a Web service provider on WebSphere Application Server from a .NET client
- WS-SecurityKerberos to secure Web services traffic at the message level
- Policy sets to specify Kerberos authentication
- Microsoft Active Directory and Kerberos key distribution center (KDC)
- .NET client application: WSWindowsClient WCF sample application
- WebSphere Application Server configuration:
  - User registry: Microsoft Active Directory
  - Authentication mechanism: LTPA
  - Kerberos authentication and configuration file
  - Web service provider application: EchoService of JAXWSServicesSamples

11.1.1 Step-by-step flow of this scenario

Figure 11-1 illustrates the basic flow of this scenario.

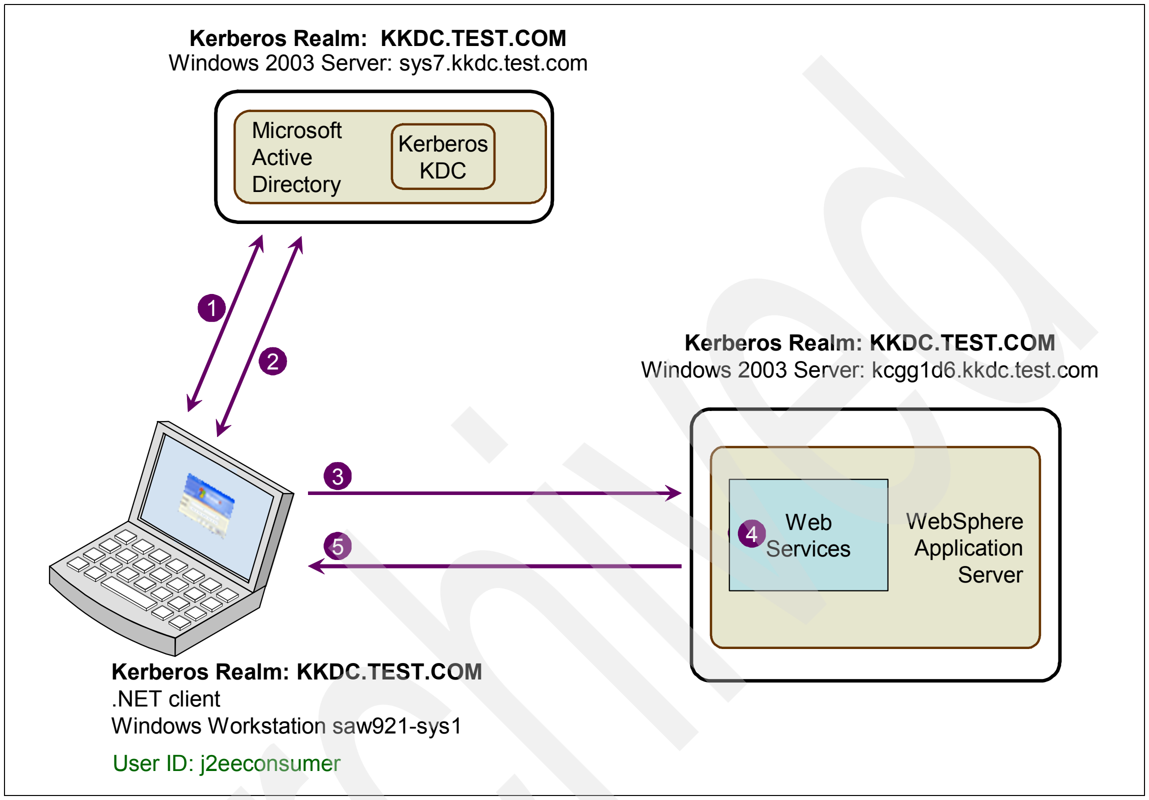

Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: sys7.kkdc.test.com
Microsoft Active Directory
Kerberos KDC
1
2
Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: kcgg1d6.kkdc.test.com
3
4
Web Services
WebSphere Application Server
5
Kerberos Realm: KKDC.TEST.COM
.NET client
Windows Workstation saw921-sys1
User ID: j2eeconsumer

Figure 11-1 A .NET client using Kerberos authentication to access J2EE provider service

The numbers in the figure correspond to the following steps in this process:

1. To begin, the user logs on to the Windows domain from the workstation. After the user enters a user name and password, Windows logon sends the information to the workstation local security authority, which passes the request to the Kerberos authentication package.

   The client sends an initial authentication request (AS_REQ), which includes the user's credentials and an encrypted time stamp, to the KDC. This request is a request for authentication and a ticket-granting ticket (TGT).

   The first domain controller in the domain generates the `krbtgt/`*`domain_name`* ticket. The KDC uses the secret key to decrypt the time stamp and issues a TGT to the client.

The AS_REP response is encrypted with the user's key and is returned to the user. The ticket is encrypted with the KDC's key and is enclosed in the AS_REP.

2. The Web service client sends a request (using the TGT) for a service ticket to the ticket-granting server for communication with the Web services. The ticket-granting server then returns a service ticket (TGS_REP) and the session key to the client. The client uses this information to create the Kerberos token.
3. After the Kerberos token is created, the WSS generator encodes and inserts the token into the SOAP message and propagates the token for token consumption or acceptance.
4. The Web service provider receives and extracts the Kerberos token from the SOAP request. The provider then accepts the Kerberos token by validating the token with its own secret key. The secret key of the service is stored in a keytab file. After acceptance, the Web services provider stores the associated request token information into the context Subject.
5. The provider return a SOAP response.

11.1.2 Scenario components

The scenario in this chapter uses the following components:

- Microsoft .NET Framework Version 3.5
- The application server

  WebSphere Application Server V7.0.0.5 installed on Windows XP Professional 2002 SP2
- The Kerberos KDC

  Microsoft Windows Server 2003 SP2 with Microsoft Active Directory. The Windows 2003 Support Tools are installed.

Download materials: A sample WCF console application (.NET client) is included in the `WCFSample.zip` file. For information about how to download the file, see Appendix E, "Additional material" on page 525.

11.1.3 Basic requirements for this scenario

This scenario requires the following basic components:

- A working domain controller and at least one client computer in that domain

  Using SPNEGO from the domain controller does not work.
- Microsoft .NET Framework 3.5 installed on the .NET client system
- Windows SDK for Windows Server 2008 installed on the .NET client system

  This SDK contains the `svcconfigeditor` utility that is used to edit the WCF client and service custom bindings.
- The sample WCF console application (.NET client), which is included in the `WCFSample.zip` file in the additional material for this book, downloaded to the system where Microsoft .NET Framework 3.5 is installed

  For more information about the additional material for this book, see Appendix E, “Additional material” on page 525.
- A functioning Microsoft Windows 2000 or 2003 Active Directory Domain that includes the following components:
  – A domain controller
  – A client workstation

  Users must be able to log in to the client workstation.
- WebSphere Application Server for Windows installed and application security enabled for the profile

  The profile must be created after WebSphere Application Server V7.0.0.1 is installed (`kcgg1d6.kkdc.test.com`).

  Important: WebSphere Application Server Version V7.0.0.1 (or later) bundles new general bindings, policy sets, and run times that are used for this scenario. The WebSphere Application Server fix pack installation process does not import the new Kerberos general bindings and policy set that are required for this scenario to an existing WebSphere Application Server profile. Thus, you need to create a new WebSphere Application Server profile after installing WebSphere Application Server V7.0.0.1 (or later) to ensure that you can use new bindings and policy sets that are shipped with WebSphere Application Server.

- Access for users from the Microsoft Active Directory to WebSphere Application Server’s protected resources using a native authentication mechanism, such as basic authentication (BA) or forms authentication

- Windows 200x Support Tools on the Microsoft Windows Server, including the **`setspn`**, the **`ktpass`**, and the **`adsiedit`** tools that this scenario uses

  These support tools are available on the Windows Server installation CD or from the following Microsoft Web site:

  `http://www.microsoft.com/downloads/details.aspx?familyid=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D`

We also recommend that you install the Windows Server 2003 Resource Kit Tools, which provide the **`kerbtray.exe`** utility to see Kerberos tickets on the user workstation. These tools are available at:

`http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd`

11.2 Configuring the KDC and client system

To configure the KDC and client system, first complete these steps:

1. Set up a Windows Server with Microsoft Active Directory domain and associated Kerberos KDC. The Windows administrator needs to complete this step.
2. Join the .NET client system and the Microsoft Active Directory domain controller. Right-click the **My Computer** icon. Then, select **Properties**. In the System Properties dialog box, go to the Computer Name tab, and click **Change**. Enter your Microsoft Active Directory domain as the system name. For example, in our scenario, computer `saw921-sys1` joined the Microsoft Active Directory domain `kkdc.test.com`.

11.2.1 Creating the SPN for WebSphere Application Server

Next, you need to create a user account for WebSphere Application Server in the Microsoft Active Directory. This account is mapped to the Kerberos service principal name (SPN). The SPN identifies the WebSphere Application Server installation as a Kerberized service. To create this SPN for WebSphere Application Server, follow these steps:

1. Log on as an administrator to the system that serves as the domain controller.
2. Invoke the Microsoft Active Directory users and computers snap-in. Click **Start** → **Run**. Then type the following command, and click **OK**:

   ```
   dsa.msc /server=localhost
   ```

3. Create a user account for WebSphere Application Server. This scenario creates the *j2eeprovider* user account.

11.2.2 Creating the keytab file

Now, you can create the Kerberos keytab file. Use the **ktpass** tool to set up the Kerberos keytab file (`j2eeProvider.keytab`) for the previously created SPN. The **ktpass** command allows you to configure a non-Windows Server 2003 Kerberos service (such as WebSphere applications on a z/OS platform) as a security principal in the Windows Server 2003 Active Directory. This tool generates a keytab file containing the shared key of the service. In our scenario, we use the following command shown in Example 11-1.

Example 11-1 Example output from ktpass utility

```
C:\Documents and Settings\Administrator>ktpass.exe
-out c:\j2eeProvider.keytab
-princ echoSrvProvider/kcgg1d6.kkdc.test.com@KKDC.TEST.COM
-pass itso4you -mapUser j2eeprovider -mapOp set
-ptype KRB5_NT_PRINCIPAL -crypto RC4-HMAC-NT
Targeting domain controller: sys7.kkdc.test.com
Using legacy password setting method
Successfully mapped echoSrvProvider/kcgg1d6.kkdc.test.com to
j2eeprovider.
Key created.
Output keytab to c:\j2eeProvider.keytab:
Keytab version: 0x502
keysize 86 echoSrvProvider/kcgg1d6.kkdc.test.com@KKDC.TEST.COM ptype 1
(KRB5_NT_
PRINCIPAL) vno 4 etype 0x17 (RC4-HMAC) keylength 16
(0x31a1f13010baa0c4f9fc8ead6
de09db5)

C:\Documents and Settings\Administrator>setspn -L j2eeprovider
Registered ServicePrincipalNames for
CN=j2eeprovider,CN=Users,DC=kkdc,DC=test,DC
=com:
    echoSrvProvider/kcgg1d6.kkdc.test.com
```

11.3 Configuring WebSphere Application Server

To configure the WebSphere Application Server for this scenario, you need to complete the following tasks:

- Creating the Kerberos configuration file
- Configuring the user registry

11.3.1 Creating the Kerberos configuration file

Create the Kerberos configuration file in the system that hosts WebSphere Application Server (`kcgg1d6.kkdc.test.com`). Use the **`wsadmin`** utility to generate the Kerberos configuration file as follows:

1. Copy the keytab file (`j2eeProvider.keytab`) from the KDC system to the WebSphere Application Server system. This scenario copies the file to the `c:\windows` folder.
2. Start the **`wsadmin`** command from the *`app_server_root`*`/bin` folder. Use **`wsadmin`** to create the associated Kerberos configuration as shown in Example 11-2.

Example 11-2 Generating the krb5.ini file using wsadmin

```
wsadmin>$AdminTask createKrbConfigFile {-krbPath c:/WINDOWS/krb5.ini
-realm KKDC.TEST.COM -kdcHost sys7.kkdc.test.com -dns kkdc.test.com
-keytabPath c:/WINDOWS/j2eeProvider.keytab}
```

Example 11-3 shows the Kerberos configuration file used for this scenario.

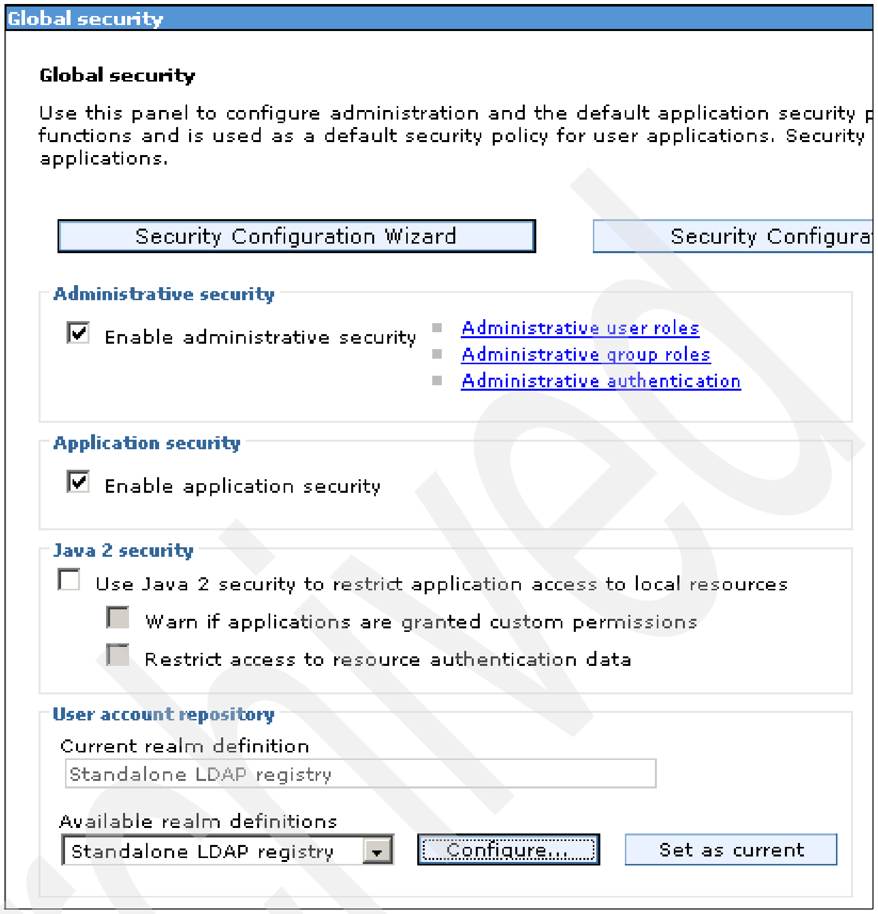
Example 11-3 Kerberos configuration file used for the scenario

```
[libdefaults]
   default_realm = KKDC.TEST.COM
   default_keytab_name = FILE:c:/windows/J2EEProvider.keytab
   default_tkt_enctypes = rc4-hmac des-cbc-md5
   default_tgs_enctypes = rc4-hmac des-cbc-md5
   forwardable  = true
   renewable  = true
   noaddresses = true
   clockskew  = 300
[realms]
   KKDC.TEST.COM = {
      kdc = sys7.kkdc.test.com:88
      default_domain = kkdc.test.com
   }
[domain_realm]
   .kkdc.test.com = KKDC.TEST.COM
```

11.3.2 Configuring the user registry

Configure WebSphere Application Server global security to connect to the Microsoft Active Directory as follows:

1. Log on to the WebSphere Application Server administrative console, and navigate to **Security** → **Global security**. In the Global security panel, shown in Figure 11-2, complete these steps:
   a. Select the “Enable administrative security” and “Enable application security” options.
   b. Clear the “Use Java 2 security to restrict application access to local resources” option.
   c. Select **Standalone LDAP registry** in the Available realm definitions section, and click **Configure**.


Global security

Global security

Use this panel to configure administration and the default application security p
functions and is used as a default security policy for user applications. Security
applications.

Administrative security

[✓] Enable administrative security

- Administrative user roles
- Administrative group roles
- Administrative authentication

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

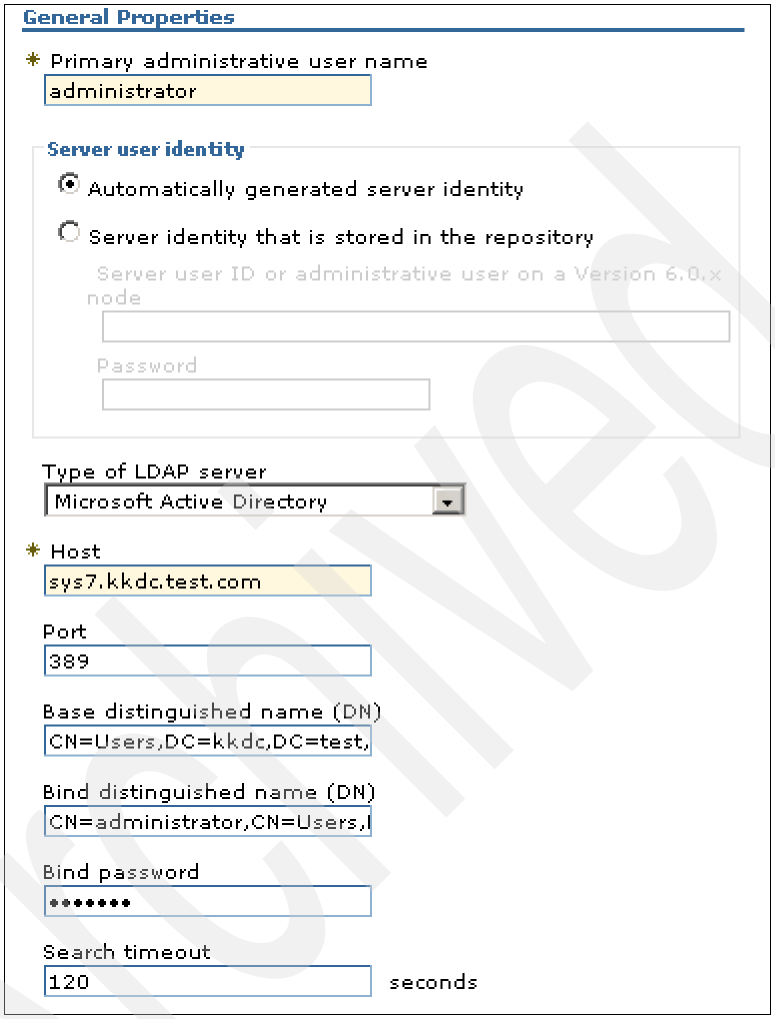
Standalone LDAP registry

Available realm definitions

Standalone LDAP registry

Figure 11-2 Set Global security

2. Enter the active directory information, as shown in Figure 11-3, and click **OK.**


General Properties

* Primary administrative user name: administrator

Server user identity

(●) Automatically generated server identity

( ) Server identity that is stored in the repository

Server user ID or administrative user on a Version 6.0.x node:

Password:

Type of LDAP server: Microsoft Active Directory

* Host: sys7.kkdc.test.com

Port: 389

Base distinguished name (DN): CN=Users,DC=kkdc,DC=test,

Bind distinguished name (DN): CN=administrator,CN=Users,I

Bind password: ●●●●●●●

Search timeout: 120 seconds

Figure 11-3 Active Directory configuration

3. Click **Set as Current**.
4. Click **Save** to save changes to the master repository.

11.4 Configuring the .NET client system

Follow these steps to configure the .NET client system:

1. Log on to the Microsoft Windows Active Directory domain with an identity known in Active Directory. In our scenario, we log on to the Microsoft Active Directory domain with the user name `j2eeconsumer`.
2. Make sure that you can connect to the Kerberos KDC. Use the **`klist`** tool that ships with the Windows Resource Toolkit to test the connection, as shown in Example 11-4.

Example 11-4 Use klist to test the connection

```
C:\Documents and Settings\j2eeconsumer>klist tgt

Cached TGT:

ServiceName: krbtgt
TargetName: krbtgt
FullServiceName: j2eeconsumer
DomainName: KKDC.TEST.COM
TargetDomainName: KKDC.TEST.COM
AltTargetDomainName: KKDC.TEST.COM
TicketFlags: 0x40e00000
KeyExpirationTime: 0/39/4 0:00:10776
StartTime: 7/22/2009 16:46:43
EndTime: 7/23/2009 2:46:43
RenewUntil: 7/29/2009 16:46:43
TimeSkew: 7/29/2009 16:46:43
```

11.4.1 Configuring Microsoft .NET Framework 3.5

In this scenario, Microsoft .NET Framework 3.5 is used to run the .NET client application. You can download and install Microsoft .NET Framework from the following Microsoft Web site:

`http://www.microsoft.com/downloads/details.aspx?FamilyID=333325fd-ae52-4e35-b531-508d977d32a6&DisplayLang=en`

Windows SDK for Windows Server 2008 and Microsoft .NET Framework 3.5 includes utilities, such as `svcconfigeditor`, that can use edit .NET provider applications. You can download and install Windows SDK from the following Microsoft Web site:

`http://www.microsoft.com/downloads/details.aspx?FamilyId=F26B1AA4-741A-433A-9BE5-FA919850BDBF&displaylang=en`

11.4.2 Configuring the bindings for the .NET client application

The .NET client application is configured to interoperate with the J2EE provider application. The sample .NET client application is based on the same WSDL as the J2EE provider service (EchoService of JaxWSServicesSamples).

Download material: The additional material for this book includes a sample WCF console application (.NET client) application. You can download the compressed file, `WCFSample.zip`, and extract the files to the `C:\` directory. For information about how to download this file, see Appendix E, “Additional material” on page 525.

The .NET client application configuration already has a custom binding called *EchoSOAP*, which has the `httptransport` and `textMessageEncoding` (`Soap11WSAddressing10`) extensions enabled. The Microsoft Service Configuration Editor is used to update this binding to include WSS. Perform the following steps to add a security binding extension and to configure the security settings to match the J2EE provider application configuration:

1. Select **All Programs** → **Microsoft Windows SDK v6.1** → **Tools** → **Service Configuration Editor**.
2. Select **File** → **Open** → **Config File**.
3. Select the configuration file that ships with the samples, `C:\IBM\WCFClient\WSWindowsClient.exe.config`, and click **Open.**
4. In the left pane, select **Bindings** → **EchoSOAP(customBinding)**.
5. In the right pane, click **Add** to open the Adding Binding Element Extensions dialog box. Then, select **Security**, and click **Add**.
6. In the left pane, select **Security**, and change the following settings in the **General** tab in the right pane:
   a. Set AuthenticationMode to `Kerberos`.
   b. Set Default Algorithm to `Basic128Rsa15`.
   c. Use the default value SignBeforeEncryptAndEncryptSignature for MessageProtectionOrder.

   d. Set MessageSecurityVersion to `WSSecurity11WSTrust13WSSecureConversation13WSSecurityPolicy12BasicSecurityProfile10`.
   e. Set the default value `True` for requireDerivedKeys.
   f. Set the default value `True` for equireSecurityContextCancellation.
   g. Set requireSignatureConfirmation to `True`.
   h. Set default value `Strict` for securityHeaderLayout.
7. In the left pane, select **Client** → **Endpoints** → **EchoServicePort**.
8. In the right pane, set the Address to:

   ```
   http://kcgg1d6.kkdc.test.com:9080/WSSampleSei/EchoService
   ```
9. Switch to the Identity tab, and set the ServicePrincipalName to:

   ```
   echoSrvProvider/kcgg1d6.kkdc.test.com
   ```
10. Select **File** → **Save** to save the changes to WSWindowsClient.exe.config.

11.5 Configuring the provider application

This section provides information about deploying the provider application and configuring the policy sets.

11.5.1 Deploying the J2EE provider application

The Service Endpoint Interface (SEI) sample application `JaxWSServicesSamples.ear` is located in the following directory:

```
app_server_root\samples\lib\JaxWSServicesSamples
```

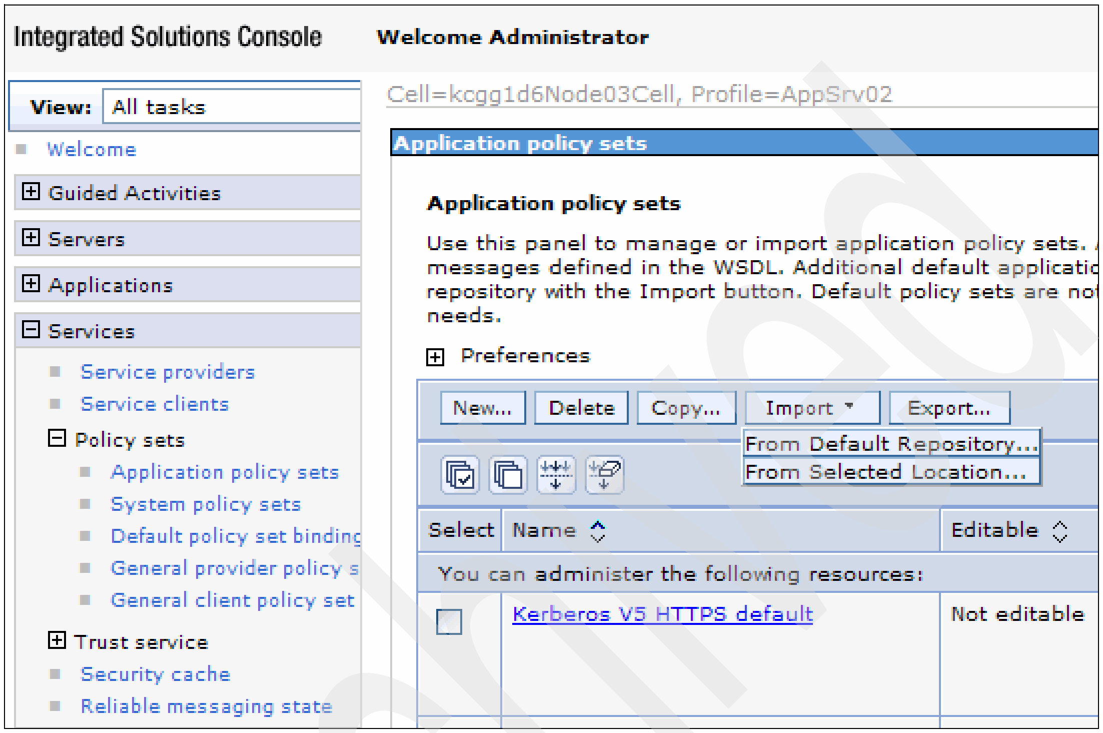
Deploy the application `JaxWSServicesSamples.ear` using the WebSphere Application Server administrative console.

11.5.2 Configuring the policy set for the Web service

Perform the following steps to create a custom policy set, using the Kerberos V5 WS-Security default policy set as the base:

1. Log in to the WebSphere Application Server administrative console.
2. Click **Services** in the left navigation bar.
3. Expand the **Policy sets**, and click **Application policy sets** to launch the Application policy sets panel.

4. Click **Import** from the collection table, and select **From Default Repository** from the drop-down menu, as shown in Figure 11-4, to launch the Import panel.



Cell=kcgg1d6Node03Cell, Profile=AppSrv02

Application policy sets

Application policy sets

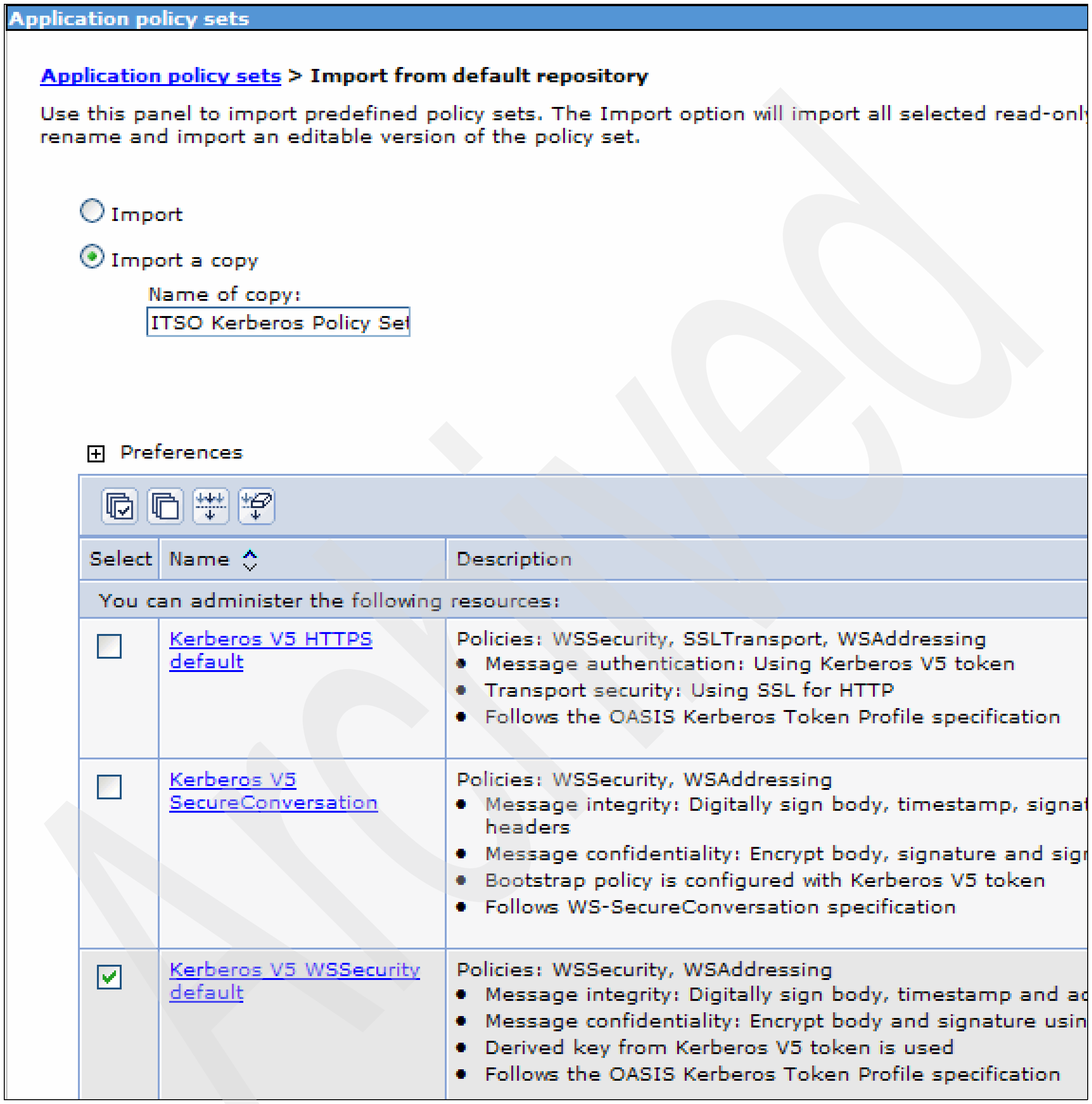
Use this panel to manage or import application policy sets.
messages defined in the WSDL. Additional default applicatio
repository with the Import button. Default policy sets are no
needs.

Select	Name	Editable
You can administer the following resources:		
☐	Kerberos V5 HTTPS default	Not editable

Figure 11-4 Create Application policy sets

5. Select the “Import a copy radio button” option, and specify `ITSO Kerberos Policy Set` as the new name for the copy.

6. Select **Kerberos V5 WSSecurity default** as shown in Figure 11-5, and then click **OK** to create the copy.



Application policy sets

Application policy sets > **Import from default repository**

Use this panel to import predefined policy sets. The Import option will import all selected read-onl rename and import an editable version of the policy set.

Import [✓ Import a copy]

Name of copy: ITSO Kerberos Policy Set

Preferences

Select	Name	Description
You can administer the following resources:		
☐	Kerberos V5 HTTPS default	Policies: WSSecurity, SSLTransport, WSAddressing • Message authentication: Using Kerberos V5 token • Transport security: Using SSL for HTTP • Follows the OASIS Kerberos Token Profile specification
☐	Kerberos V5 SecureConversation	Policies: WSSecurity, WSAddressing • Message integrity: Digitally sign body, timestamp, signat headers • Message confidentiality: Encrypt body, signature and sign • Bootstrap policy is configured with Kerberos V5 token • Follows WS-SecureConversation specification
✓	Kerberos V5 WSSecurity default	Policies: WSSecurity, WSAddressing • Message integrity: Digitally sign body, timestamp and ad • Message confidentiality: Encrypt body and signature usin • Derived key from Kerberos V5 token is used • Follows the OASIS Kerberos Token Profile specification

Figure 11-5 Import from default repository

Attention: WebSphere Application Server V7.0.0.1 ships several default policy sets that are preconfigured for certain common scenarios. As mentioned in 11.1.3, “Basic requirements for this scenario” on page 245, you need to use a profile that was created after WebSphere Application Server V7.0.0.1 (or later) is installed. If not, you will *not* have the option to import the predefined Kerberos V5 WSSecurity default policy set.

7. Save the changes.

There are no additional changes required for the ITSO Kerberos Policy Set.

Perform the following steps to review this newly created policy set to get an understanding of its requirements:

1. From the WebSphere Application Server console, select **Services** → **PolicySets** → **Application policy sets**.
2. Select **ITSO Kerberos Policy Set**.

   Note that the WS-Security and WS-Addressing policies are enabled.
3. Select **Security** → **Main policy**.
4. In the Main policy dialog box, shown in Figure 11-6 on page 258, configure the following settings:
   a. Select **Require signature confirmation flag**.

   b. Select **Include timestamp in security header**.

   c. Select **Strict** as the Security header layout.

   d. Click the **Request message part protection** link to determine the message parts that are encrypted and signed for the *request*.

      This policy signs the Timestamp, WS-Addressing header element, and Body, and encrypts the Signature and Body. Return to the Main policy panel.

   e. Click the **Response message part protection** link to determine the message parts that are expected to be encrypted and signed for the *response*.

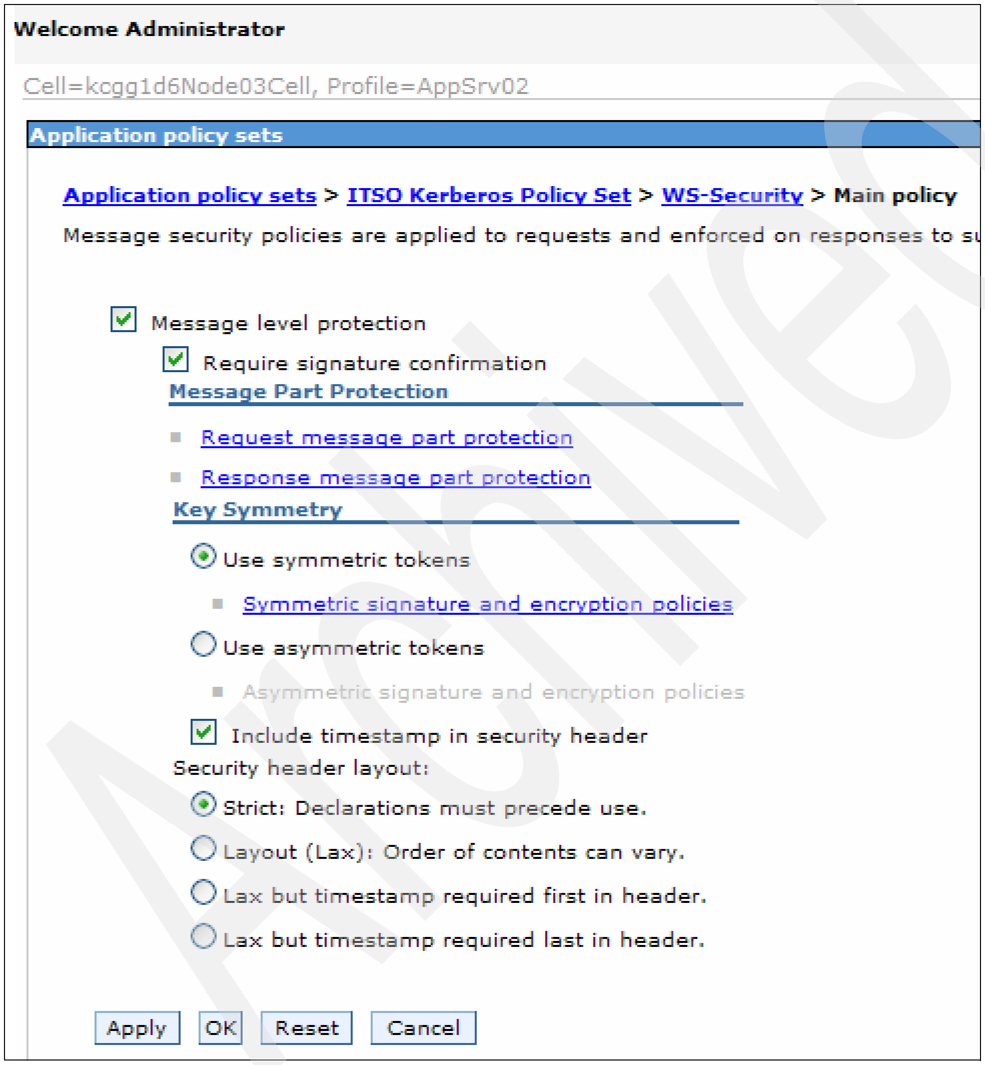
      This policy signs the Timestamp, WS-Addressing header element, and Body, and encrypts the Signature and Body. Return to the Main policy panel.

   f. Click the **Symmetric signature and encryption policies** link. Then, click **Action** → **Edit Selected policy** to determine the Kerberos token that this policy uses.

Note that you are using the following Kerberos token type for the Local part, which enables you to interoperate with the WCF implementation:

```
http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-
1.1#GSS_Kerberosv5_AP_REQ
```

Return to the Main policy panel.



Welcome Administrator

Cell=kcgg1d6Node03Cell, Profile=AppSrv02

Application policy sets

Application policy sets > ITSO Kerberos Policy Set > WS-Security > Main policy

Message security policies are applied to requests and enforced on responses to su

[✓] Message level protection

[✓] Require signature confirmation

Message Part Protection

- Request message part protection
- Response message part protection

Key Symmetry

[● Use symmetric tokens]

- Symmetric signature and encryption policies

○ Use asymmetric tokens

- Asymmetric signature and encryption policies

[✓] Include timestamp in security header

Security header layout:

[● Strict: Declarations must precede use.]

○ Layout (Lax): Order of contents can vary.

○ Lax but timestamp required first in header.

○ Lax but timestamp required last in header.

Figure 11-6 Review Main policy

5. Click **OK**, and then **Save**.

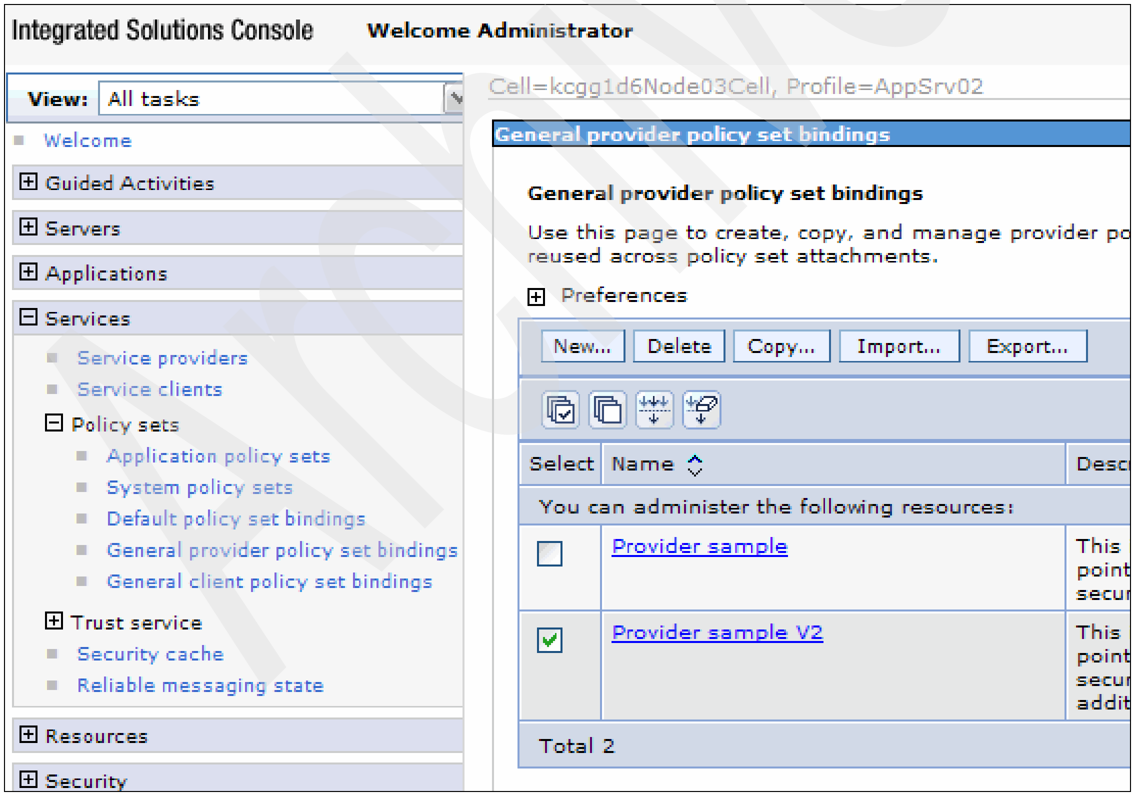
11.5.3 Configuring general bindings for the J2EE provider application

The next step is to create a compatible binding for the new ITSO Kerberos Policy Set.

WebSphere Application Server V7.0.0.1 (or later) ships a new version of the default provider binding that is not available in previous versions. The name of this new general binding is *Provider sample V2*. This scenario creates a new binding based on this binding and then updates the binding so that it can interoperate with the WCF client.

Perform the following steps to create a new general provider policy set binding based on the default binding Provider sample V2:

1. Click **Services** → **Policy sets** and then click the **General provider policy set bindings** link in the left navigation bar.
2. Select **Provider sample V2**, as shown in Figure 11-7.
3. Click **Copy**.



Cell=kcgg1d6Node03Cell, Profile=AppSrv02

General provider policy set bindings

General provider policy set bindings

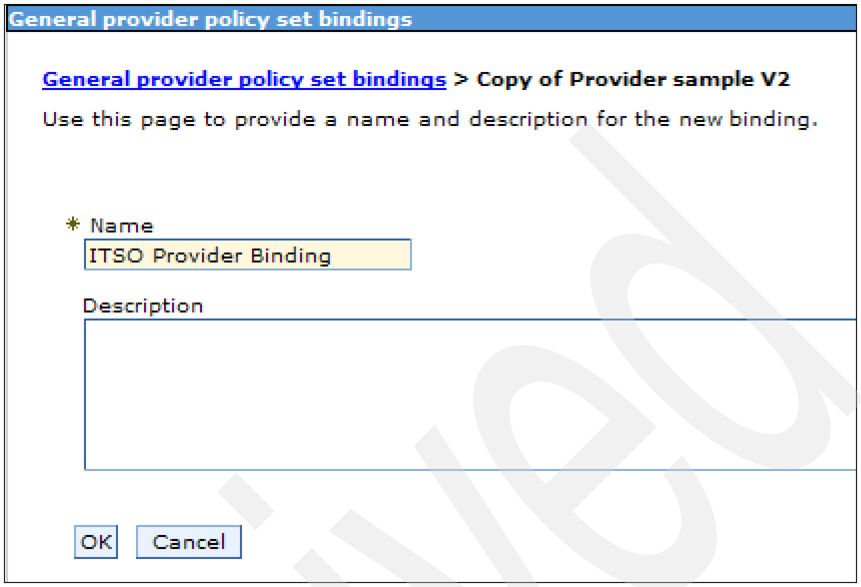
Use this page to create, copy, and manage provider po
reused across policy set attachments.

Preferences

Select	Name	Desc
You can administer the following resources:		
☐	Provider sample	This point secu
✓	Provider sample V2	This point secu addit
Total 2		

Figure 11-7 General provider policy set bindings collection table

4. Specify `ITSO Provider Binding` as the name, as shown in Figure 11-8.



General provider policy set bindings > Copy of Provider sample V2

Use this page to provide a name and description for the new binding.

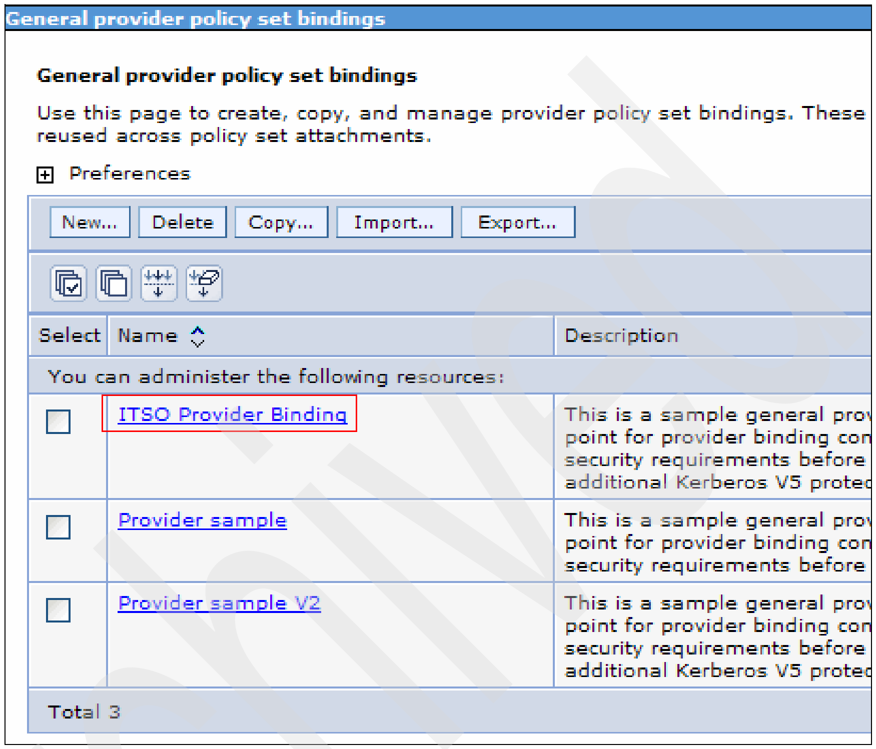
* Name
ITSO Provider Binding

Description

Figure 11-8 Create a copy of the binding Provider sample V2

5. Click **OK**, and save the changes.
6. Click **Save** to save changes to master configuration.

7. Click the **ITSO Provider Binding** link in the General provider policy set bindings collection table, shown in Figure 11-9 to navigate to the general bindings detail page.

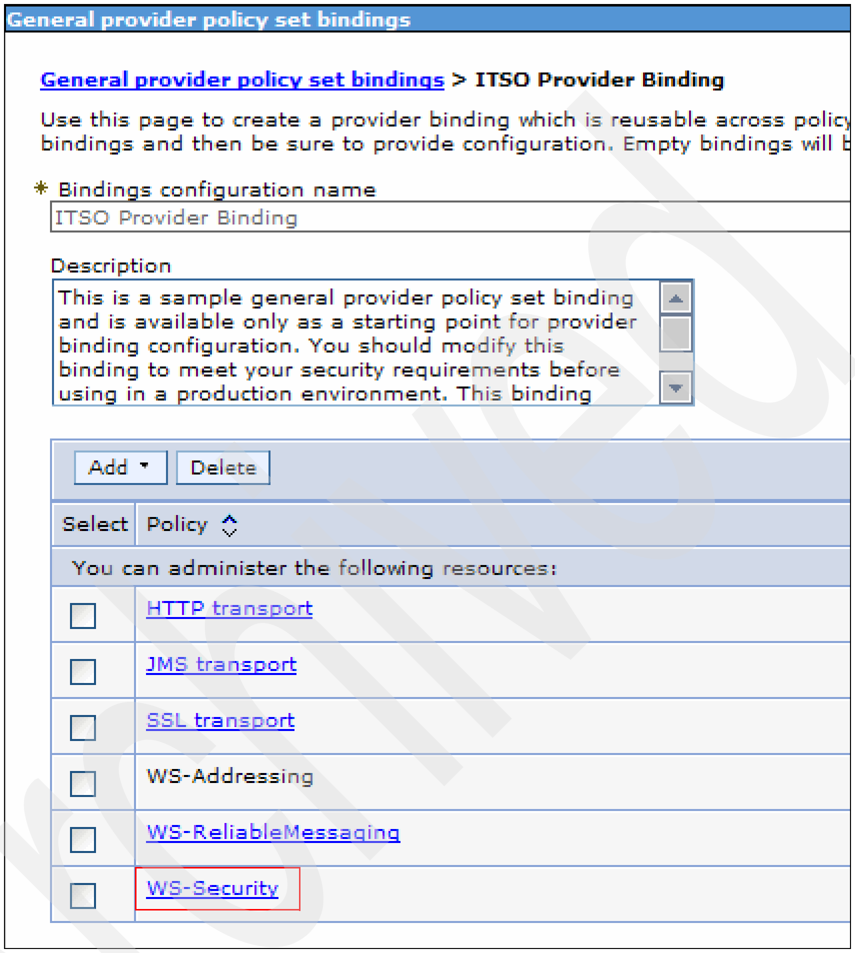

General provider policy set bindings
General provider policy set bindings
⊞ Preferences
New...
Delete
Copy...
Import...
Export...

Use this page to create, copy, and manage provider policy set bindings. These
reused across policy set attachments.

Select	Name	Description
You can administer the following resources:		
☐	ITSO Provider Binding	This is a sample general prov point for provider binding con security requirements before additional Kerberos V5 protec
☐	Provider sample	This is a sample general prov point for provider binding con security requirements before
☐	Provider sample V2	This is a sample general prov point for provider binding con security requirements before additional Kerberos V5 protec
Total 3		

Figure 11-9 General provider policy set bindings collection table

8. Click the **WS-Security** link, shown in Figure 11-10, to navigate to the bindings detail for Web service security.



General provider policy set bindings

General provider policy set bindings > **ITSO Provider Binding**

Use this page to create a provider binding which is reusable across policy bindings and then be sure to provide configuration. Empty bindings will b

* Bindings configuration name

ITSO Provider Binding

Description

This is a sample general provider policy set binding and is available only as a starting point for provider binding configuration. You should modify this binding to meet your security requirements before using in a production environment. This binding

Add ▾ Delete

Select	Policy
You can administer the following resources:	
☐	HTTP transport
☐	JMS transport
☐	SSL transport
☐	WS-Addressing
☐	WS-ReliableMessaging
☐	WS-Security

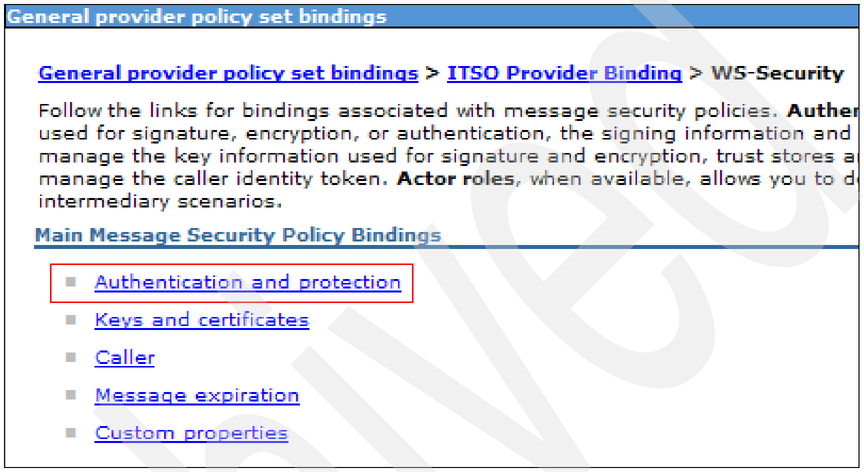
Figure 11-10 ITSO Provider Binding configuration

9. Click the **Authentication and protection** link shown in Figure 11-11, to navigate to the detail page for the following information:

   – Protection tokens (tokens used for signing and encryption)

   – Authentication tokens (tokens used for authentication).

   – Signing information and encryption information for both request and response messages



General provider policy set bindings

General provider policy set bindings > ITSO Provider Binding > WS-Security

Follow the links for bindings associated with message security policies. **Auther** used for signature, encryption, or authentication, the signing information and manage the key information used for signature and encryption, trust stores a manage the caller identity token. **Actor roles**, when available, allows you to d intermediary scenarios.

Main Message Security Policy Bindings

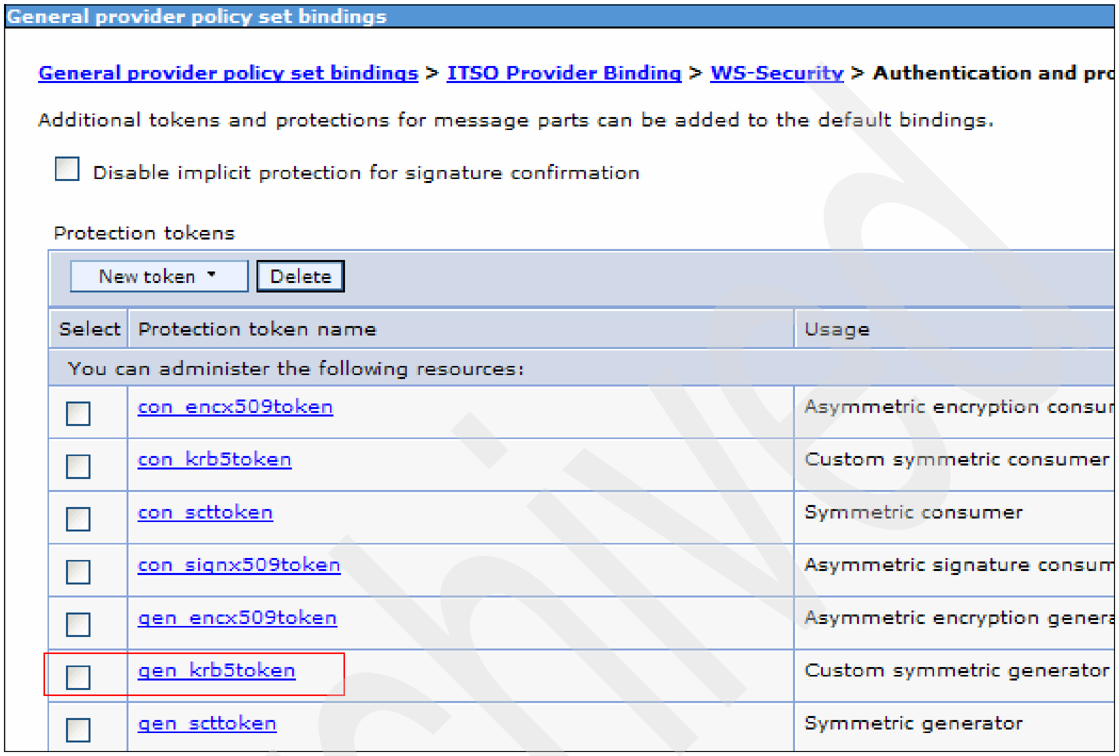
- Authentication and protection
- Keys and certificates
- Caller
- Message expiration
- Custom properties

Figure 11-11 WS-Security configuration for ITSO Provider Binding

The OASIS Web Services Security (WSS) Kerberos Token Profile 1.1 specification suggests that the use of SHA1 encoded key for every subsequent message after the initial AP_REQ packet is accepted. WebSphere Application Server V7.0 supports this SHA1 caching as described in the Kerberos token profile, by default. However, WebSphere Application Server V7.0 also provides the ability to generate new AP_REQ tokens for each request with the existing service Kerberos ticket. To interoperate with the WCF client, SHA1 caching is not used. Instead, AP_REQ packet is generated for each request.

The remaining steps update the Kerberos protection outbound token to generate an AP_REQ packet for each request.

10. Click the **gen_krb5token** link, shown in Figure 11-12, to navigate to the Kerberos token generator for the signing and encryption of the Kerberos client request.


General provider policy set bindings

General provider policy set bindings > ITSO Provider Binding > WS-Security > Authentication and pro

Additional tokens and protections for message parts can be added to the default bindings.

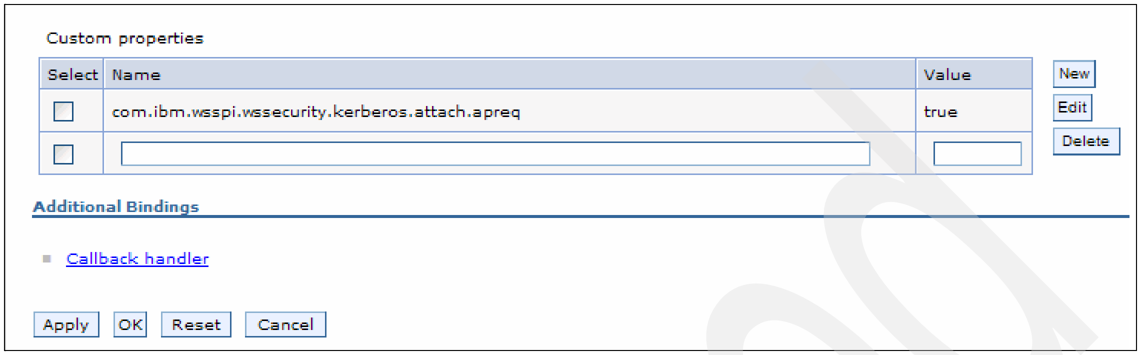
☐ Disable implicit protection for signature confirmation

Protection tokens

Select	Protection token name	Usage
You can administer the following resources:		
☐	con_encx509token	Asymmetric encryption consu
☐	con_krb5token	Custom symmetric consumer
☐	con_scttoken	Symmetric consumer
☐	con_signx509token	Asymmetric signature consum
☐	gen_encx509token	Asymmetric encryption genera
☐	gen_krb5token	Custom symmetric generator
☐	gen_scttoken	Symmetric generator

Figure 11-12 Authentication and Protection configuration for ITSO Provider Binding

11. Under the Kerberos outbound token Custom properties section, specify `com.ibm.wsspi.wssecurity.kerberos.attach.apreq` in the Name field and `true` in the Value field as shown in Figure 11-13.



Select	Name	Value
☐	com.ibm.wsspi.wssecurity.kerberos.attach.apreq	true
☐		

Figure 11-13 gen_krb5token details for the ITSO Provider Binding

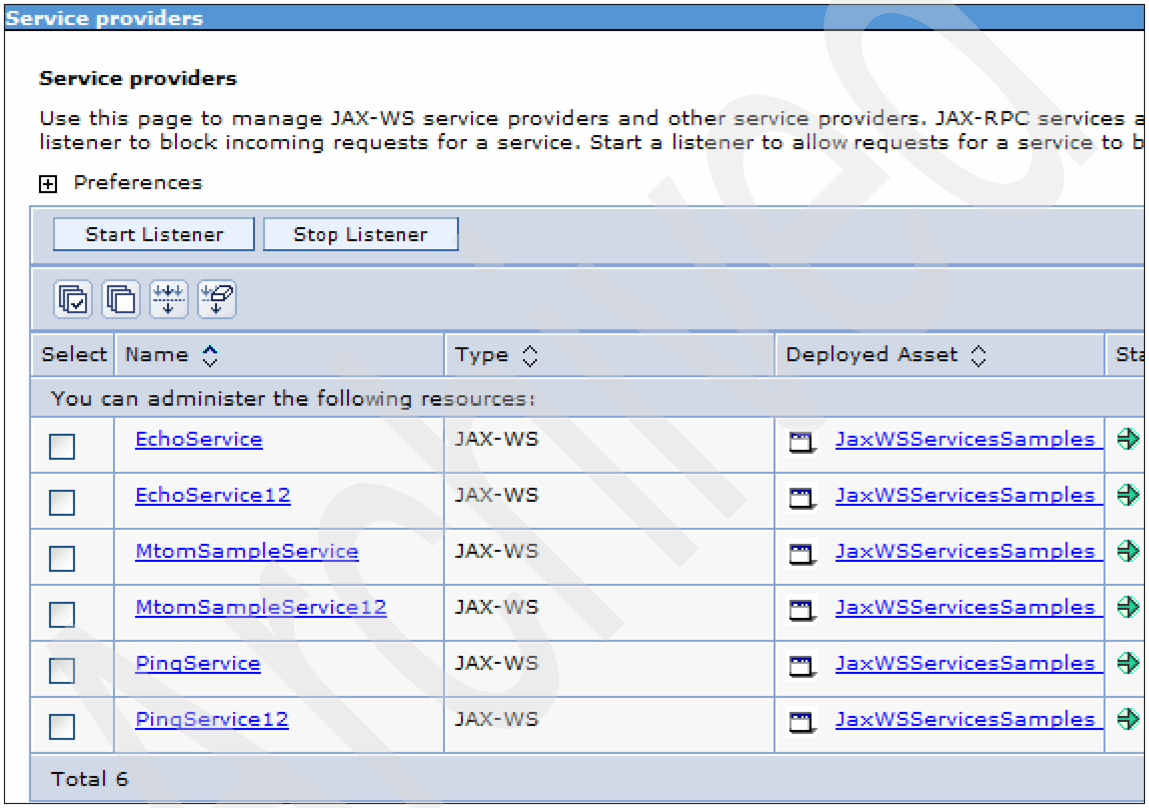
12. Click **Apply**, and then click **Save** to save the changes to master configuration.

The binding ITSO Provider binding is now configured so that it can consume and respond to a Kerberos-protected request from a service consumer.

11.5.4 Attaching the policy set and bindings to the provider application

The next step is to attach the ITSO Kerberos Policy Set and the ITSO Provider Binding to the provider service EchoService of the JaxWSServicesSamples. Follow these steps:

1. From the administrative console, select **Services** → **Service providers** to list all the service providers that are installed, as shown in Figure 11-14.



Service providers

Service providers

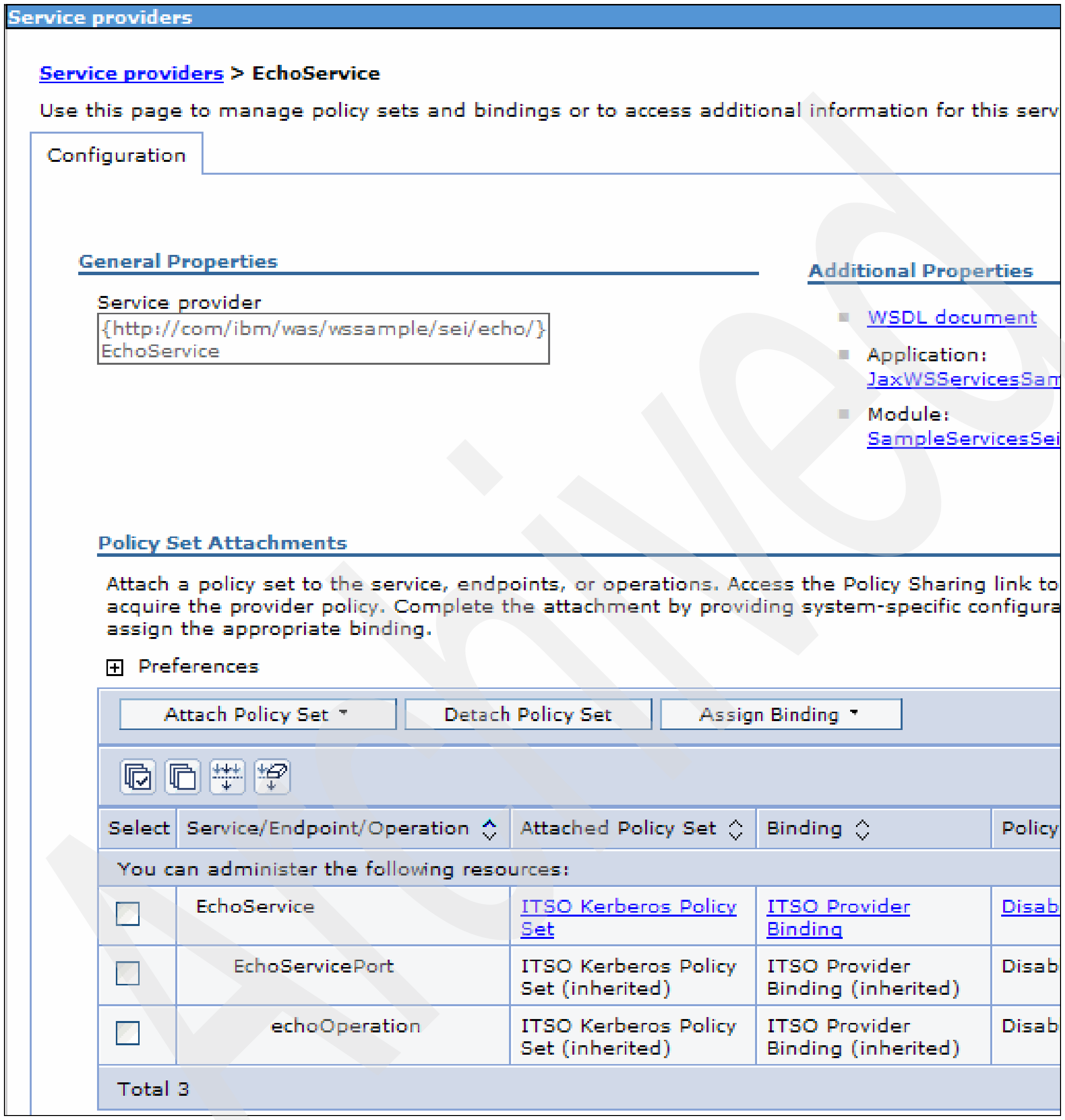
Use this page to manage JAX-WS service providers and other service providers. JAX-RPC services a
listener to block incoming requests for a service. Start a listener to allow requests for a service to b

Select	Name	Type	Deployed Asset	Sta
You can administer the following resources:				
☐	EchoService	JAX-WS	JaxWSServicesSamples	
☐	EchoService12	JAX-WS	JaxWSServicesSamples	
☐	MtomSampleService	JAX-WS	JaxWSServicesSamples	
☐	MtomSampleService12	JAX-WS	JaxWSServicesSamples	
☐	PingService	JAX-WS	JaxWSServicesSamples	
☐	PingService12	JAX-WS	JaxWSServicesSamples	
Total 6				

Figure 11-14 Service providers collection

2. Click the **EchoService** link in the collection table to navigate to the detail page.
3. Select **EchoService**.
4. Click **Attach Policy Set** → **ITSO Kerberos Policy Set**.
5. Click **Assign Binding** → **ITSO Provider Binding**.

The Policy Set Attachments collection table should look similar to that shown in Figure 11-15.


Service providers

Service providers > EchoService

Use this page to manage policy sets and bindings or to access additional information for this serv

Configuration

General Properties

Service provider
{http://com/ibm/was/wssample/sei/echo/}EchoService

Additional Properties

- WSDL document
- Application: JaxWSServicesSam
- Module: SampleServicesSei

Policy Set Attachments

Attach a policy set to the service, endpoints, or operations. Access the Policy Sharing link to acquire the provider policy. Complete the attachment by providing system-specific configura assign the appropriate binding.

⊞ Preferences

Attach Policy Set ▾ | Detach Policy Set | Assign Binding ▾

Select	Service/Endpoint/Operation	Attached Policy Set	Binding	Policy
You can administer the following resources:				
☐	EchoService	ITSO Kerberos Policy Set	ITSO Provider Binding	Disab
☐	EchoServicePort	ITSO Kerberos Policy Set (inherited)	ITSO Provider Binding (inherited)	Disab
☐	echoOperation	ITSO Kerberos Policy Set (inherited)	ITSO Provider Binding (inherited)	Disab
Total 3				

Figure 11-15 EchoService provider details

6. Save the changes.

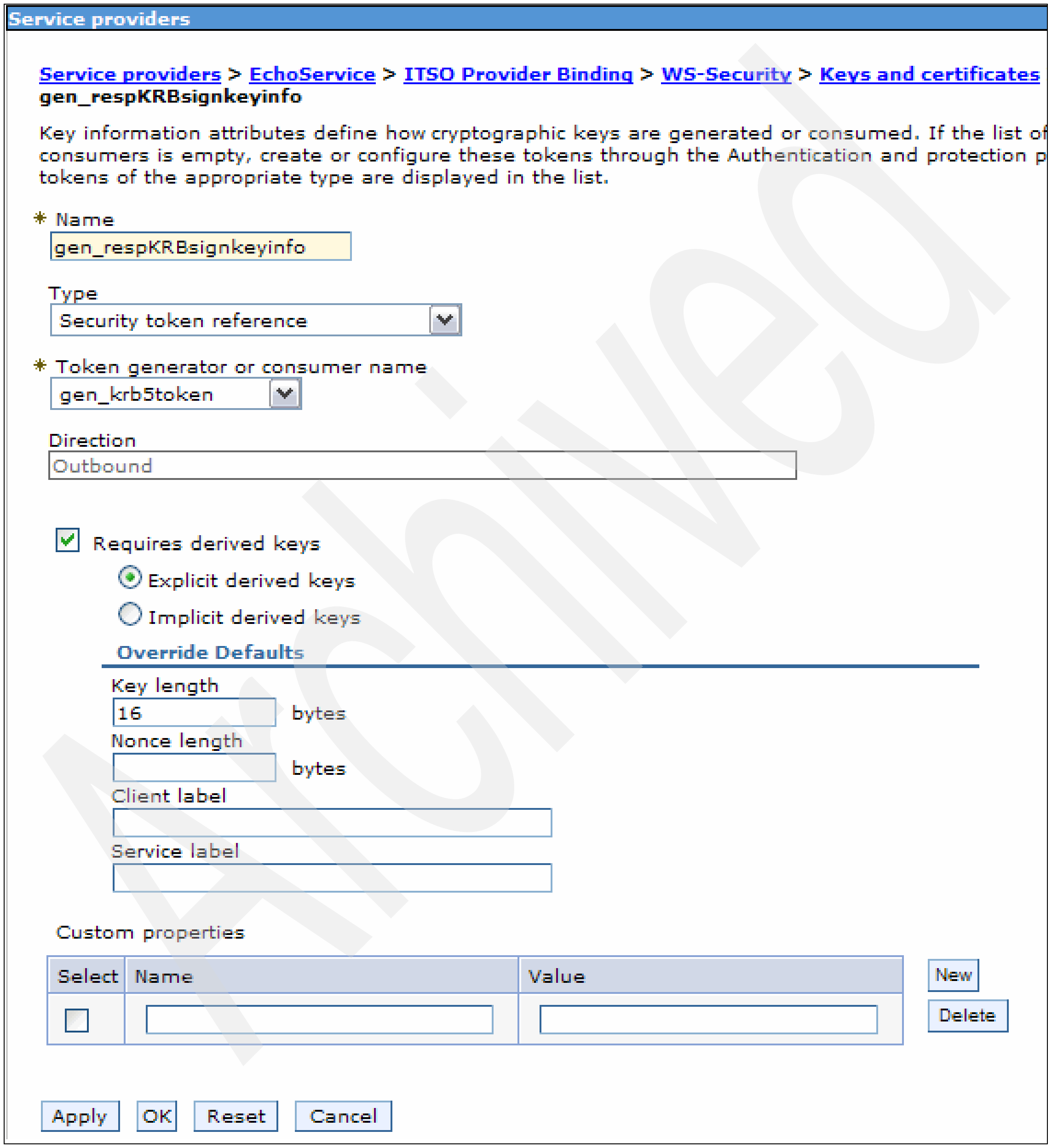
11.5.5 Changing the provider signing key size of to match the client

WebSphere Application Server V7.0.0.5 and WCF V3.5 use a different key size for the signing key. WebSphere Application Server uses a key size of 20 bytes, and WCF allows a maximum key length of 16 bytes. To be interoperable with the WCF custom Kerberos binding, the configuration of the WebSphere Application Server provider service key size must match.

Complete the following steps to configure the signing key size to 16 for the binding ITSO Provider Binding:

1. In the administrative console, select **Services** → **Service providers**.
2. Click **EchoService** → **ITSO Provider Binding** → **WS-Security**.
3. Select **Keys and certificates**.
4. Select **gen_respKRBsignkeyinfo**

5. Under the Requires derived keys section, specify 16 for the Key length field, as shown in Figure 11-16.
6. Click **OK**, and then **Save**.


Service providers

Service providers > EchoService > ITSO Provider Binding > WS-Security > Keys and certificates
gen_respKRBsignkeyinfo

Key information attributes define how cryptographic keys are generated or consumed. If the list of consumers is empty, create or configure these tokens through the Authentication and protection p tokens of the appropriate type are displayed in the list.

* Name
gen_respKRBsignkeyinfo

Type
Security token reference

* Token generator or consumer name
gen_krb5token

Direction
Outbound

✓ Requires derived keys
[● Explicit derived keys]
○ Implicit derived keys

Override Defaults

Key length
16 bytes

Nonce length
bytes

Client label

Service label

Custom properties

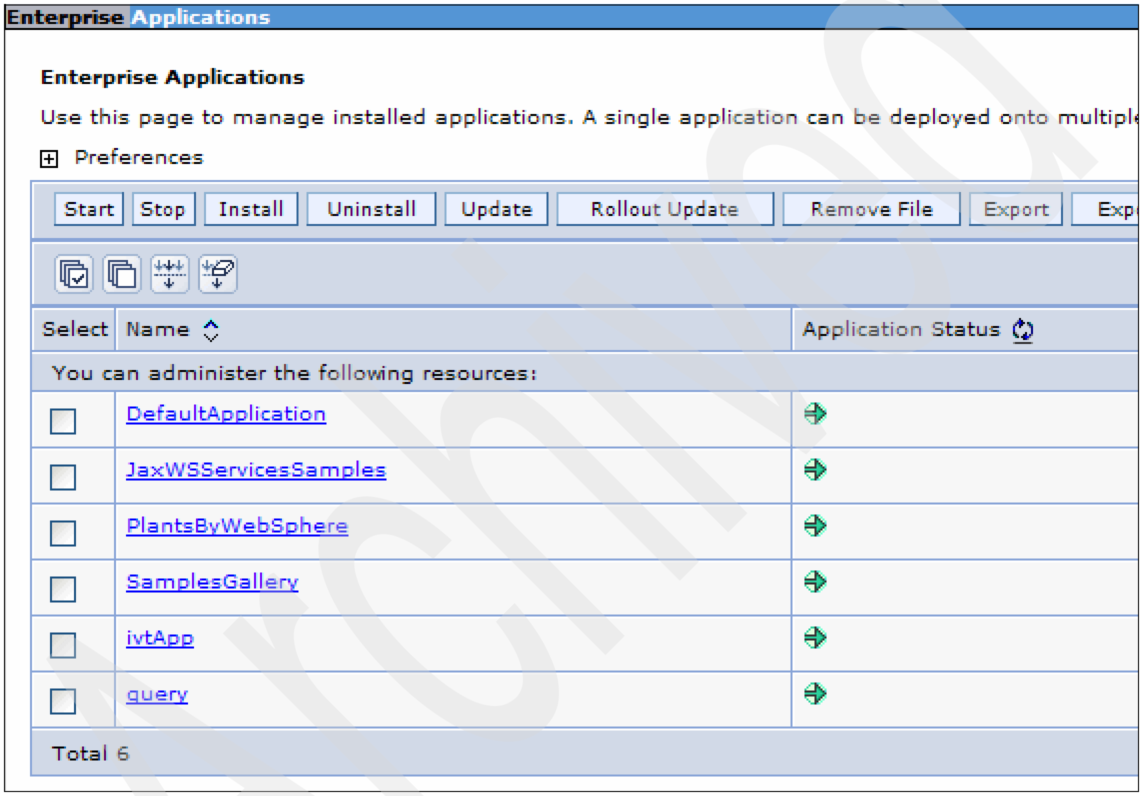
Select	Name	Value
☐		

Figure 11-16 Change the key size used for the signing key

11.6 Validating the scenario

Use the following steps to verify that the WCF console application (.NET client) can interoperate with J2EE provider service:

1. Stop and start WebSphere Application Server.
2. Using WebSphere Application Server, ensure that the sample application, JaxWSServicesSamples, is started, as shown in Figure 11-17.



Select	Name	Application Status
You can administer the following resources:		
☐	DefaultApplication	➔
☐	JaxWSServicesSamples	➔
☐	PlantsByWebSphere	➔
☐	SamplesGallery	➔
☐	ivtApp	➔
☐	query	➔
Total 6		

Figure 11-17 J2EE application JaxWSServicesSamples status

3. Open a command prompt and change directory to `C:\ITSO\WCFClient`.
4. Run the command **`WSWindowsClient.exe -e`**. Example 11-5 shows the output of this command.

Example 11-5 Scenario validation

```
C:\ITSO\WCFClient>WSWindowsClient.exe -e
CLIENT>> Using Echo endpoint from config file.
CLIENT>> Connecting to Echo Service...
CLIENT>> Sending message 'HELLO' ...
CLIENT>> The answer is 'JAX-WS==>>HELLO'
```

WS-SecurityKerberos with a Thin Client for JAX-WS and .NET provider

This chapter describes the scenario in which a Thin Client for JAX-WS sends a WS-SecurityKerberos request to Microsoft Windows Foundation Framework (WCF) Service. It includes the following topics:

12.1 Scenario overview

This scenario illustrates how to configure a Kerberos environment to secure SOAP messages that are exchanged between WebSphere Application Server V7.0.0.5 and Windows Communication Foundation (WCF) V3.5.

The scenario included in this chapter provides an example in which Thin Client for JAX-WS communicates with an application on the .NET provider (WCFService) using WS-SecurityKerberos. This scenario illustrates the changes that are required for WebSphere Application Server to interoperate with a .NET provider. Thin Client for JAX-WS and the .NET provider share the same Microsoft Active Directory for their user registry.

Note: This scenario features the following components:

- Thin Client for JAX-WS, which is installed as part of WebSphere Application Server, to access a .NET Web service
- WS-SecurityKerberos to secure Web services traffic at the message level
- Policy sets to specify Kerberos authentication
- Microsoft Kerberos key distribution center (KDC)
- .NET Web service provider application: WCFService
- Thin Client for JAX-WS script: `runSampleSei.bat`

12.1.1 Step-by-step flow of this scenario

Figure 12-1 illustrates the basic flow of this scenario.

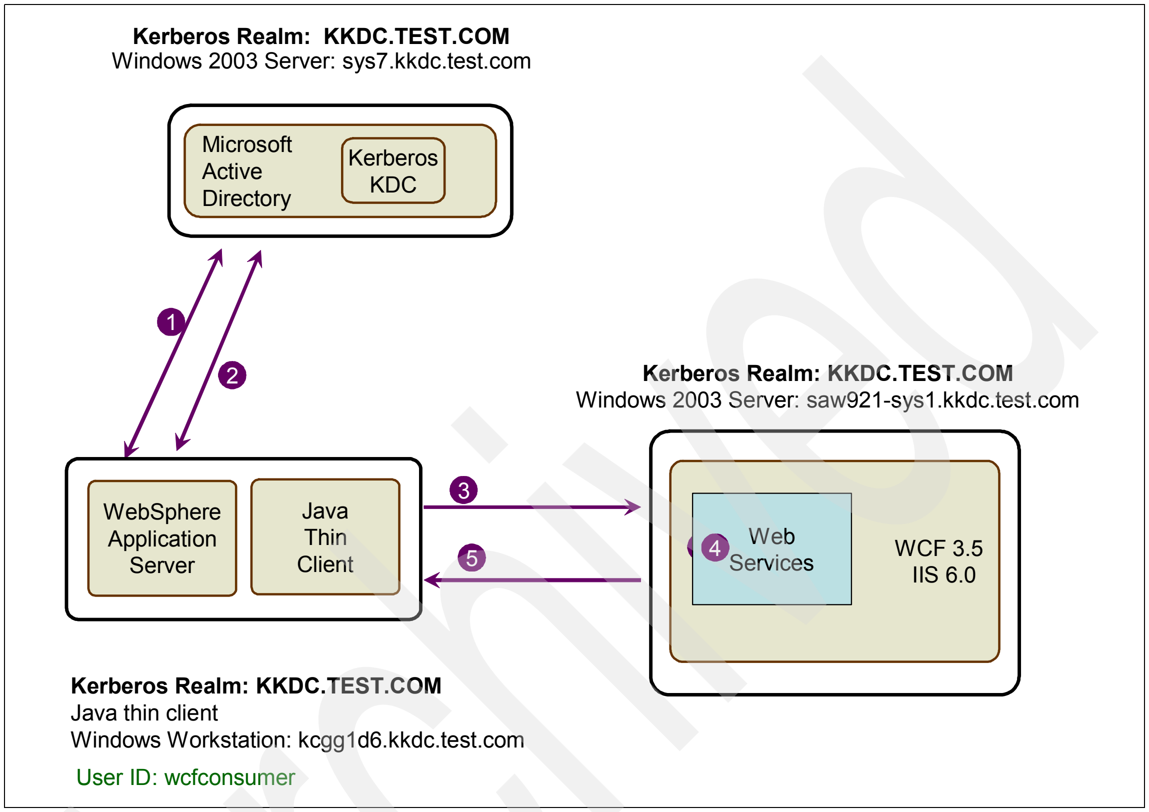

Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: sys7.kkdc.test.com
Microsoft Active Directory
Kerberos KDC
1
2
Kerberos Realm: KKDC.TEST.COM
Windows 2003 Server: saw921-sys1.kkdc.test.com
WebSphere Application Server
Java Thin Client
3
5
4
Web Services
WCF 3.5
IIS 6.0
Kerberos Realm: KKDC.TEST.COM
Java thin client
Windows Workstation: kcgg1d6.kkdc.test.com
User ID: wcfconsumer

Figure 12-1 Thin Client for JAX-WS using Kerberos authentication for .NET provider

The numbers in the figure correspond to the following steps in this process:

1. To begin, the user logs on to the Windows domain from the workstation. After the user enters a user name and password, Windows logon sends the information to the workstation local security authority, which passes the request to the Kerberos authentication package.

   The client sends an initial authentication request (AS_REQ), which includes the user's credentials and an encrypted time stamp, to the KDC. This request is a request for authentication and a ticket-granting ticket (TGT).

   The first domain controller in the domain generates the `krbtgt/`*`domain_name`* ticket. The KDC uses the secret key to decrypt the time stamp and issues a TGT to the client.

The response AS_REP is encrypted with the user's key and is returned to the user. The ticket is encrypted with the KDC's key and is enclosed in the AS_REP.

2. The Web service client sends a request (using the TGT) for a service ticket to the ticket-granting server for communication with the Web services. The ticket-granting server then returns a service ticket (TGS_REP) and the session key to the client. The client uses this information to create the Kerberos token.
3. After the Kerberos token is created, the WSS generator encodes and inserts the token into the SOAP message and propagates the token for token consumption or acceptance.
4. The Web service provider receives and extracts the Kerberos token from the SOAP request. The provider then accepts the Kerberos token by validating the token with its own secret key. The secret key of the service is stored in a keytab file. After acceptance, the Web services provider stores the associated request token information into the context Subject.
5. The provider return a SOAP response.

12.1.2 Scenario components

The scenario in this chapter uses the following components:

- Microsoft .NET Framework Version 3.5
- The application server

  WebSphere Application Server V7.0.0.5 is installed on Windows XP Professional 2002 SP2.
- The client system, Windows XP Professional 2002 SP2

  This application is used to host Thin Client for JAX-WS. WebSphere Application Server V7.0.0.5 is installed on this host.
- The Kerberos KDC, Microsoft Windows Server 2003 SP2 with Active Directory

  The Windows 2003 Support Tools are installed.

Download materials: A sample .NET Web service called `WCFService` is included in the additional material for this book. To use this sample, download the compressed file, `WCFSample.zip`, and extract the files to the `C:\` directory. For more information about how to download this file, see Appendix E, "Additional material" on page 525.

12.1.3 Basic requirements for this scenario

A working domain controller and at least one client computer in that domain are required for this solution. Using SPNEGO from the domain controller does not work.

We recommend that you also have the following components before configuring:

- Internet Information Services (IIS) 6.0 and Microsoft .NET Framework 3.5 installed on the Web services provider system
- Windows SDK for Windows Server 2008 installed on the Web services provider system

  This SDK includes the utilities, such as `svcconfigeditor`, that are used to edit the WCF provider and service custom bindings.
- The sample .NET provider application (WCFService), which is included in the `WCFSample.zip` file in the additional material for this book, downloaded to the to the system where the Microsoft .NET Framework 3.5 is installed

  The WCFService sample application is hosted in an IIS 6.0 installation (`saw921-sys1.kkdc.test.com`) and accessed from the remote client system (`kcgg1d6.kkdc.test.com`).

  For more information about the additional material for this book, see Appendix E, "Additional material" on page 525.
- A functioning Microsoft Windows 2000 or 2003 Active Directory Domain that includes the following components:
  - Domain controller
  - Client workstation

  Users must be able to log in to the client workstation.
- WebSphere Application Server for Windows installed that include Thin Client for JAX-WS

  The profile must be created after WebSphere Application Server V7.0.0.1 is installed (`kcgg1d6.kkdc.test.com`).

Important: WebSphere Application Server Version V7.0.0.1 (or later) bundles new general bindings, policy sets, and run times that are used for this scenario. The WebSphere Application Server fix pack installation process does not import the new Kerberos general bindings and policy sets that are required for this scenario to an existing WebSphere Application Server profile. Thus, you need to create a new WebSphere Application Server profile after installing WebSphere Application Server V7.0.0.1 (or later) to ensure that you can use new bindings and policy sets that are shipped with WebSphere Application Server.

- Access for users from the Microsoft Active Directory to WebSphere Application Server's protected resources using a native authentication mechanism, such as basic authentication (BA) or forms authentication
- Windows 200x Support Tools on the Microsoft Windows Server, including the **`setspn`**, the **`ktpass`**, and the **`adsiedit`** tools that this scenario uses

  These support tools are available on the Windows Server installation CD or from the following Microsoft Web site:

  `http://www.microsoft.com/downloads/details.aspx?familyid=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D`

We also recommend that you install the Windows Server 2003 Resource Kit Tools on the Thin Client for JAX-WS system. These tools provide the **`kerbtray.exe`** utility to see Kerberos tickets on the user workstation. These tools are available at:

`http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd`

12.2 Configuring the KDC and client system

To configure the KDC and client system, first complete these steps:

1. Set up a Windows Server with Microsoft Active Directory domain and associated Kerberos KDC. The Windows administrator needs to complete this step.
2. Join the .NET client system and the Microsoft Active Directory domain controller. To join the Microsoft Active Directory domain, right-click the **My Computer** icon. Then, click **Properties**. In the System Properties dialog box, go to the Computer Name tab, and click **Change**. Enter the Microsoft Active Directory domain as the system name. For example, in our scenario, the

computer `kcgg1d6` joined the Microsoft Active Directory domain `kkdc.test.com`.

3. Log on to the Microsoft Windows Active Directory domain with an identity known in Active Directory. In this scenario, we logged on to the Microsoft Active Directory domain with the user name `wcfconsumer`.
4. Make sure that you can connect to the Kerberos KDC from the client system. Use the **`kinit`** tool that ships with the JDK to test the connection as follows:
   a. Open a command prompt, and navigate to the *`app_server_root`*`\java\jre\bin` directory.
   b. Type **`kinit`** and your user name. Then, enter your password.

   Example 12-1 shows a successful connection to Microsoft Active Directory domain.

Example 12-1 Use kinit to test the connection

```
C:\WebSphere\AppServer\java\jre\bin>kinit wcfconsumer
Password for wcfconsumer@KKDC.TEST.COM:

Done!
New ticket is stored in cache file C:\Documents and
Settings\wcfconsumer\krb5cc_
wcfconsumer
```

12.2.1 Creating an account for WCFService

Next, you need to create a user account for the WCFService application in the Microsoft Active Directory. This account is mapped to the Kerberos service principal name (SPN). This SPN identifies the .NET provider as a Kerberized service.

To create this SPN for WebSphere Application Server, follow these steps:

1. Log on as an administrator to the Windows 2003 Server system that serves as the domain controller.
2. Invoke the Microsoft Active Directory users and computers snap-in. Click **Start** → **Run**. Then type the following command, and click **OK**:

   ```
   dsa.msc /server=localhost
   ```
3. Create a user account for WCFService. This scenario creates the `wcfprovider` user account.

12.2.2 Creating the keytab file

Now, you can create the Kerberos keytab file. Use the **ktpass** tool to set up the Kerberos keytab file (wcfProvider.keytab) for the previously created SPN. The **ktpass** command allows you to configure a non-Windows Server 2003 Kerberos service (such as WebSphere applications on a z/OS platform) as a security principal in the Windows Server 2003 Active Directory. This tool generates a keytab file that contains the shared key of the service. In our scenario, we use the commands listed in Example 12-2.

Example 12-2 Example output from ktpass utility

```
C:\Documents and Settings\Administrator>ktpass.exe -out
c:\wcfProvider.keytab -p
rinc echoSrvProvider/saw921-sys1.kkdc.test.com@KKDC.TEST.COM -pass
itso4you -map
User wcfprovider -mapOp set -ptype KRB5_NT_PRINCIPAL -crypto
RC4-HMAC-NT
Targeting domain controller: sys7.kkdc.test.com
Using legacy password setting method
Successfully mapped echoSrvProvider/saw921-sys1.kkdc.test.com to
wcfprovider.
Key created.
Output keytab to c:\wcfProvider.keytab:
Keytab version: 0x502
keysize 90 echoSrvProvider/saw921-sys1.kkdc.test.com@KKDC.TEST.COM
ptype 1 (KRB5
_NT_PRINCIPAL) vno 5 etype 0x17 (RC4-HMAC) keylength 16
(0x31a1f13010baa0c4f9fc8
ead6de09db5)

C:\Documents and Settings\Administrator>setspn -L wcfprovider
Registered ServicePrincipalNames for
CN=wcfprovider,CN=Users,DC=kkdc,DC=test,DC=
com:
    echoSrvProvider/saw921-sys1.kkdc.test.com
```

12.2.3 Creating the configuration file

Next, you create the Kerberos configuration file in the system that hosts WCFService, which in this scenario is `saw921-sys1.kkdc.test.com`. To create the configuration file, follow these steps:

1. Copy the key tab file (`wcfProvider.keytab`) to the WCFService system. In this scenario, we copied the file to the `c:\windows` folder.
2. Create the Kerberos configuration file `c:\windows\krb5.ini` using the commands listed in Example 12-3.

Example 12-3 Kerberos configuration file used for the scenario

```
[libdefaults]
   default_realm = KKDC.TEST.COM
   default_keytab_name = FILE:///c:/WINDOWS/wcfProvider.keytab
   default_tkt_enctypes = rc4-hmac
   default_tgs_enctypes = rc4-hmac
   forwardable  = false
   renewable  = false
   clockskew  = 300
[realms]
   KKDC.TEST.COM = {
      kdc = sys7.kkdc.test.com:88
      default_domain = kkdc.test.com
   }
[domain_realm]
   .kkdc.test.com = KKDC.TEST.COM
```

12.3 Configuring WCF 3.5 and IIS 6.0

In this scenario, the .NET Framework 3.5 is used to run the WCFService application. You can install this application from the following Microsoft Web site. It is recommended that you install .NET Framework 3.5 *after* IIS 6.0 is already installed.

http://www.microsoft.com/downloads/details.aspx?FamilyID=333325fd-ae52-4e35-b531-508d977d32a6&DisplayLang=en

Run the command **`aspnet_regiis /i`** to ensure that ASP.NET V2.0.50727 is registered with IIS 6.0. The following Microsoft link has additional details for this command:

http://msdn.microsoft.com/en-us/library/k6h9cz8h(VS.71).aspx

Run the command **`WFServicesReg.exe /c`** to ensure that WCF 3.5 is configured correctly. The following link has additional details for this command:

`http://msdn.microsoft.com/en-us/library/bb924408.aspx`

If the system is configured correctly, when IIS Manager starts, ASP.NET v2.0.50727 is listed as one of the Web Service Extensions. Right-click **ASP.NET v2.0.50727**, and then click **Allow**.

> **Note:** If you have previous version of ASP.NET installed (such as v1.1.4322), make sure that you uninstall the previous version using the **`aspnet_regiis -u`** command from the corresponding .NET directory and then reboot the system.

Windows SDK for Windows Server 2008 and .NET Framework 3.5 includes utilities, such as `svcconfigeditor`, that you can use to edit .NET provider applications. You can download and install Windows SDK from the following Microsoft Web site:

`http://www.microsoft.com/downloads/details.aspx?FamilyId=F26B1AA4-741A-433A-9BE5-FA919850BDBF&displaylang=en`

12.4 Configuring the bindings for WCFService

Now, you need to configure the .NET provider application to interoperate with the J2EE Consumer application. The sample .NET provider application is based on the same WSDL as the J2EE consumer application.

The .NET provider application configuration already has a custom binding called *EchoSOAP*. The EchoSOAP binding has the `httptransport` and `textMessageEncoding` (`Soap11WSAddressing10`) extensions enabled. You can use the Microsoft Service Configuration Editor to update this binding to include WS-Security. Perform the following steps to a security binding extension and configure the security settings to match the J2EE Consumer application configuration:

1. Click **All Programs** → **Microsoft Windows SDK v6.1** → **Tools** → **Service Configuration Editor**.
2. Click **File** → **Open** → **Config File**.
3. Select the configuration file that ships with the samples, `C:\ITSO\WCFService\web.config`, and click **Open**.
4. In the left pane, select **Bindings** → **EchoSOAP(customBinding)**.
5. In the right pane, Click **Add** to open the Adding Binding Element Extensions dialog box. Then, click **Security**, and click **Add**.

6. In the left pane, select **Security** and change the following settings in the General tab in the right pane:
   a. Set AuthenticationMode to `Kerberos`.
   b. Set Default Algorithm to `Basic128Rsa15`.
   c. Use the default value `SignBeforeEncryptAndEncryptSignature` for MessageProtectionOrder.
   d. Set MessageSecurityVersion to `WSSecurity11WSTrust13WSSecureConversation13WSSecurityPolicy12BasicSecurityProfile10`.
   e. Use the default value `True` for requireDerivedKeys.
   f. Use the default value `True` for equireSecurityContextCancellation.
   g. Set requireSignatureConfirmation to `True`.
   h. Use the default value `Strict` for securityHeaderLayout.
7. Select **File** → **Save** to save the changes to the `wswindowsservice.exe`.config configuration file.
8. In the left pane, select **Services** → **Endpoints** → **EchoServiceEndPoint**.
9. In the right pane, set Address to `WSSampleSei/EchoService`.
10. Switch to the Identity tab, and set ServicePrincipalName to `echoSrvProvider/kcgg1d6.kkdc.test.com`.
11. Select **File** → **Save** to save the changes to the configuration file `web.config`.

12.5 Deploying the WCFService application

Ensure that WCF 3.5 and IIS 6.0 are installed and configured as described in 12.3, “Configuring WCF 3.5 and IIS 6.0” on page 281.

Configure the KKDC\wcfprovider user with the correct access rights as follows:

- The KKDC\wcfprovider user should be member of the group Users and IIS_WPG.
- The KKDC\wcfprovider user should have read/write access to the `C:\WINWOWS\TEMP` directory. If not, the exception listed in Example 12-4 is issued by the .NET framework.

Example 12-4 Example error when wcfprovider does not have access rights

```
...
error CS2001: Source file 'C:\WINDOWS\TEMP\pxhe3toi.0.cs' could not
be found
```

```
error CS2008: No inputs specified
...
```

- Run the **aspnet_regiis -ga KKDC\wcfprovider**, as shown in Example 12-5, command so that the KKDC\wcfprovider user has correct access rights.

Example 12-5 The wcfprovider user given access to IIS metabase

```
C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727>aspnet_regiis -ga
KKDC\wcfprovider

Start granting KKDC\wcfprovider access to the IIS metabase and other
directories
 used by ASP.NET.
Finished granting KKDC\wcfprovider access to the IIS metabase and
other director
ies used by ASP.NET.
```

Use the following link as a reference when performing the following actions:

http://msdn.microsoft.com/en-us/library/ms998297.aspx

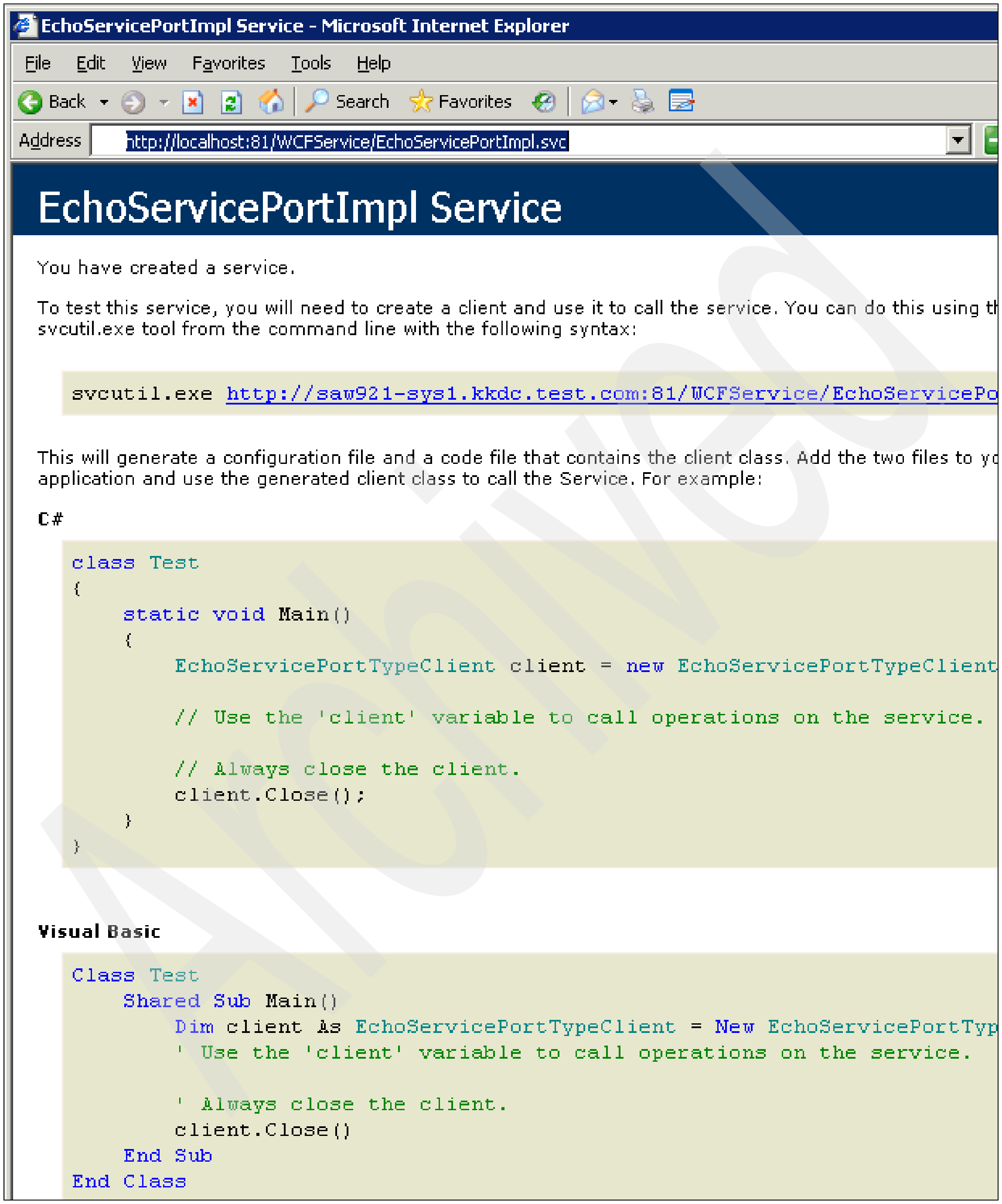
1. Assign ASP.NET permissions to the new account.
2. Create an application pool with a custom identity.
3. Configure the application to run in the new application pool.

For our current scenario, we use KKDC\wcfprovider as the service account. To deploy WCFService, follow these steps:

1. Start IIS Manager.
2. Create a new Web site called *DOTNETPROVIDER* as follows:
   a. Use the TCP port 81. You can use a different port number to suit your environment.
   b. Use `C:\IBM\ITSO` as the home directory.
   c. Allow Read and Execute permission.
3. Create a new application pool called `EchoServicePool`.
4. Right-click the EchoServicePool application pool, and select **Properties**.
5. Go to the Identity tab. Use the KKDC\wcfprovider for the application pool identity.
6. Select the DOTNETPROVIDER Web site, right-click WCFService, and select **Properties**.
7. Click **Create** to create the IIS application.

8. Also, use the EchoServicePool application pool for the IIS application.
9. Switch to the ASP.NET tab, and ensure that ASP.NET version is set to `v2.0.50727`.
10. Stop the DOTNETPROVIDER Web site and the EchoServicePool application pool, and then start the applications again.
11. Test that the WCFService is started and available, as shown in Figure 12-2, using the following link:

    `http://localhost:81/WCFService/EchoServicePortImpl.svc`


EchoServicePortImpl Service - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost:81/WCFService/EchoServicePortImpl.svc

EchoServicePortImpl Service

You have created a service.

To test this service, you will need to create a client and use it to call the service. You can do this using th
svcutil.exe tool from the command line with the following syntax:

```
svcutil.exe http://saw921-sys1.kkdc.test.com:81/WCFService/EchoServicePo
```

This will generate a configuration file and a code file that contains the client class. Add the two files to yo
application and use the generated client class to call the Service. For example:

C#

```
class Test
{
    static void Main()
    {
        EchoServicePortTypeClient client = new EchoServicePortTypeClient

        // Use the 'client' variable to call operations on the service.

        // Always close the client.
        client.Close();
    }
}
```

Visual Basic

```
Class Test
    Shared Sub Main()
        Dim client As EchoServicePortTypeClient = New EchoServicePortTyp
        ' Use the 'client' variable to call operations on the service.

        ' Always close the client.
        client.Close()
    End Sub
End Class
```

Figure 12-2 Successful creation of EchoServicePortImpl Service

12. Open a command prompt, and change to the `C:\ITSO\WCFClient` directory.

13. Copy the following file:

    `WSWindowsClient.exe.config.NET`

    Name the new copy of the file:

    `WSWindowsClient.exe.config`

    Make sure that the following values are set correctly:

    – The servicePrincipalName attribute has the following value:

      `echoSrvProvider/saw921-sys1.kkdc.test.com`

    – The address attribute has the following value:

      ```
      http://saw921-sys1.kkdc.test.com:81/WCFService/EchoServicePort
      Impl.svc/WSSampleSei/EchoService
      ```

    Note: The `WSWindowsClient.exe.config` configuration file does not include a security binding. The `WSWindowsClient.exe.config.NET` configuration file has the required security binding to test the WCFService.

14. Run the **`WSWindowsClient.exe -e`** command, as shown in Example 12-6.

Example 12-6 Sample WCF Client is communicating with WCFService

```
C:\ITSO\WCFClient>WSWindowsClient.exe -e
CLIENT>> Using Echo endpoint from config file.
CLIENT>> Connecting to Echo Service...
CLIENT>> Sending message 'HELLO' ...
CLIENT>> The answer is 'DotNet==> HELLO'
```

Tip: The files `C:\ITSO\WCFService\EchoServicePortImpl_messages.svclog` and `C:\ITSO\WCFService\EchoServicePortImpl_messages.svclog` are generated when the WCFService is running. You can view these files using the Service Trace Viewer utility that ships with Windows SDK for Windows Server 2008 and .NET Framework 3.5 for troubleshooting.

12.6 Configuring the J2EE client

This section shows how to enable Kerberos for the sample Thin Client for JAX-WS application. You can use the WebSphere Application Server administrative console to create the Kerberos policy set and bindings that are needed for the sample application.

12.6.1 Creating a new policy set

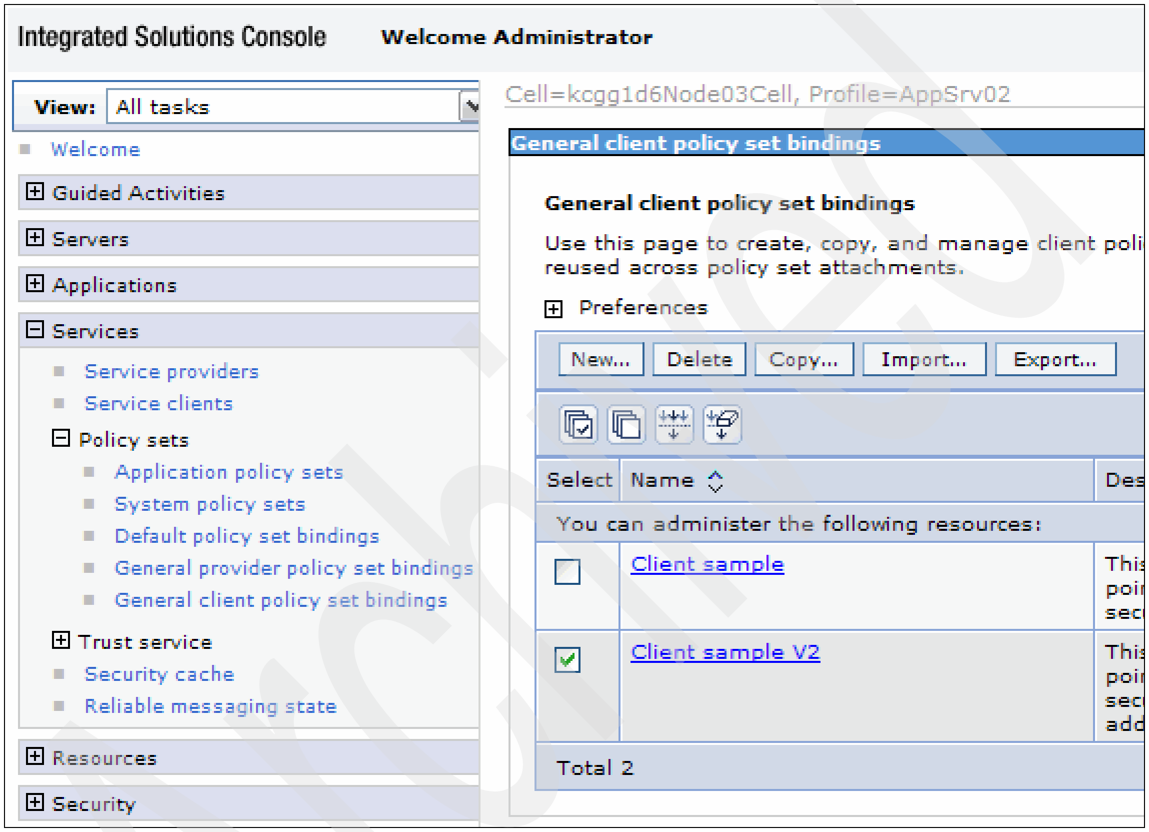
To create a new policy set named *ITSO Kerberos Policy Set*, follow the instructions in 11.5.2, “Configuring the policy set for the Web service” on page 254.

12.6.2 Configuring the bindings

To create a compatible binding for the new ITSO Kerberos Policy Set WebSphere Application Server V7.0.0.1 (or later) ships a new version of the default client binding that is not available in V7.0.0.0. The name of this new general binding is *Client sample V2*. The new binding for this scenario is created based on this binding and then is updated so that it can interoperate with WCFService.

To create a new general provider policy set binding based on the default binding Client sample V2, follow these steps:

1. Click the **General client policy set bindings** link in the left navigation bar under **Services** → **Policy sets**.
2. Select **Client sample V2**, and click **Copy**, as shown in Figure 12-3.



Cell=kcgg1d6Node03Cell, Profile=AppSrv02

General client policy set bindings

General client policy set bindings

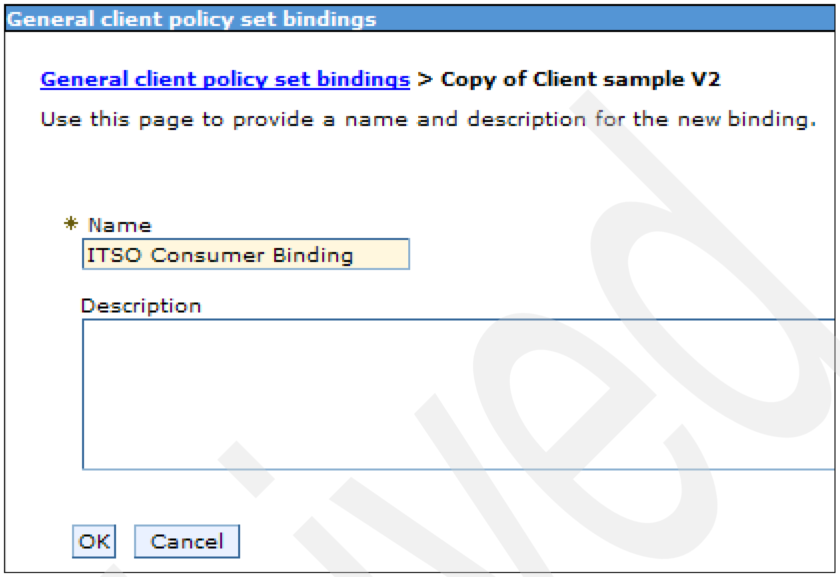
Use this page to create, copy, and manage client poli
reused across policy set attachments.

⊞ Preferences

Select	Name	Des
You can administer the following resources:		
☐	Client sample	This poi sec
☑	Client sample V2	This poi sec add
Total 2		

Figure 12-3 General client policy set binding collection table

3. In the settings for the General client policy set bindings window, specify `ITSO Consumer Binding` as the name, as shown in Figure 12-4.


General client policy set bindings

General client policy set bindings > Copy of Client sample V2

Use this page to provide a name and description for the new binding.

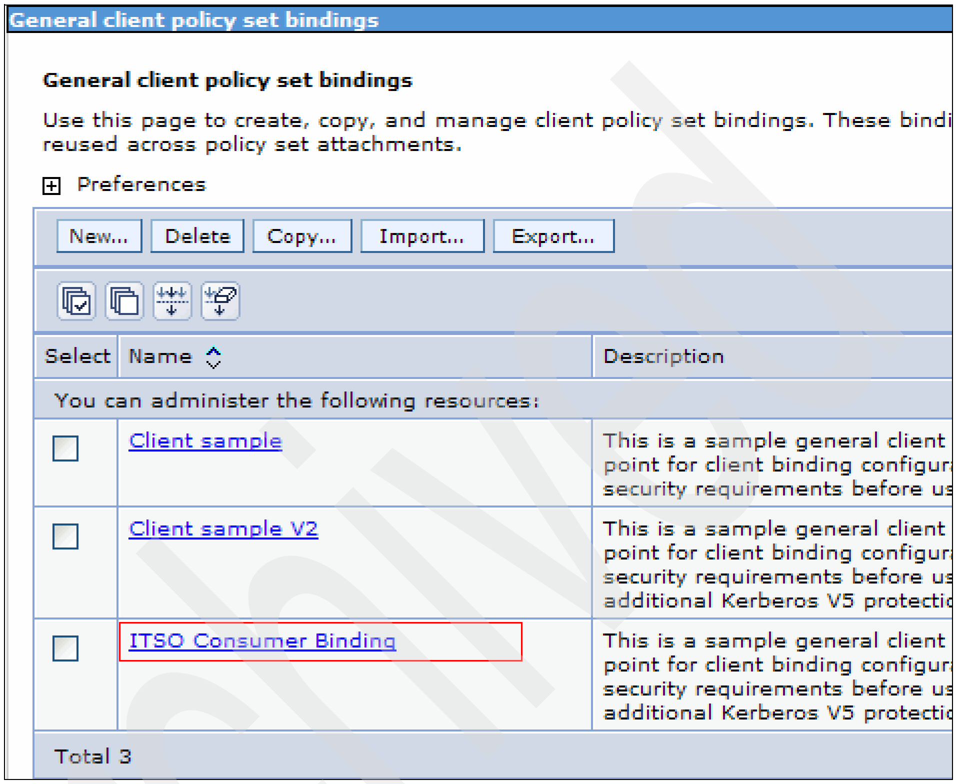
* Name: ITSO Consumer Binding

Description:

Figure 12-4 Create a copy of the cinding Client sample V2

4. Click **OK** and save the changes.
5. Click **Save** to save changes to master configuration.

6. Click the **ITSO Consumer Binding** link in the General client policy set bindings collection table, shown in Figure 12-5, to navigate to the general bindings detail page.



General client policy set bindings

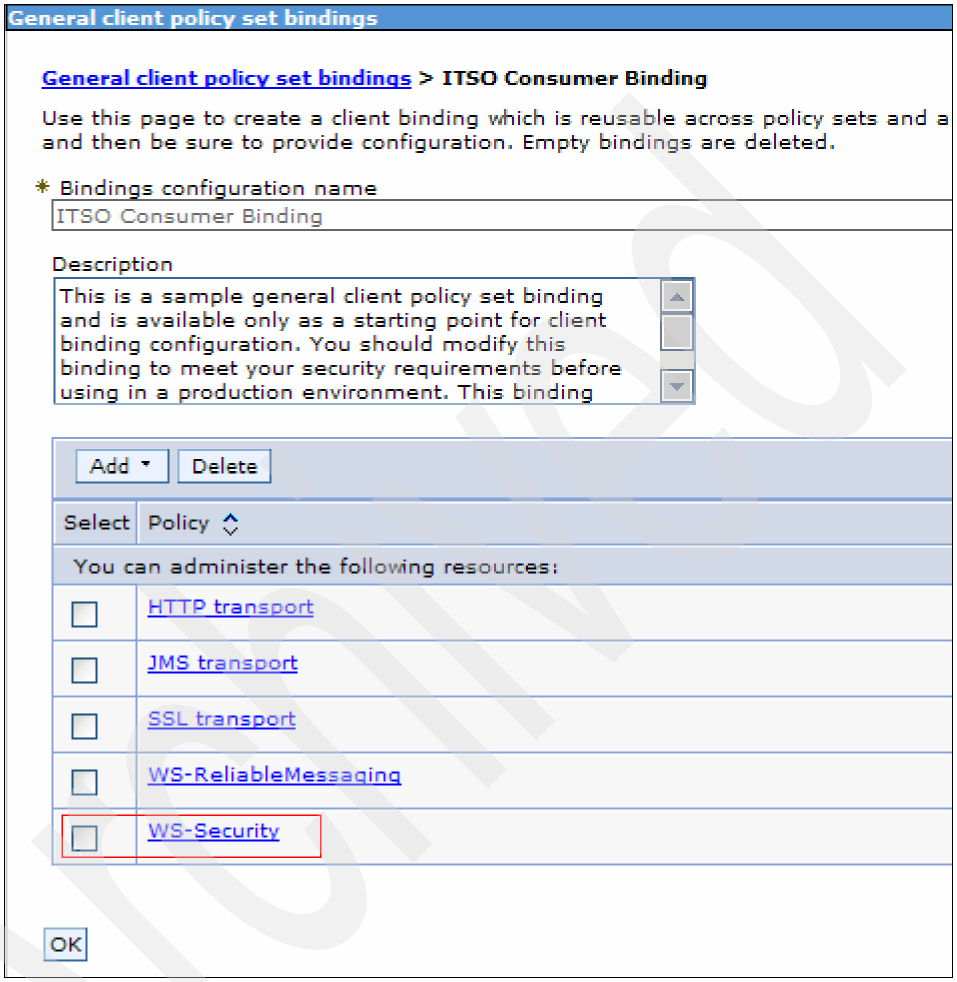
Use this page to create, copy, and manage client policy set bindings. These bindi
reused across policy set attachments.

⊞ Preferences

Select	Name	Description
You can administer the following resources:		
☐	Client sample	This is a sample general client point for client binding configur security requirements before us
☐	Client sample V2	This is a sample general client point for client binding configur security requirements before us additional Kerberos V5 protectio
☐	ITSO Consumer Binding	This is a sample general client point for client binding configur security requirements before us additional Kerberos V5 protectio
Total 3		

Figure 12-5 General client policy set bindings collection table

7. Click the **WS-Security** link, shown in Figure 12-6, to navigate to the bindings detail for Web service security.


General client policy set bindings

General client policy set bindings > ITSO Consumer Binding

Use this page to create a client binding which is reusable across policy sets and a and then be sure to provide configuration. Empty bindings are deleted.

* Bindings configuration name
ITSO Consumer Binding

Description
This is a sample general client policy set binding and is available only as a starting point for client binding configuration. You should modify this binding to meet your security requirements before using in a production environment. This binding

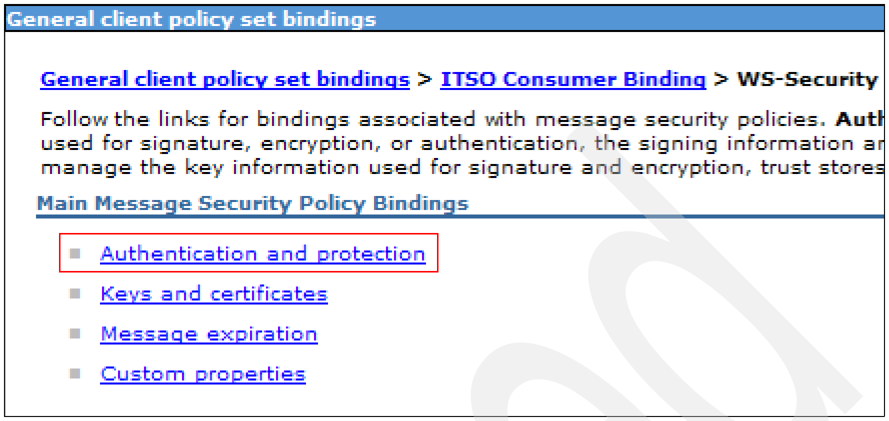
Add ▾ | Delete

Select	Policy
You can administer the following resources:	
☐	HTTP transport
☐	JMS transport
☐	SSL transport
☐	WS-ReliableMessaging
☐	WS-Security

OK

Figure 12-6 ITSO Consumer Binding conffiguration

8. Click the **Authentication and protection** link, shown in Figure 12-7.


General client policy set bindings

General client policy set bindings > ITSO Consumer Binding > WS-Security

Follow the links for bindings associated with message security policies. Auth
used for signature, encryption, or authentication, the signing information ar
manage the key information used for signature and encryption, trust stores

Main Message Security Policy Bindings

- Authentication and protection
- Keys and certificates
- Message expiration
- Custom properties

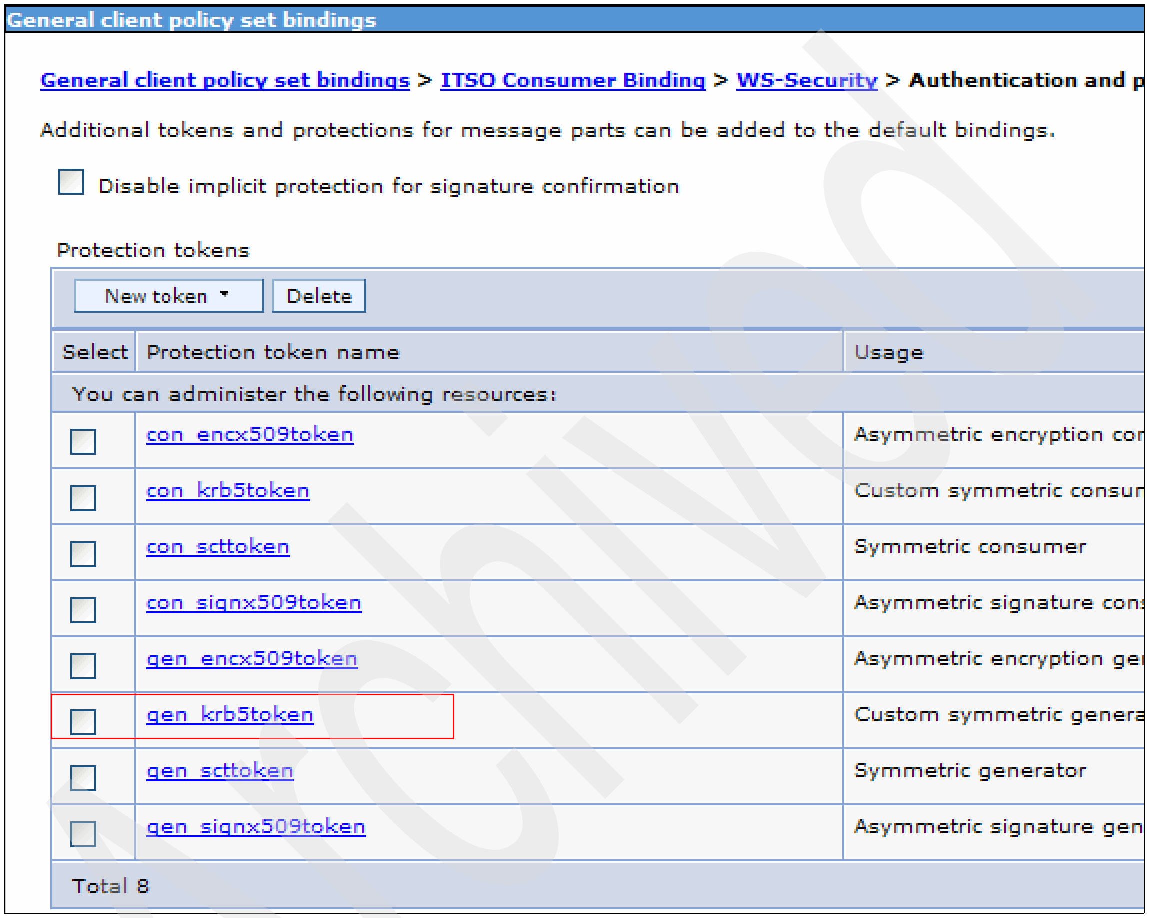
Figure 12-7 WS-Security configuration for ITSO Consumer Binding

9. The next page, shown in Figure 12-8 on page 294, includes the following links:
   - Protection tokens (tokens used for signing and encryption).
   - Authentication tokens (tokens used for authentication).
   - Signing information and encryption information for both request and response messages.

   The OASIS Web Services Security (WSS) Kerberos Token Profile 1.1 specification suggests that the use of SHA1 encoded key for every subsequent message after the initial AP_REQ packet is accepted. WebSphere Application Server V7.0 supports this SHA1 caching as described in the Kerberos token profile, by default. However, WebSphere Application Server V7.0 also provides the ability to generate new AP_REQ tokens for each request with the existing service Kerberos ticket.

   To interoperate with WCFService, SHA1 caching is not used. Instead, an AP_REQ packet is generation for each request.

To update the Kerberos protection outbound token to generate an AP_REQ packet for each request with correct SPN, click **gen_krb5token**, as shown in Figure 12-8, to navigate to the Kerberos token generator for the signing and encryption of the Kerberos client request.



General client policy set bindings

General client policy set bindings > ITSO Consumer Binding > WS-Security > **Authentication and p**

Additional tokens and protections for message parts can be added to the default bindings.

☐ Disable implicit protection for signature confirmation

Protection tokens

Select	Protection token name	Usage
You can administer the following resources:		
☐	con_encx509token	Asymmetric encryption cor
☐	con_krb5token	Custom symmetric consur
☐	con_scttoken	Symmetric consumer
☐	con_signx509token	Asymmetric signature con
☐	gen_encx509token	Asymmetric encryption ge
☐	gen_krb5token	Custom symmetric genera
☐	gen_scttoken	Symmetric generator
☐	gen_signx509token	Asymmetric signature gen
Total 8		

Figure 12-8 Authentication and Protection configuration for ITSO Consumer Binding

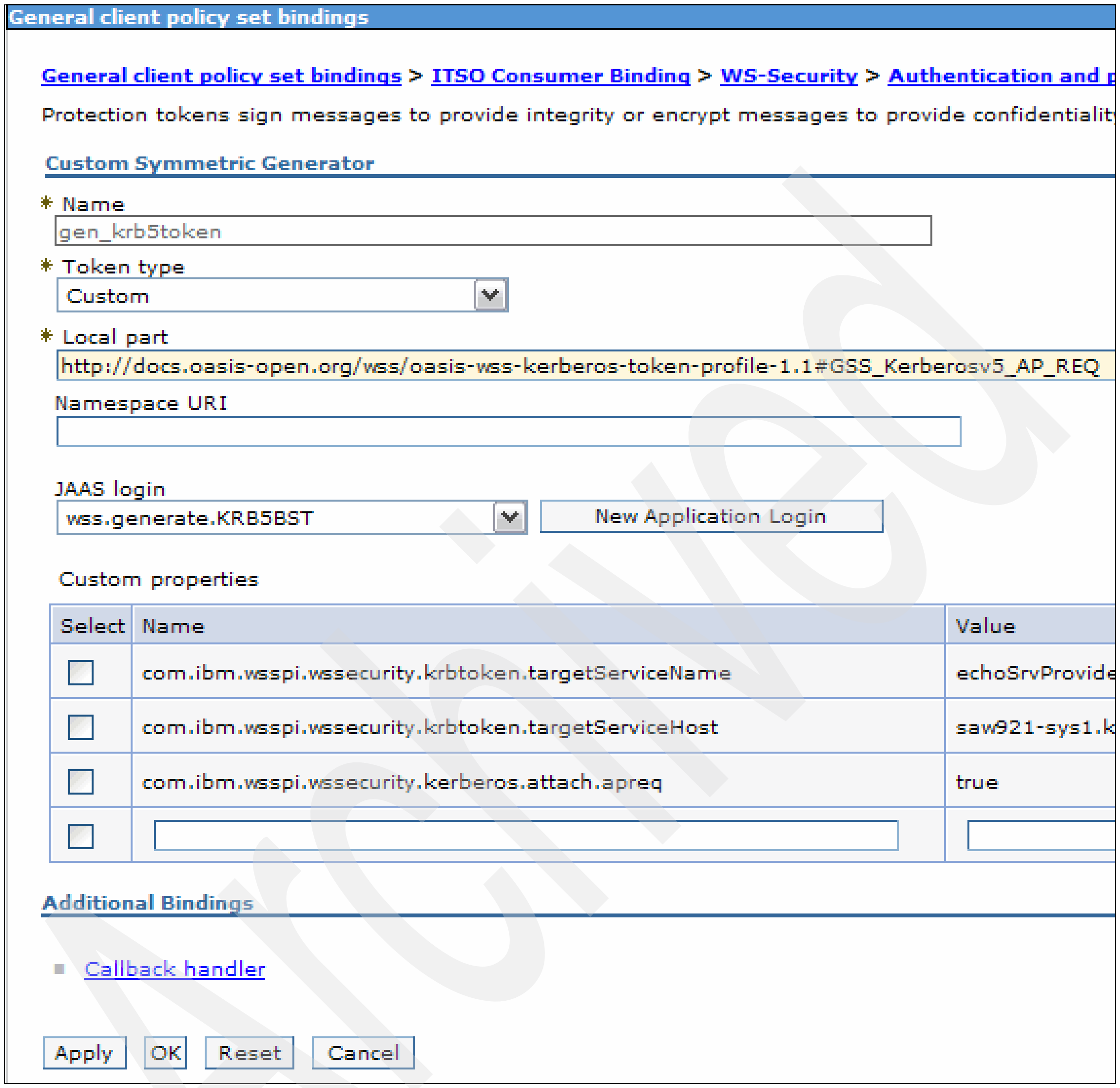
10. Under custom properties, specify the proper values for the following properties:

   – `com.ibm.wsspi.wssecurity.krbtoken.targetServiceName`
   – `com.ibm.wsspi.wssecurity.krbtoken.targetServiceHost`

   Follow these steps:

   a. Select **com.ibm.wsspi.wssecurity.krbtoken.targetServiceName**. Click **Edit**, and then update the Value field to `echoSrvProvider`. Click **Apply**.
   b. Select **com.ibm.wsspi.wssecurity.krbtoken.targetServiceName**. Click **Edit**, and then update the Value field to `saw921-sys1.kkdc.test.com`. Click **Apply**.

11. Under the Kerberos outbound token custom property section, specify `com.ibm.wsspi.wssecurity.kerberos.attach.apreq` in the Name field and specify `true` in the Value field as shown in Figure 12-9 on page 296.


General client policy set bindings

General client policy set bindings > ITSO Consumer Binding > WS-Security > Authentication and p

Protection tokens sign messages to provide integrity or encrypt messages to provide confidentialit

Custom Symmetric Generator

* Name
gen_krb5token

* Token type
Custom

* Local part
http://docs.oasis-open.org/wss/oasis-wss-kerberos-token-profile-1.1#GSS_Kerberosv5_AP_REQ

Namespace URI

JAAS login
wss.generate.KRB5BST

Custom properties

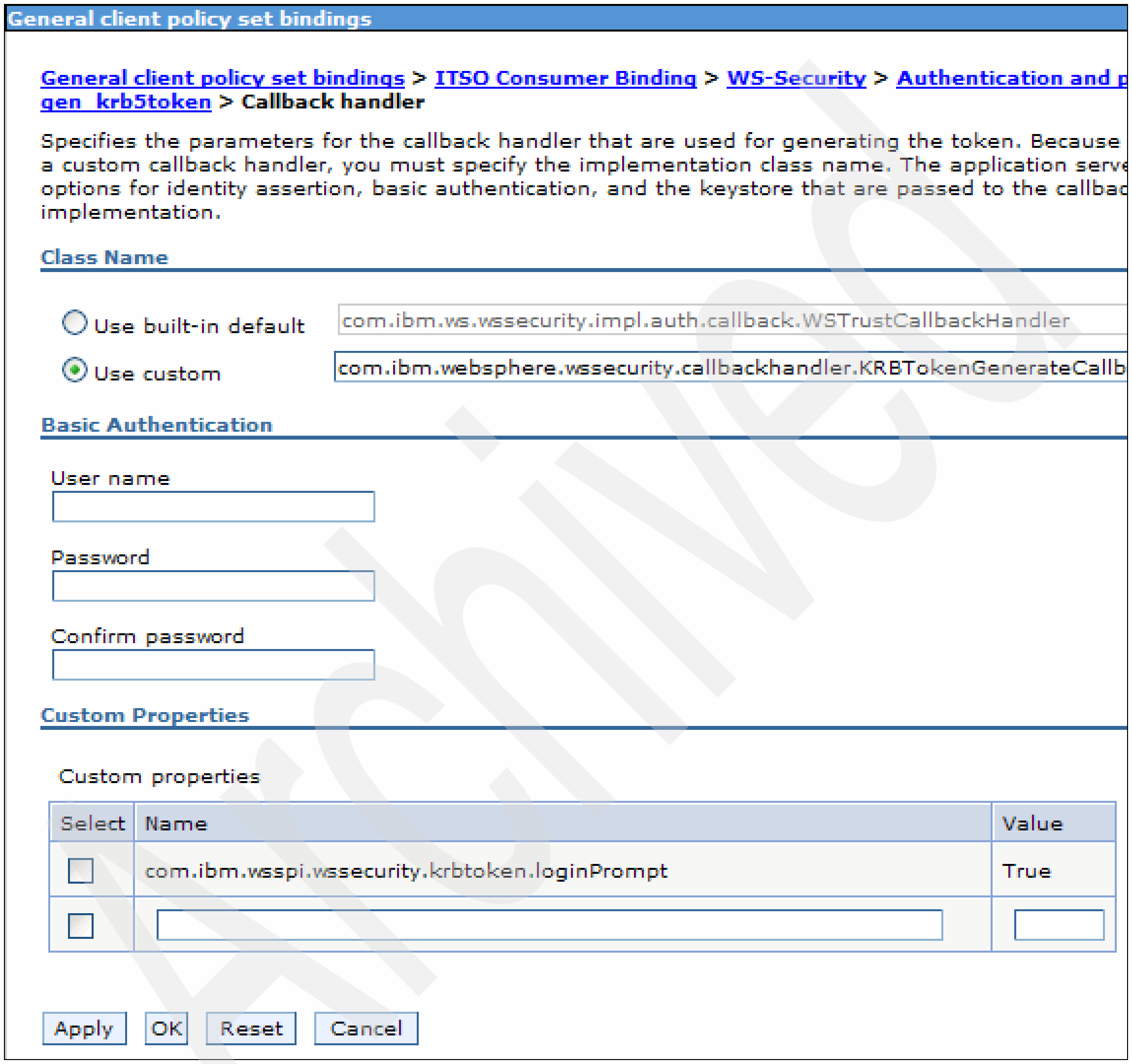
Select	Name	Value
☐	com.ibm.wsspi.wssecurity.krbtoken.targetServiceName	echoSrvProvide
☐	com.ibm.wsspi.wssecurity.krbtoken.targetServiceHost	saw921-sys1.k
☐	com.ibm.wsspi.wssecurity.kerberos.attach.apreq	true
☐		

Additional Bindings

- Callback handler

Figure 12-9 The gen_krb5token details for the ITSO Consumer Binding

12.Click **Callback handler** and add a new custom property, `com.ibm.wsspi.wssecurity.krbtoken.loginPrompt` with a value of `True`, as shown in Figure 12-10.



General client policy set bindings

General client policy set bindings > ITSO Consumer Binding > WS-Security > Authentication and p
gen_krb5token > Callback handler

Specifies the parameters for the callback handler that are used for generating the token. Because
a custom callback handler, you must specify the implementation class name. The application serve
options for identity assertion, basic authentication, and the keystore that are passed to the callbac
implementation.

Class Name

Use built-in default: com.ibm.ws.wssecurity.impl.auth.callback.WSTrustCallbackHandler

[✓ Use custom]: com.ibm.websphere.wssecurity.callbackhandler.KRBTokenGenerateCallb

Basic Authentication

User name

Password

Confirm password

Custom Properties

Custom properties

Select	Name	Value
☐	com.ibm.wsspi.wssecurity.krbtoken.loginPrompt	True
☐		

Apply OK Reset Cancel

Figure 12-10 Callback handler

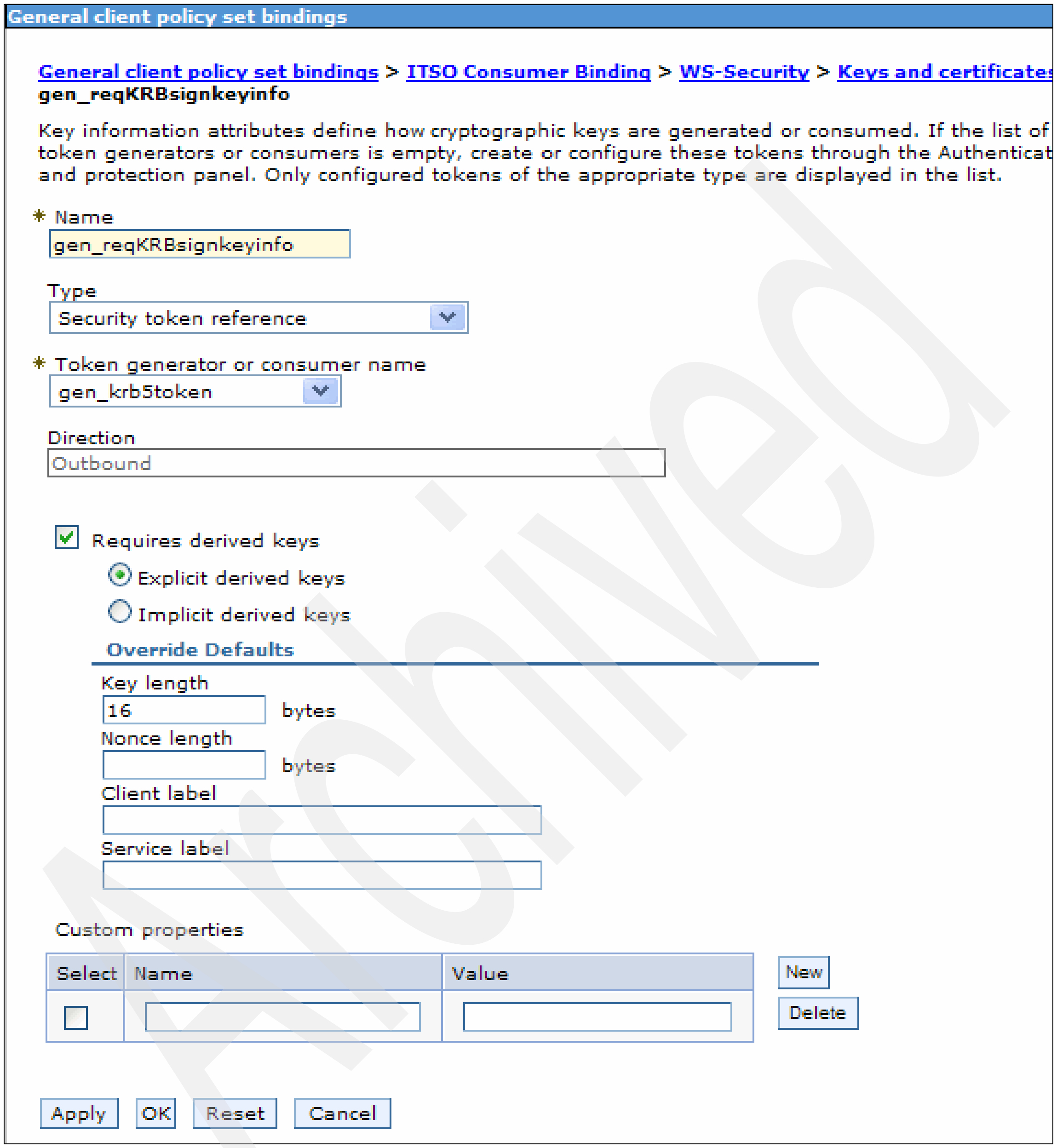
13.Click **OK**, and click **Save** to save the changes to master configuration.

12.6.3 Changing size of signing keys of Thin Client for JAX-WS to match the .NET provider

WebSphere Application Server V7.0.0.5 and WCF V3.5 use a different key size for the signing key. WebSphere Application Server uses a key size of 20 bytes, and WCF allows a maximum key length of 16 bytes. To be interoperable with the WCF custom Kerberos binding, the configuration of the WebSphere Application Server client key size must match.

To set the signing key size to 16 for the binding ITSO Client Binding, follow these steps from the WebSphere Application Server administrative console:

1. Click **Services** → **Policy Sets** and then select the **General client policy set bindings** link on the left navigation bar.
2. Click **ITSO Client Binding**, and then click **WS-Security**.
3. Click **Keys and certificates**.
4. Click **gen_reqKRBsignkeyinfo**
5. In the “Requires derived keys” section, specify `16` for the “Key length” field. as shown in Figure 12-11.
6. Click **OK**, and then click **Save**.


General client policy set bindings

General client policy set bindings > ITSO Consumer Binding > WS-Security > Keys and certificate

gen_reqKRBsignkeyinfo

Key information attributes define how cryptographic keys are generated or consumed. If the list of token generators or consumers is empty, create or configure these tokens through the Authenticat and protection panel. Only configured tokens of the appropriate type are displayed in the list.

* Name
gen_reqKRBsignkeyinfo

Type
Security token reference

* Token generator or consumer name
gen_krb5token

Direction
Outbound

[✓] Requires derived keys

● Explicit derived keys

○ Implicit derived keys

Override Defaults

Key length
16 bytes

Nonce length
bytes

Client label

Service label

Custom properties

Select	Name	Value
☐		

Figure 12-11 Change the key size used for the signing key

12.6.4 Updating the Thin Client for JAX-WS client script and policy sets

To configure the sample Thin Client for JAX-WS that ships with WebSphere Application Server, follow these steps:

1. Open a command prompt and change to the following directory:

   ```
   C:\WebSphere\AppServer\samples\bin\JaxWSServicesSamples
   ```

2. Edit the script `runSampleSei.bat` with the following changes:

   a. Update `CLASSPATH` to include a semicolon and period (`;.`) at the end.

   b. Add the path to the Java Authentication and Authorization (JAAS) login file using the following statement:

   ```
   set
   JAASLOGIN=-Djava.security.auth.login.config=C:/IBM/WebSphere/p
   rofiles/AppSrv02/properties/wsjaas_client.conf
   ```

   c. Add the following parameter to the Java command:

   ```
   %JAASLOGIN%
   -DWAS_PROPS_DIR=C:/WebSphere/AppServer/profiles/AppSrv02/prope
   rties
   -DUSER_INSTALL_ROOT=C:/WebSphere/AppServer/profiles/AppSrv02
   ```

   Example 12-7 shows the revised `runSampleSei.bat` script.

Example 12-7 Revised script runSampleSei.bat

```
:# This program may be used, executed, copied, modified and distributed
:# without royalty for the purpose of developing, using, marketing, or
distributing.
@echo off
setlocal

call %~dp0..\..\..\bin\setupCmdLine

set THIN_JAR_FILE=com.ibm.jaxws.thinclient_7.0.0.jar
set THIN_JAR=%WAS_HOME%\runtimes\%THIN_JAR_FILE%

if exist %THIN_JAR% goto INSTALLOK
echo Jar file %THIN_JAR_FILE% cannot be located.
echo This sample requires installation of "Stand-alone thin clients".
echo Please correct the installation options before running this
sample.
goto END
```

```
:INSTALLOK
set
CLASSPATH=.;%THIN_JAR%;%WAS_HOME%\samples\lib\JaxWSServicesSamples\WSSa
mpleClientSei.jar;.
set
JAASLOGIN=-Djava.security.auth.login.config=C:/WebSphere/AppServer/prof
iles/AppSrv02/properties/wsjaas_client.conf
%JAVA_HOME%\bin\java %JAASLOGIN%
-DWAS_PROPS_DIR=C:/WebSphere/AppServer/profiles/AppSrv02/properties
-DUSER_INSTALL_ROOT=C:/WebSphere/AppServer/profiles/AppSrv02 -cp
"%CLASSPATH%" com.ibm.was.wssample.sei.cli.SampleClient %*
:END
```

3. Configure the following required directories:
   - Create a directory named `META-INF` within the *app_server_root*`\samples\bin\JaxWSServicesSamples` directory.
   - Create a directory named `PolicySets\ITSO Kerberos Policy Set` within the *app_server_root*`\samples\bin\JaxWSServicesSamples\META-INF` directory.
   - Create a directory named `bindings\ITSO Consumer Binding` within the *app_server_root*`\samples\bin\JaxWSServicesSamples\META-INF` directory.
4. Copy the policy set as follows:

   profile_root`\config\cells\`*CELL_NAME*`\PolicySets\ITSO Kerberos Policy Set`

   to

   app_server_root`\samples\bin\JaxWSServicesSamples\META-INF\PolicySets\ITSO Kerberos Policy Set`
5. Copy the custom bindings as follows:

   profile_root`\config\cells\`*CELL_NAME*`\bindings\ITSO Consumer Binding`

   to

   app_server_root`\samples\bin\JaxWSServicesSamples\META-INF\bindings\ITSO Consumer Binding`

6. Create a new file, `clientPolicyAttachments.xml`, in the directory *app_server_root*`\samples\bin\JaxWSServicesSamples\META-INF` using the contents shown in Example 12-8.

Example 12-8 Sample clientPolicyAttachments.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<psa:PolicySetAttachment
xmlns:psa="http://www.ibm.com/xmlns/prod/websphere/200605/policysetatta
chment"
xmlns:ps="http://www.ibm.com/xmlns/prod/websphere/200605/policyset">
    <psa:PolicySetReference name="ITSO Kerberos Policy Set" id="1165">
        <psa:PolicySetBinding scope="domain" name="ITSO Consumer
Binding"/>
        <psa:Resource pattern="WebService:/"/>
    </psa:PolicySetReference>
</psa:PolicySetAttachment>
```

7. Allow the wcfconsumer user complete access to the *app_server_root* directory.

12.7 Validating the scenario

Verify that the Thin Client for JAX-WS application can interoperate with WCFService using the following steps:

1. Ensure that WCFService is available using the verification steps listed in 12.5, “Deploying the WCFService application” on page 283.
2. Ensure that the sample Thin Client for JAX-WS is set up as described in 12.6.4, “Updating the Thin Client for JAX-WS client script and policy sets” on page 300.
3. Log on to the domain system using the wcfconsumer user ID. Execute the **`kinit`** command as shown in Example 12-1 on page 279.
4. Open a command prompt, and run the **`setupCmdLine.bat`** command from the associated profiles directory.
5. Change to the following directory:

   ```
   C:\WebSphere\AppServer\samples\bin\JaxWSServicesSamples
   ```

6. Invoke the sample Thin Client for JAX-WS using the following command:

   ```
   runSampleSei.bat -h saw921-sys1.kkdc.test.com -p 81 -f
   /WCFService/EchoServicePortImpl.svc/WSSampleSei/EchoService -s
   echo -w y
   ```

7. Enter the wcfconsumer user ID and password when prompted.

   Example 12-9 shows the output of this command.

Example 12-9 Scenario validation

```
C:\WebSphere\AppServer\samples\bin\JaxWSServicesSamples>runSampleSei
.bat -h saw9
21-sys1.kkdc.test.com -p 81 -f
/WCFService/EchoServicePortImpl.svc/WSSampleSei/E
choService -s echo -w y
>> CLIENT: SEI Echo to
http://saw921-sys1.kkdc.test.com:81/WCFService/EchoServic
ePortImpl.svc/WSSampleSei/EchoService
>> CLIENT: SEI Echo invocation complete.
>> CLIENT: SEI Echo response is: DotNet==> HELLO
```

Single sign-on to WebSphere Application Server and DB2 from a Java application client

The scenario in this chapter illustrates the use of Kerberos authentication with a Java application client using single sign-on (SSO) to access WebSphere Application Server V7.0.0.5 and DB2 v9.5. It includes the following topics:

- Scenario overview
- Configuring the AIX KDC
- Configuring DB2
- Configuring WebSphere Application Server
- Configuring the JEE application client

13.1 Scenario overview

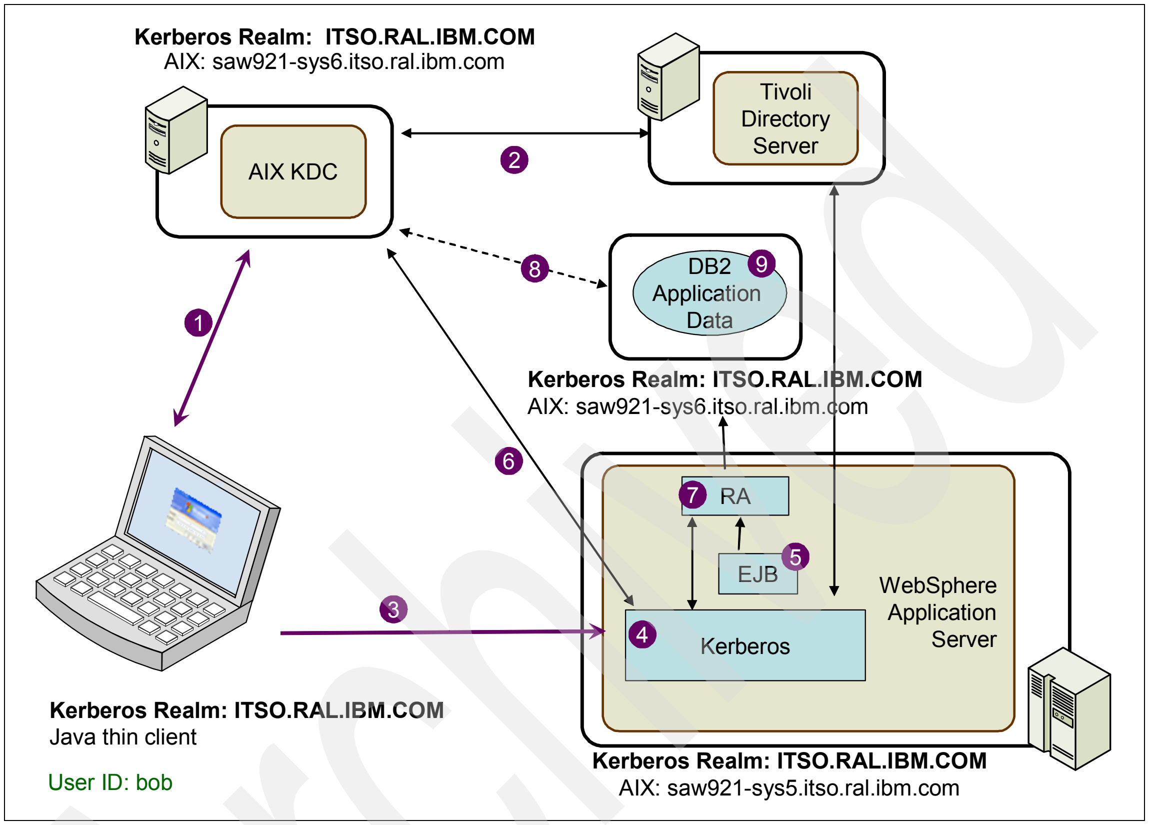
This scenario shows end-to-end propagation of the user credentials from a Java application client, through WebSphere Application Server to DB2. In the scenario, WebSphere Application Server and DB2 are configured to use Kerberos authentication, allowing the client Kerberos delegation credentials to be propagated and preserving the original requester identity.

Note: This scenario features the following components:

- SSO from a Java application client to WebSphere Application Server and then to DB2
- A Kerberos credential cache on the client
- An AIX with IBM Tivoli Directory Server
- WebSphere Application Server configuration:
  - User registry: IBM Tivoli Directory Server
  - Authentication mechanism: Kerberos and LTPA
  - Kerberos authentication and configuration file
- Sample application, krbSample, which consists of an EJB module, a Web module, and an application client module

13.1.1 Step-by-step flow of the scenario

Figure 13-1 illustrates the basic flow of this scenario.



Kerberos Realm: ITSO.RAL.IBM.COM
AIX: saw921-sys6.itso.ral.ibm.com
AIX KDC
Tivoli Directory Server
2
8
DB2 Application Data
9
1
Kerberos Realm: ITSO.RAL.IBM.COM
AIX: saw921-sys6.itso.ral.ibm.com
6
7
RA
EJB
5
WebSphere Application Server
3
4
Kerberos
Kerberos Realm: ITSO.RAL.IBM.COM
Java thin client
User ID: bob
Kerberos Realm: ITSO.RAL.IBM.COM
AIX: saw921-sys5.itso.ral.ibm.com

Figure 13-1 Scenario overview

The numbers in the figure correspond to the following steps in this process:

1. The client requests a Kerberos token (containing a ticket-granting ticket and service-granting ticket) by authenticating itself to the KDC. A detailed description of the authentication process is described in "Kerberos authentication protocol" on page 5. Two variants are possible:
   - Using a user ID and password
   - Using credential cache
2. The KDC validates the user in the principal registry, Tivoli Directory Server. The Kerberos token is returned to the client.
3. The client sends the Kerberos token to WebSphere Application Server.

4. WebSphere Application Server decrypts the token and validates the user in the Tivoli Directory Server registry.
5. If the user is authorized to use the EJB, the requested EJB method is called. The EJB calls DB2 using a configured data source and resource reference.
6. WebSphere Application Server requests a propagation token from the KDC to authenticate to DB2.
7. DB2 is invoked using a resource adapter that is configured to use Kerberos authentication. WebSphere Application Server obtains a Kerberos service ticket from the KDC for DB2 on behalf of the client and uses the client's delegated Kerberos credentials.
8. The Kerberos token returned from the KDC is added to the DB2 authentication request and sent to DB2 for authentication.
9. If the user authenticates successfully, the SQL query is executed.

13.1.2 Basic requirements for this scenario

This scenario assumes that following components are properly configured:

- The Kerberos KDC

  In this example, we use an AIX KDC that is configured to use Tivoli Directory Server as its registry.

- DB2 v9.5 installed and instance created

  To use an AIX KDC, DB2 must be installed on AIX or Linux. Our scenario uses AIX.

  Restriction: If DB2 is installed on a Microsoft Windows platform, you can only use Microsoft Active Directory as the KDC.

  If DB2 is configured on node that is remote from the KDC, the Network Authentication Service (NAS) Client must be installed and configured. For simplicity, this scenario installs DB2 on the KDC system. If you encounter a problem in installing DB2 v9.5 on an AIX system, note that installing DB2 9.5 on AIX 5.3 requires a 64-bit kernel.

- WebSphere Application Server V7.0.0.5 configured to run the JEE application that calls DB2
- WebSphere Application Server V7.0.0.5 as the Java Application Client (JAC) that connects to the JEE application

Download material: This scenario uses an application called *krbSample* that is installed on WebSphere Application Server. The krbSample application uses a DB2 database called *testdb*. The application and SQL statements that are used to configure the database are included in the Web material for this book. For more information about how to download and use this material, see Appendix E, “Additional material” on page 525.

13.1.3 Summary of the implementation steps

To implement this scenario requires the following basic steps:

1. Configure the AIX KDC, which includes creating the required principals and generating the keytab files.
2. Configure DB2 to support Kerberos authentication.
3. Configure WebSphere Application Server.

   This process is multi-step process that includes configuring the user registry, Kerberos authentication mechanism, and resource adapter.
4. Install, configure, and test the JEE Application Client.

13.2 Configuring the AIX KDC

This section describes how to configure the AIX KDC. For more information about the KDC, see Chapter 3, “Configuring IBM Network Authentication Service KDC on AIX” on page 29.

13.2.1 Creating principals

Kerberos principals are used to authenticate users and services during Kerberos authentication. This scenario require the following Kerberos principals:

- bob

  This Kerberos user principal name represents the user who authenticates to the application. This user’s credentials are passed through WebSphere Application Server to the DB2.
- db2inst1/db2inst1

  This Kerberos user principal represents the user who starts the DB2 instance.

- db2inst1/saw921-sys6.itso.ral.ibm.com

  Kerberos service principal name (SPN) that the DB2 server users. The SPN for the DB2 UDB instance must be in the following format:

  ```
  <instance name>/<fully qualified hostname>@REALM
  ```

- wasadmin

  This Kerberos user principal represents the user who is the WebSphere Application Server administrator.

- WAS/saw921-sys5.itso.ral.ibm.com

  The Kerberos SPN that WebSphere Application Server uses.

> **Note:** The Kerberos SPN for the WebSphere Application Server must be `<service>/<fully qualified hostname>@REALM`. By convention *service* is `WAS`.

> **Important:** If you have nodes on multiple hosts in the cell, you need an SPN for each host.

To create the required principals, follow these steps:

1. Log in to the KDC system.
2. Start the **kadmin** tool. To invoke the **kadmin** interface, run this command:

   ```
   /usr/krb5/sbin/kadmin -p admin/admin
   ```

   This command prompts you for the password of the Kerberos user principal (admin/admin). Enter the password to open the interface.
3. Create the first principal using the following command:

   ```
   addprinc -pw bob bob
   ```

   Example 13-1 shows the output of this command.

   Example 13-1 Creating the Kerberos principal "bob"

   ```
   kadmin:  addprinc -pw bob bob
   WARNING: no policy specified for bob@ITSO.RAL.IBM.COM;
     defaulting to no policy. Note that policy may be overridden by
     ACL restrictions.
   Principal "bob@ITSO.RAL.IBM.COM" created.
   ```

4. Repeat step 3 to create the remainder of the principals.

For more details about creating Kerberos principals see, 3.3.2, “Creating a Kerberos principal” on page 42.

13.2.2 Generating Kerberos keytab files

The Kerberos keytab file stores the Kerberos SPNs and keys that are used to validate the incoming Kerberos token request. The keytab file is created using tools provided by KDC.

In this scenario, two keytab files are required:

- One keytab file for the WebSphere Application Server service principal
- One keytab file for the DB2 service principal

Creating a keytab file for the WebSphere Application Server service principal

To create the keytab file for WebSphere Application Server service principal, follow these steps:

1. Log in to the KDC system.
2. Start the **kadmin** tool. To invoke the **kadmin** interface, run this command:

   ```
   /usr/krb5/sbin/kadmin -p admin/admin
   ```

   This command prompts you for the password of the Kerberos principal (admin/admin). Enter the password to open the interface.
3. Create the keytab file using the following command:

   ```
   kadmin:  ktadd -k /etc/krb5/was.keytab -e arcfour-hmac:normal
   WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM
   ```

   Example 13-2 shows the output of the command.

   Example 13-2 Generating keytab for WebSphere Application Server

   ```
   kadmin:  ktadd -k /etc/krb5/was.keytab -e arcfour-hmac:normal
   WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM
   Entry for principal
   WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM with kvno 2,
   encryption type ArcFour with HMAC/md5 added to keytab
   WRFILE:/etc/krb5/was.keytab.
   ```

   This command generates a new Kerberos keytab file that is stored in `/etc/krb5/was.keytab`. This file includes the Kerberos SPN and its `arcfour-hmac` key.

Tip: Because this keytab is transferred to the WebSphere Application Server system, which is usually a separate system from the KDC system, and because this keytab is used only to store Kerberos SPNs and keys that are used by the application server, it is a good practice to create the keytab in a separate file instead of the default file.

Note: If you have nodes on multiple hosts, you need to add keys for all Kerberos SPNs to one keytab file by specifying the same keytab file in the **`ktadd`** commands.

WebSphere Application Server v7.0.0.5 supports the following encryption types:

- des-cbc-md5
- des-cbc-crc
- des3-cbc-sha1
- rc4-hmac
- arcfour-hmac
- arcfour-hmac-md5
- aes128-cts-hmac-sha1-96
- aes256-cts-hmac-sha1-96

Caution: Ensure that you have a common encryption type for the Kerberos configuration file, the Kerberos keytab file, the Kerberos SPN, and the Kerberos client.

4. Copy the new Kerberos keytab file to the WebSphere Application Server system.

Creating a keytab file for the DB2 Kerberos service principal

To create the Kerberos keytab file for the DB2 Kerberos service principal, follow these steps:

1. Start the **`kadmin`** tool. To invoke the kadmin interface, run this command:

   ```
   /usr/krb5/sbin/kadmin -p admin/admin
   ```

   This command prompts you for the password of the Kerberos principal (admin/admin). Enter the password to open the command line interface.

2. Create the keytab file using the following command:

   ```
   kadmin:  ktadd -e arcfour-hmac:normal
   db2inst1/saw921-sys6.itso.ral.ibm.com
   Entry for principal db2inst1/saw921-sys6.itso.ral.ibm.com with
   kvno 2, encryption type ArcFour with HMAC/md5 added to keytab
   WRFILE:/etc/krb5/krb5.keytab.
   ```

   Example 13-3 shows the output of this command.

   Example 13-3 Generating the keytab file for DB2

   ```
   kadmin:  ktadd -e arcfour-hmac:normal
   db2inst1/saw921-sys6.itso.ral.ibm.com
   Entry for principal db2inst1/saw921-sys6.itso.ral.ibm.com with kvno
   2, encryption type ArcFour with HMAC/md5 added to keytab
   WRFILE:/etc/krb5/krb5.keytab.
   ```

   This command adds Kerberos SPN and its key for the DB2 server to the default keytab file. This is appropriate in the scenario because DB2 is located on the same system as the KDC.

   Note: This step is not required for DB2 on a Microsoft Windows platform, because the system automatically handles storing and acquiring credentials from Microsoft Active Directory.

For more details about creating keytabs, see 3.3.3, “Generating a keytab” on page 44.

13.3 Configuring DB2

The Kerberos prerequisites for DB2 are as follows:

- For the AIX and Solaris operating environments and for Linux platforms, the IBM Network Authentication Service (NAS) Toolkit v1.4 or higher is required. You can download NAS Toolkits at:

  https://www6.software.ibm.com/dl/dm/dm-nas-p

- For the Windows platforms, there are no prerequisites.

Restriction: DB2 for the Windows platform can use only Microsoft Active Directory as a KDC.

In this scenario, we use AIX and assume that the KDC is already configured (see Chapter 3, "Configuring IBM Network Authentication Service KDC on AIX" on page 29 for details). Because the DB2 server is on the same system as the KDC, a separate NAS configuration is not required.

To configure DB2 for Kerberos authentication, follow these steps:

1. Prepare the principals. (This step is described in 13.2.1, "Creating principals" on page 309).
2. Enable Kerberos authentication in DB2.
   a. Log in as the database instance owner, and execute the following command:

      ```
      #su - db2inst1
      ```

   b. Change the database manager configuration.

      Update the Server Connection Authentication (`SRVCON_AUTH`) parameter with `KERBEROS` or `KRB_SERVER_ENCRYPT` as the value as follows:

      ```
      $update dbm cfg using SRVCON_AUTH KERBEROS
      ```

      A value of `KERBEROS` means that authentication is performed using the Kerberos security protocol for authentication. With an authentication type of `KRB_SERVER_ENCRYPT`, if the clients do not support Kerberos, the authentication type is effectively equivalent to `SERVER_ENCRYPT`.

   Note: You can find more details about different authentication methods at:

   http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.admin.sec.doc/doc/c0005435.html

3. DB2 uses the default Kerberos configuration and keytab files that are located in the `/etc/krb5` directory. These files can be read only by the root user. To ensure that the instance owner (`db2inst1`) has read access to these files, execute the following commands from the root account:

   ```
   #cd /etc/krb5
   #chmod a+r krb5*
   ```

4. Restart the DB2 instance. After changes in the database configuration, you must restart DB2. Log in as the db2inst1 user, and complete the following steps:

   a. Issue the following command to stop DB2:

      ```
      bash-2.05b$ db2stop
      ```

      Example 13-4 shows the output of this command.

      Example 13-4 Stopping the DB2

      ```
      bash-2.05b$ db2stop
      07/16/2009 06:55:22     0   0   SQL1064N  DB2STOP processing was
      successful.
      SQL1064N  DB2STOP processing was successful.
      ```

   b. Issue the following command to get the Kerberos credentials for users who will start the server:

      ```
      bash-2.05b$/usr/krb5/bin/kinit db2inst1/db2inst1
      Password for db2inst1/db2inst1@ITSO.RAL.IBM.COM:
      ```

   c. Verify that the ticket was created using the following command.

      ```
      bash-2.05b$/usr/krb5/bin/klist
      ```

      Example 13-5 shows the output of this command.

      Example 13-5 Checking Kerberos ticket

      ```
      bash-2.05b$ /usr/krb5/bin/klist
      Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_1004
      Default principal:  db2inst1/db2inst1@ITSO.RAL.IBM.COM

      Valid starting     Expires            Service principal
      07/16/09 06:50:52  07/17/09 06:50:47
      krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
      ```

   d. Start the DB2 server by issuing the following command. Example 13-6 shows the output of this command.

      ```
      bash-2.05b$db2start
      ```

      Example 13-6 Starting the DB2

      ```
      bash-2.05b$ db2start
      07/16/2009 06:55:30     0   0   SQL1063N  DB2START processing was
      successful.
      SQL1063N  DB2START processing was successful.
      ```

5. Validate the Kerberos authentication.

   a. Create a Kerberos ticket for the client or instance starter using the following command:

   ```
   bash-2.05b$/usr/krb5/bin/kinit bob
   Password for bob@ITSO.RAL.IBM.COM:
   ```

   b. Verify that the ticket was created by issuing the following command. Example 13-7 on page 316 shows the output of this command.

   ```
   bash-2.05b$/usr/krb5/bin/klist
   ```

   Example 13-7 Verify that the ticket was created

   ```
   bash-2.05b$ /usr/krb5/bin/klist
   Ticket cache:  FILE:/var/krb5/security/creds/krb5cc_0
   Default principal:  bob@ITSO.RAL.IBM.COM

   Valid starting     Expires            Service principal
   08/14/09 05:29:58  08/15/09 05:29:55
   krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
   ```

   c. Connect to the database using the Kerberos credentials as follows:

   ```
   bash-2.05b$ db2 connect to testdb
   ```

   Example 13-8 shows the output of this command.

   Example 13-8 Connect to the database

   ```
   bash-2.05b$ db2 connect to testdb

      Database Connection Information

   Database server        = DB2/AIX64 9.5.0
   SQL authorization ID   = BOB
   Local database alias   = TESTDB
   ```

You have successfully configured and validated Kerberos authentication to DB2.

For more details about configuring Kerberos in DB2, see the following resources:

- IBM developerWorks article

  http://www.ibm.com/developerworks/data/library/techarticle/dm-0603see/index.html#db2kerb

- IBM DB2 Database for Linux, UNIX, and Windows Information Center

  http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/topic/com.ibm.db2.luw.admin.sec.doc/doc/c0011990.html

13.4 Configuring WebSphere Application Server

Next, we describe how to configure WebSphere Application Server.

Note: Before you begin the configuration, ensure that you are using WebSphere Application Server V7.0.0.5 or later. Earlier versions do not fully support Kerberos authentication.

13.4.1 Configuring Kerberos authentication

The following items are required before you can configure Kerberos as the authentication mechanism using the administrative console:

- A Kerberos keytab file that contains the Kerberos service principal. This file was generated for `WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM` in 13.2.2, "Generating Kerberos keytab files" on page 311.
- The Kerberos configuration file (`krb5.ini` or `krb5.conf`)

  We explain how to create this file in the next step.

Creating the Kerberos configuration file

You can create the Kerberos configuration file using several different methods (such as manual, tools, and so forth); however, the preferred method is to use the **wsadmin** utility. In this section, we describe how to create the Kerberos configuration file using **wsadmin**.

To create the Kerberos configuration file, follow these steps:

1. Go to the Deployment Manager profile bin directory (*profile_root*`\bin`).
2. Start the **wsadmin** tool by issuing the following command:

   ```
   [saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
   n]# ./wsadmin.sh
   ```

3. Then, issue the following command:

   ```
   wsadmin>$AdminTask createKrbConfigFile {-krbPath /tmp/krb5.conf
   -realm ITSO.RAL.IBM.COM -kdcHost saw921-sys6.itso.ral.ibm.com
   -dns itso.ral.ibm.com -keytabPath /tmp/was.keytab -encryption
   arcfour-hmac }
   ```

In this command, the parameters are as follows:

krbPath	The directory location and file name of the Kerberos keytab file
realm	The Kerberos realm name
kdcHost	The KDC host name
dns	The DNS domain name
keytabPath	The directory location and file name of the Kerberos keytab file
encryption	The encryption type used in the keytab file

Example 13-9 shows the output of the command.

Example 13-9 Kerberos configuration file

```
[libdefaults]
        default_realm = ITSO.RAL.IBM.COM
        default_keytab_name = FILE:/tmp/was.keytab
        default_tkt_enctypes = arcfour-hmac
        default_tgs_enctypes = arcfour-hmac
        forwardable  = true
        renewable  = true
        noaddresses = true
        clockskew  = 300
[realms]
        ITSO.RAL.IBM.COM = {
                kdc = saw921-sys6.itso.ral.ibm.com:88
                default_domain = itso.ral.ibm.com
        }
[domain_realm]
        .itso.ral.ibm.com = ITSO.RAL.IBM.COM
```

4. Copy the Kerberos configuration file (`krb5.conf`) and keytab (`was.keytab`) to the following directory:

   dmgr_profile_root/config/cells/*cellname*

   In this scenario, the directory is as follows:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/config/cells/saw
   921-sys5Cell01
   ```

Tip: Because all servers in the cell must share the configuration and keytab files, put the files in the deployment manager's *config*/*cell*/*cellName* directory. This directory is replicated throughout all nodes.

Configuring the authentication mechanism

Now that the configuration file and keytab file are ready, you can configure Kerberos authentication in the application server as follows:

1. Open the administrative console.
2. In the navigation tree click **Security** → **Global Security**.
3. In the Authentication section, click **Kerberos configuration**.
4. Specify the following values, as shown in Figure 13-2:
   - Kerberos service name:

     ```
     WAS
     ```

     This entry must match the *`service`* part from the Kerberos SPN for WebSphere Application Server. In this, scenario the principal name is `WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM` and the *`service`* is `WAS`.
   - Kerberos configuration file:

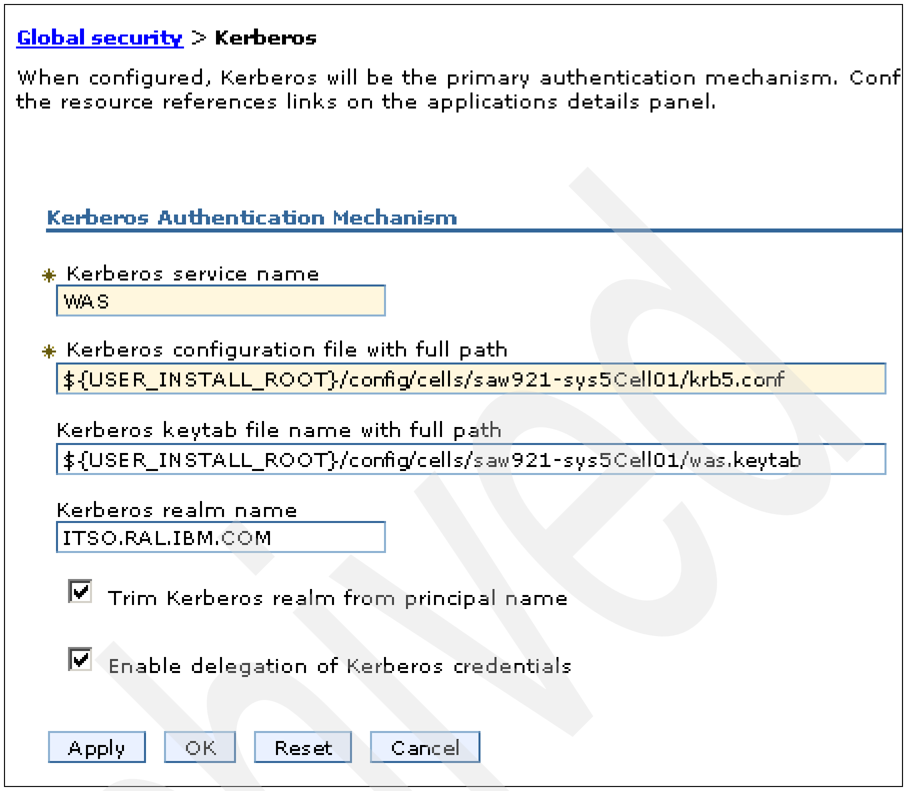
     ```
     ${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/krb5.conf
     ```
   - Kerberos keytab file:

     ```
     ${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/was.keytab
     ```
   - Kerberos realm name:

     ```
     ITSO.RAL.IBM.COM
     ```
   - Select the “Trim Kerberos realm from principal name” option

     This option specifies whether Kerberos removes the suffix of the principal user name, starting from the at sign (@) that precedes the Kerberos realm name. If this attribute is set to `true`, the suffix of the principal user name is removed. If this attribute is set to `false`, the suffix of the principal name is retained.
   - Select the “Enable delegation of Kerberos credentials” option

     This option specifies whether the Kerberos delegated credentials are stored in the subject by the Kerberos authentication. It also enables an application to retrieve the stored credentials and to propagate them to other applications downstream for additional Kerberos authentication with the credential from the Kerberos client.


Global security > Kerberos

When configured, Kerberos will be the primary authentication mechanism. Conf
the resource references links on the applications details panel.

Kerberos Authentication Mechanism

* Kerberos service name
WAS

* Kerberos configuration file with full path
${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/krb5.conf

Kerberos keytab file name with full path
${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/was.keytab

Kerberos realm name
ITSO.RAL.IBM.COM

[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

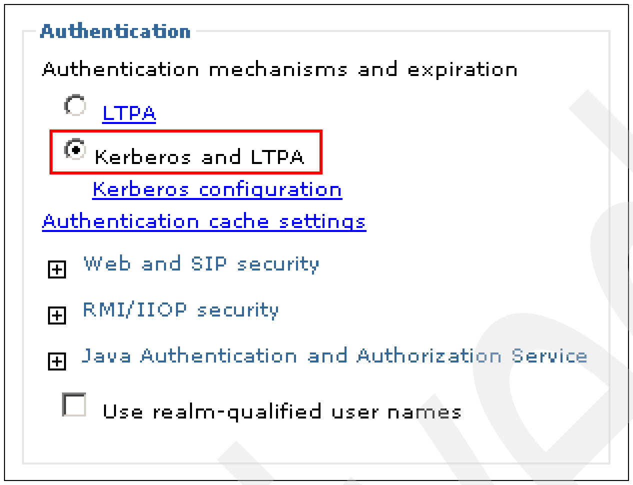
Figure 13-2 Defining authentication properties

Tip: You can use the WebSphere variable `${USER_INSTALL_ROOT}` in the path names to ensure that all nodes will have access to the Kerberos configuration file and keytab file.

It is also recommended that you specify the path to the keytab file instead of relying on the default path that is stored in the configuration file.

5. Click **OK**.

6. Back on the Global Security page, select **Kerberos and LTPA** in the Authentication section, as shown in Figure 13-3.


Authentication

Authentication mechanisms and expiration

- ○ LTPA
- ● Kerberos and LTPA
  - Kerberos configuration

Authentication cache settings

⊞ Web and SIP security

⊞ RMI/IIOP security

⊞ Java Authentication and Authorization Service

☐ Use realm-qualified user names

Figure 13-3 Selecting authentication mechanism

7. Click **Apply**, and then click **Save**.
8. Enable the Kerberos traces for the validation step. See “Enabling the JGSS and KRB traces” on page 469 for instructions about how to enable these traces. (Remember to disable the traces after you are satisfied that everything is working as expected).

You can find more information about how to configure Kerberos authentication in the information center:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/tsec_kerb_auth_mech.html`

13.4.2 Validating the Kerberos authentication

To validate Kerberos authentication, follow these steps:

1. Log in to the administrative console using the wasadmin user ID. This time, provide the Kerberos user principal password instead of the password stored in the WebSphere file-based registry (if the passwords are different).
2. Check the `SystemOut.log` file for Kerberos authentication debugging messages. Example 13-10 shows a partial output for this log.

Example 13-10 Kerberos authentication debug

```
...
[7/17/09 5:34:16:304 EDT] 00000017 SystemOut     O [KRB_DBG_KDC]
Credentials:WebContainer : 1:Client Name:wasadmin@ITSO.RAL.IBM.COM
[7/17/09 5:34:16:305 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
Doing Kerberos login for principal wasadmin@ITSO.RAL.IBM.COM
[....
[7/17/09 5:34:18:092 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
Kerberos login complete
[7/17/09 5:34:18:093 EDT] 00000017 SystemOut     O [KRB_DBG_KDC]
Credentials:WebContainer : 1:Client Name:wasadmin@ITSO.RAL.IBM.COM
[7/17/09 5:34:18:095 EDT] 00000017 SystemOut     O [KRB_DBG_KDC]
Credentials:WebContainer : 1: Session Key is Only Service Key
[7/17/09 5:34:18:095 EDT] 00000017 SystemOut     O [KRB_DBG_KDC]
Credentials:WebContainer : 1: Session Key is Only Service Key
[7/17/09 5:34:18:098 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
Login successful
[7/17/09 5:34:18:099 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
Saved principal wasadmin@ITSO.RAL.IBM.COM in shared state
[7/17/09 5:34:18:099 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
Saved password in shared state
[7/17/09 5:34:18:822 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
wasadmin@ITSO.RAL.IBM.COM added to Subject
[7/17/09 5:34:18:823 EDT] 00000017 SystemOut     O [JGSS_DBG_CRED]
Kerberos ticket for wasadmin@ITSO.RAL.IBM.COM added to Subject
.....
```

3. Disable Kerberos debugging after you validate the authentication, because it logs sensitive information and can have a negative impact on application server performance.

13.4.3 Configuring the DB2 data source

Creating the testdb database: The krbSample application uses the testdb database. Create a new database on DB2 called *testdb*. Then use the SQL statements that are available in the materials that are included with this book to configure the database. You can find these statements in the `appdb.spl` file in the `krbSample` folder. For more information about the additional materials included with this book, see Appendix E, “Additional material” on page 525.

In this section, we explain how to configure the DB2 data source to authenticate using Kerberos credentials.

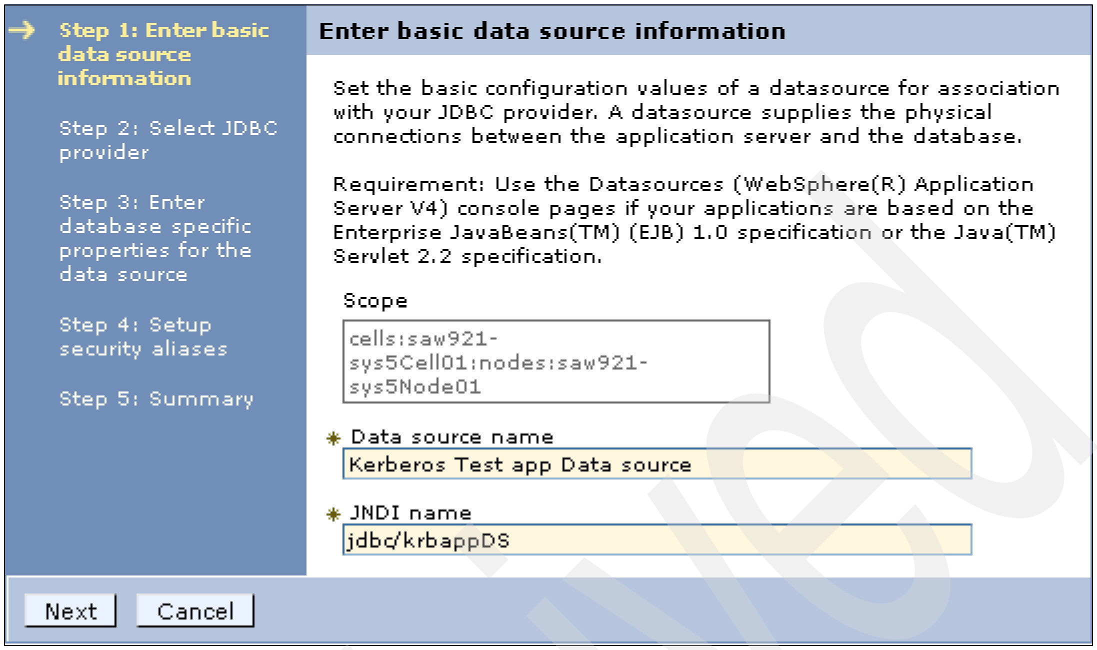
You must use a DB2 JDBC driver that supports Kerberos authentication and is operating in type 4 mode. The supported JDBC drivers include:

- IBM Data Server Driver for JDBC and SQLJ (identified in the application server as *DB2 using IBM JCC Driver*)
- IBM DB2 JDBC Universal Driver Architecture (identified in the application server as *DB2 Universal JDBC Driver Provider*)

In this scenario, we used IBM Data Server Driver for JDBC and SQLJ.

To configure the data source, follow these steps:

1. Log in to the WebSphere Application Server administrative console.
2. In the navigation tree, click **Resources** → **JDBC** → **Data Source**.
3. Select the required scope, for example `Node=<your node name>`, and click **New**.
4. Provide the data source name and the JNDI name as shown in Figure 13-4, and click **Next**.


Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Next
Cancel

Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.

Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

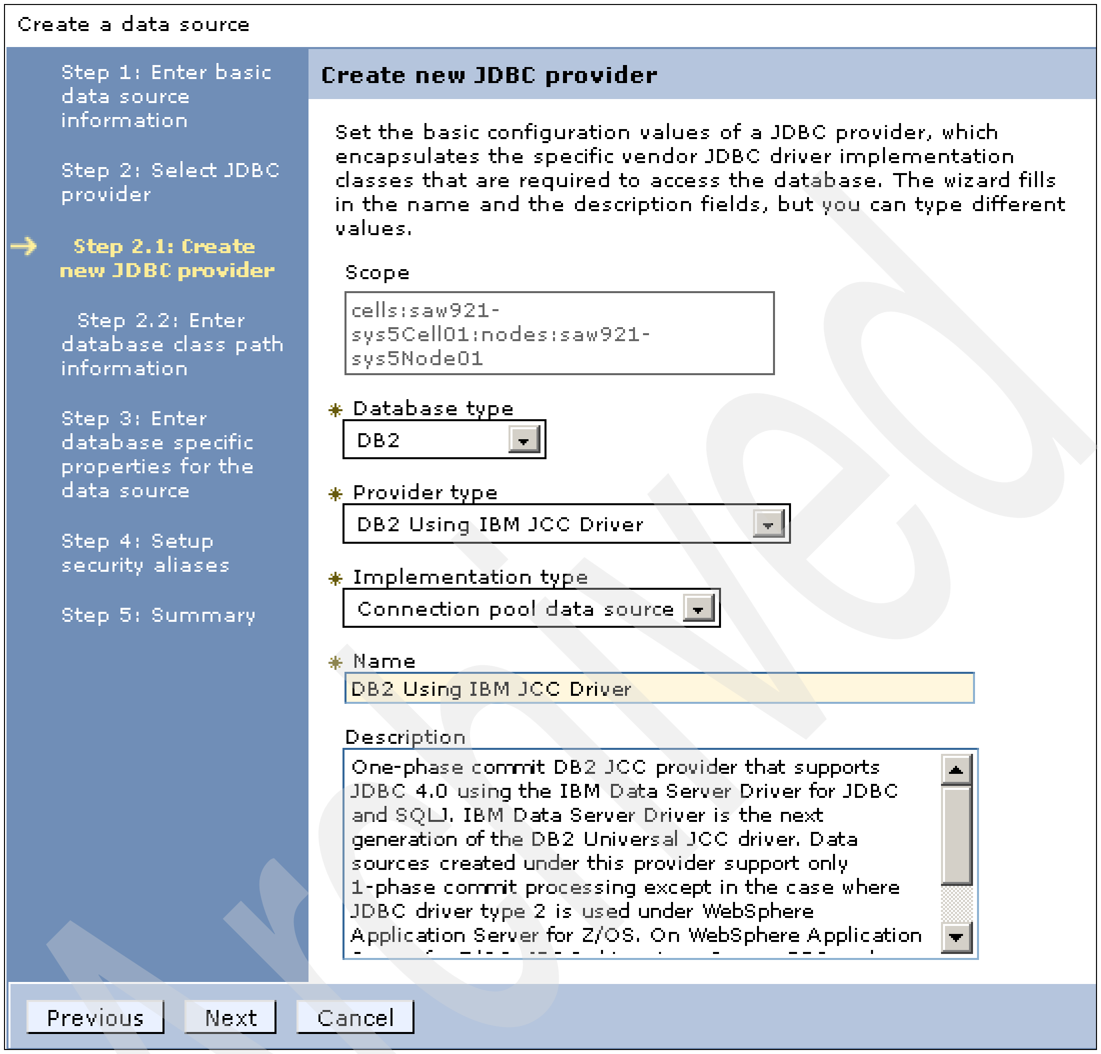
Scope
cells:saw921-sys5Cell01:nodes:saw921-sys5Node01

* Data source name
Kerberos Test app Data source

* JNDI name
jdbc/krbappDS

Figure 13-4 Creating the data source

5. Select **Create new JDBC Provider**, and click **Next**.
6. Provide the JDBC provider details as shown in Figure 13-5, and click **Next**.


Create a data source

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 2.1: Create new JDBC provider

Step 2.2: Enter database class path information

Step 3: Enter database specific properties for the data source

Step 4: Setup security aliases

Step 5: Summary

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope
cells:saw921-sys5Cell01:nodes:saw921-sys5Node01

* Database type
DB2

* Provider type
DB2 Using IBM JCC Driver

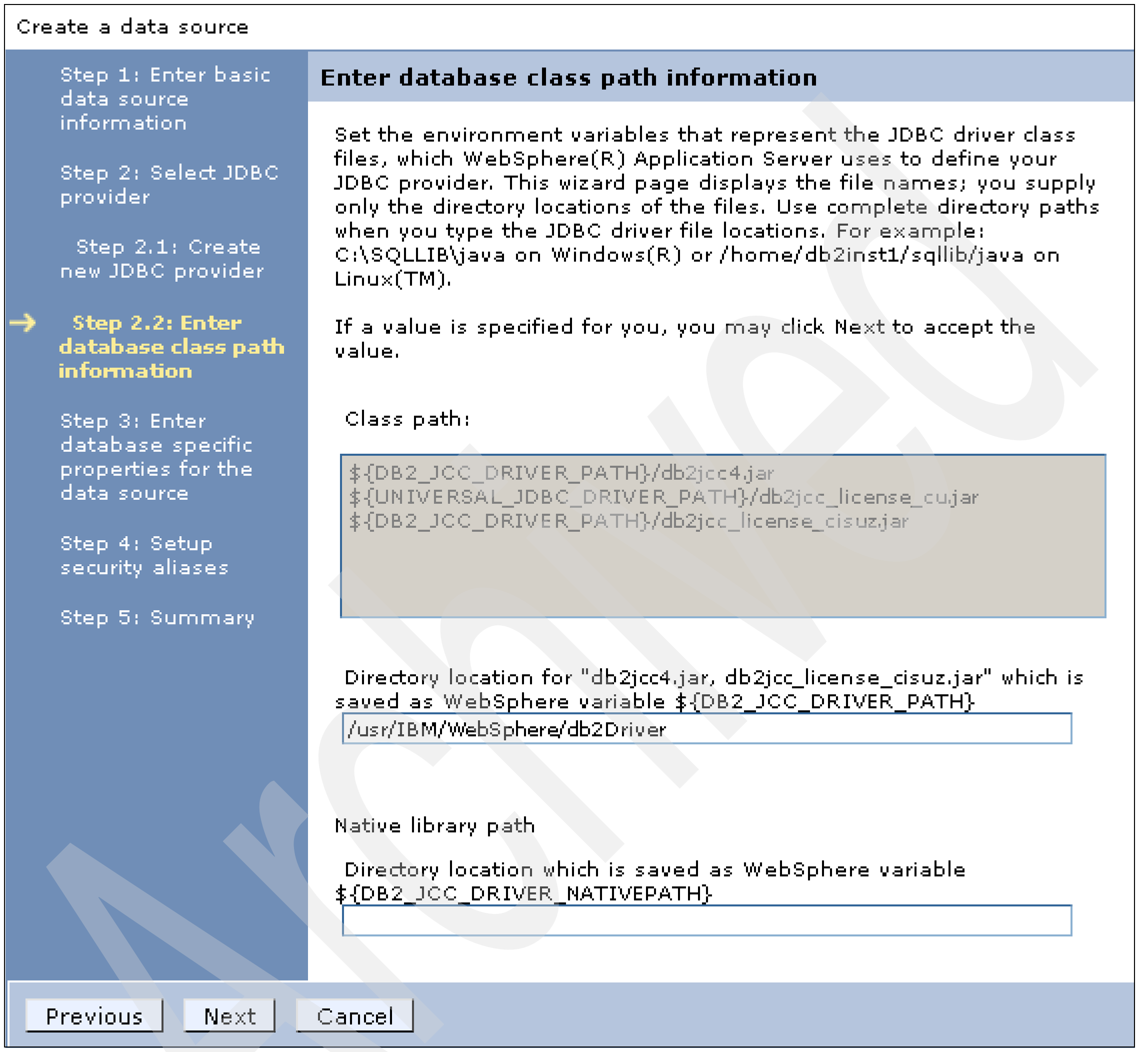
* Implementation type
Connection pool data source

* Name
DB2 Using IBM JCC Driver

Description
One-phase commit DB2 JCC provider that supports JDBC 4.0 using the IBM Data Server Driver for JDBC and SQLJ. IBM Data Server Driver is the next generation of the DB2 Universal JCC driver. Data sources created under this provider support only 1-phase commit processing except in the case where JDBC driver type 2 is used under WebSphere Application Server for Z/OS. On WebSphere Application

Figure 13-5 JDBC provider basic parameters

7. Enter the directory location for the DB2 JDBC driver as shown in Figure 13-6. The driver `.jar` files must be copied from the DB2 server. Click **Next**.



Create a data source

Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 2.1: Create new JDBC provider
Step 2.2: Enter database class path information
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary

Enter database class path information

Set the environment variables that represent the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays the file names; you supply only the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

If a value is specified for you, you may click Next to accept the value.

Class path:

```
${DB2_JCC_DRIVER_PATH}/db2jcc4.jar
${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar
${DB2_JCC_DRIVER_PATH}/db2jcc_license_cisuz.jar
```

Directory location for "db2jcc4.jar, db2jcc_license_cisuz.jar" which is saved as WebSphere variable ${DB2_JCC_DRIVER_PATH}

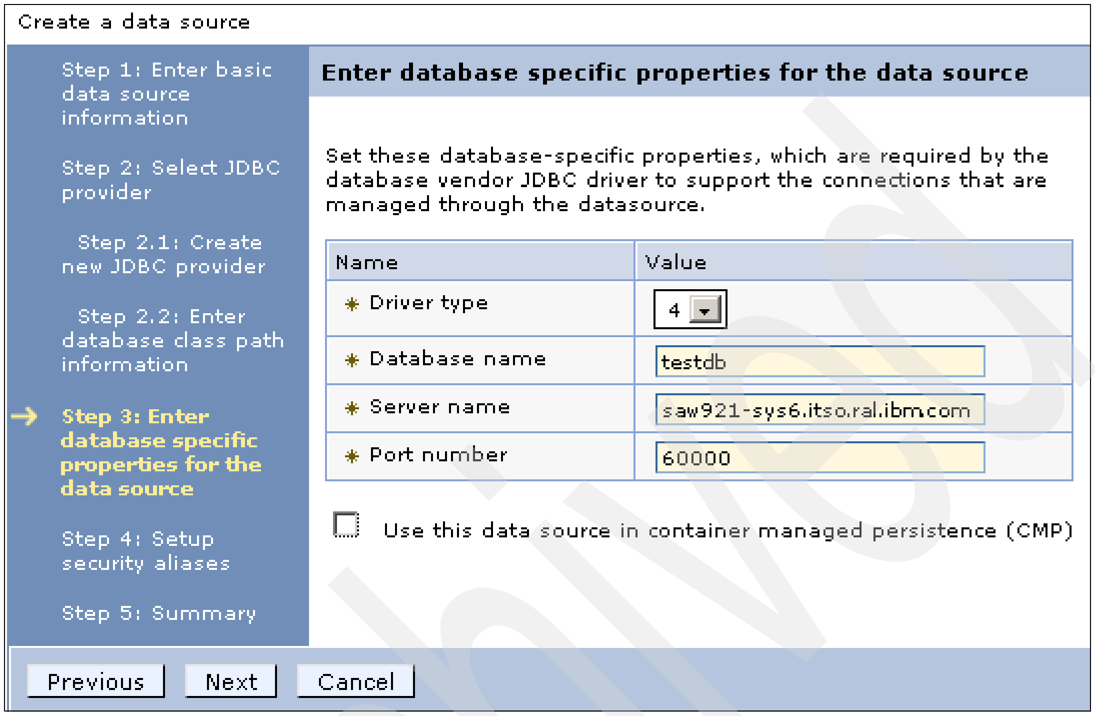
/usr/IBM/WebSphere/db2Driver

Native library path

Directory location which is saved as WebSphere variable ${DB2_JCC_DRIVER_NATIVEPATH}

Figure 13-6 Defining driver location

8. Enter the database specific properties as shown in Figure 13-7, and click **Next**.


Create a data source

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 2.1: Create new JDBC provider

Step 2.2: Enter database class path information

→ **Step 3: Enter database specific properties for the data source**

Step 4: Setup security aliases

Step 5: Summary

Enter database specific properties for the data source

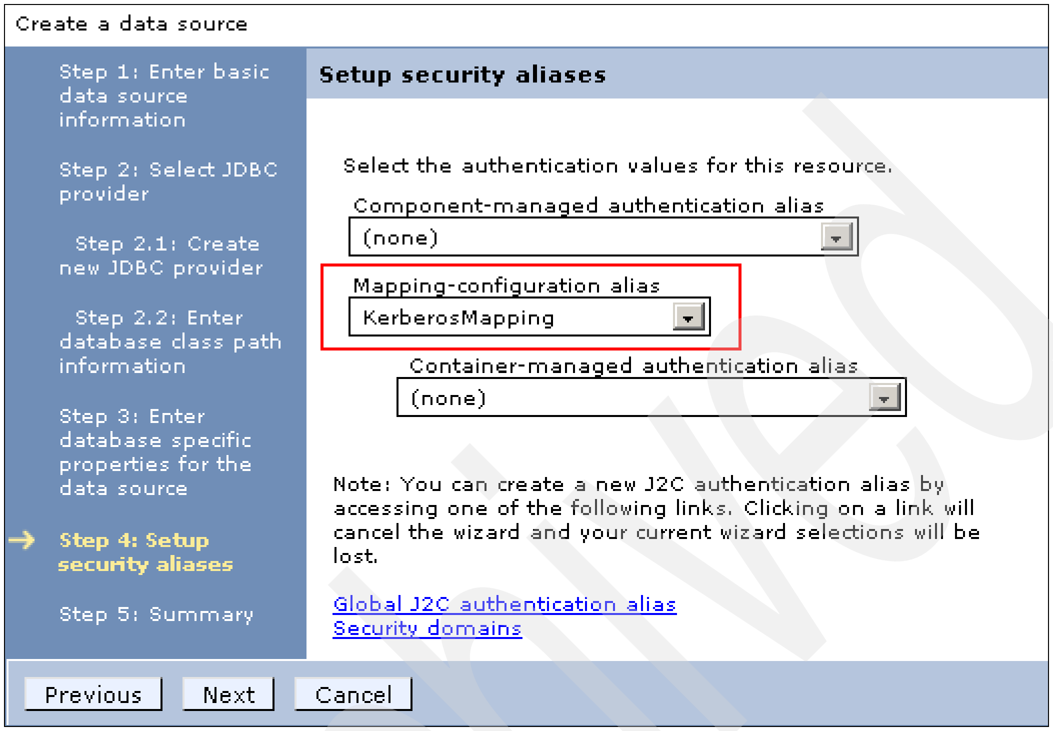
Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

Name	Value
* Driver type	4
* Database name	testdb
* Server name	saw921-sys6.itso.ral.ibm.com
* Port number	60000

☐ Use this data source in container managed persistence (CMP)

Figure 13-7 Defining database details

9. Configure the security as shown in Figure 13-8. Be sure to select **KerberosMapping** as the Mapping-configuration alias, and click **Next**.


Create a data source
Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 2.1: Create new JDBC provider
Step 2.2: Enter database class path information
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Previous
Next
Cancel

Setup security aliases

Select the authentication values for this resource.

Component-managed authentication alias: (none)

Mapping-configuration alias: KerberosMapping

Container-managed authentication alias: (none)

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

Global J2C authentication alias
Security domains

Figure 13-8 Defining security configuration

The XARecovery (in the case of an XA JDBC driver) and TestConnection facilities of the application server cannot supply delegated Kerberos credentials to the data source. Also, some applications that use this data source might not be able to provide Kerberos credentials. In these cases, the resource adapter can revert to connection authentication using Default Principle Mapping. To configure this fallback, select an alias from the container-managed authentication alias list. To disable this fallback, select **(none)** from the container-managed authentication alias list.

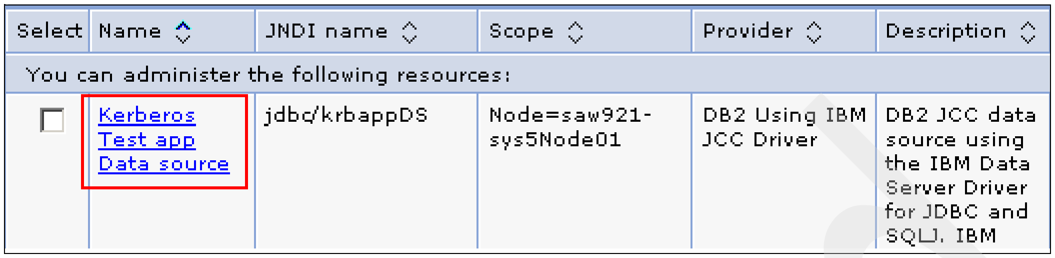
When the alias is selected, WebSphere Application Server still communicates with DB2 using Kerberos; however, it uses credentials that are defined in the alias if the application does not provide delegated credentials.

To enable the application to delegate Kerberos credentials, see 13.4.4, “Installing and configuring the krbSample application” on page 330.

10. Click **Next**, and then click **Finish** to complete data source definition.

11. Save the changes.

12. Next, change the two data source custom properties. In the Data sources list window, click the new data source, as shown in Figure 13-9.



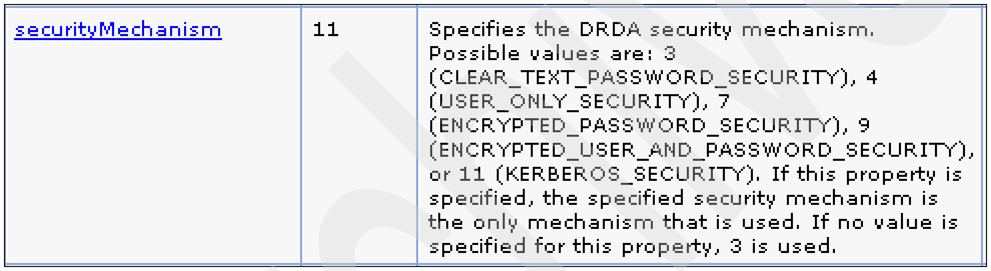
Select	Name	JNDI name	Scope	Provider	Description
You can administer the following resources:					
☐	Kerberos Test app Data source	jdbc/krbappDS	Node=saw921-sys5Node01	DB2 Using IBM JCC Driver	DB2 JCC data source using the IBM Data Server Driver for JDBC and SQLJ. IBM

Figure 13-9 Selecting data source

13. Click **Custom properties**, and then change the following properties:

   – Set `securityMechanism` to `11`.

   This property is usually at the bottom of the first page) as shown in the Figure 13-10.



securityMechanism	11	Specifies the DRDA security mechanism. Possible values are: 3 (CLEAR_TEXT_PASSWORD_SECURITY), 4 (USER_ONLY_SECURITY), 7 (ENCRYPTED_PASSWORD_SECURITY), 9 (ENCRYPTED_USER_AND_PASSWORD_SECURITY), or 11 (KERBEROS_SECURITY). If this property is specified, the specified security mechanism is the only mechanism that is used. If no value is specified for this property, 3 is used.

Figure 13-10 Specifying security mechanism

   – Set `kerberosServerPrincipal` to *<db2 service principal name>*.

   In this scenario, the DB 2 SPN is `db2inst1/saw921-sys6.itso.ral.ibm.com@ITSO.RAL.IBM.COM`. This property is usually in the middle of the second page.

14. Save the changes.

For more information about configuring data sources, see the information center:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/tdat_db2kerberos.html

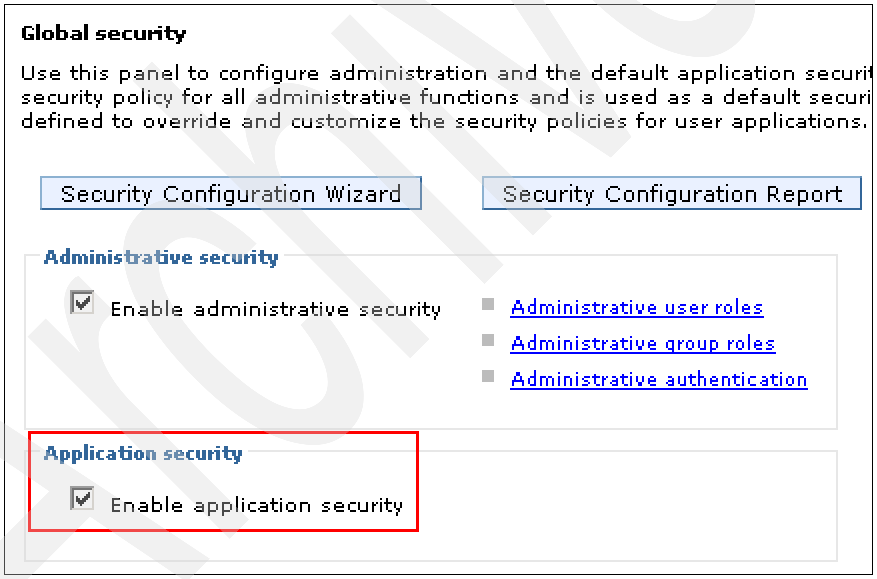
13.4.4 Installing and configuring the krbSample application

Applications that need to propagate user credentials to the database must use a resource reference. This reference can be defined either through the deployment descriptor or with an @Resource annotation.

The sample application created for this scenario consists of a Web module, EJB module, and application client module. A resource reference is defined in the EJB stateless session bean using annotation.

You can configure the resource reference either during the application installation or afterwards using the Resource references link in the application properties page.

Before installing the application, ensure that application security is enabled. To validate this setting, click **Security** → **Global Security**. Make sure that the "Enable administrative security" option is selected, as shown in Figure 13-11. After enabling application security, you need to restart all the servers.


Global security

Use this panel to configure administration and the default application securit
security policy for all administrative functions and is used as a default securi
defined to override and customize the security policies for user applications.

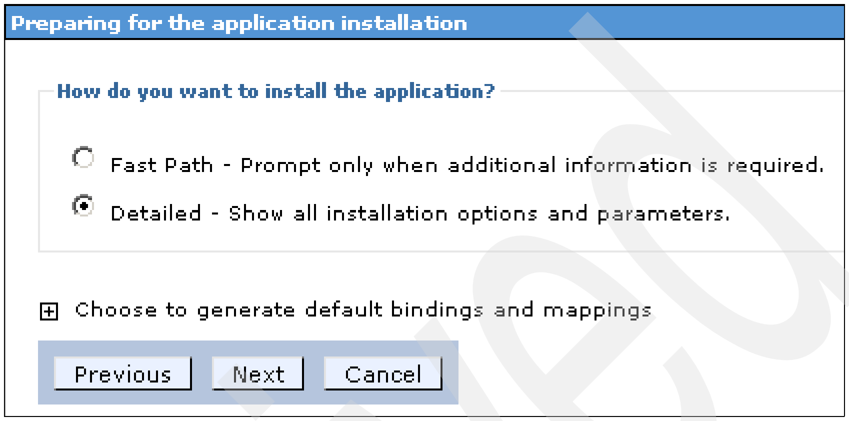
Security Configuration Wizard
Security Configuration Report
Administrative security
[✓] Enable administrative security
Administrative user roles
Administrative group roles
Administrative authentication
Application security
[✓] Enable application security

Figure 13-11 Validating Application security setting

To install the krbSample application, follow these steps:

1. In the administrative console, click **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click **Install**.

3. Click **Browse**, and select the `.ear` file for the application (`KrbSample.ear` in this example). Click **Next**.
4. In the “Preparing for the application installation” window, select the “Detailed” option as shown in Figure 13-12, and click **Next**.


Preparing for the application installation

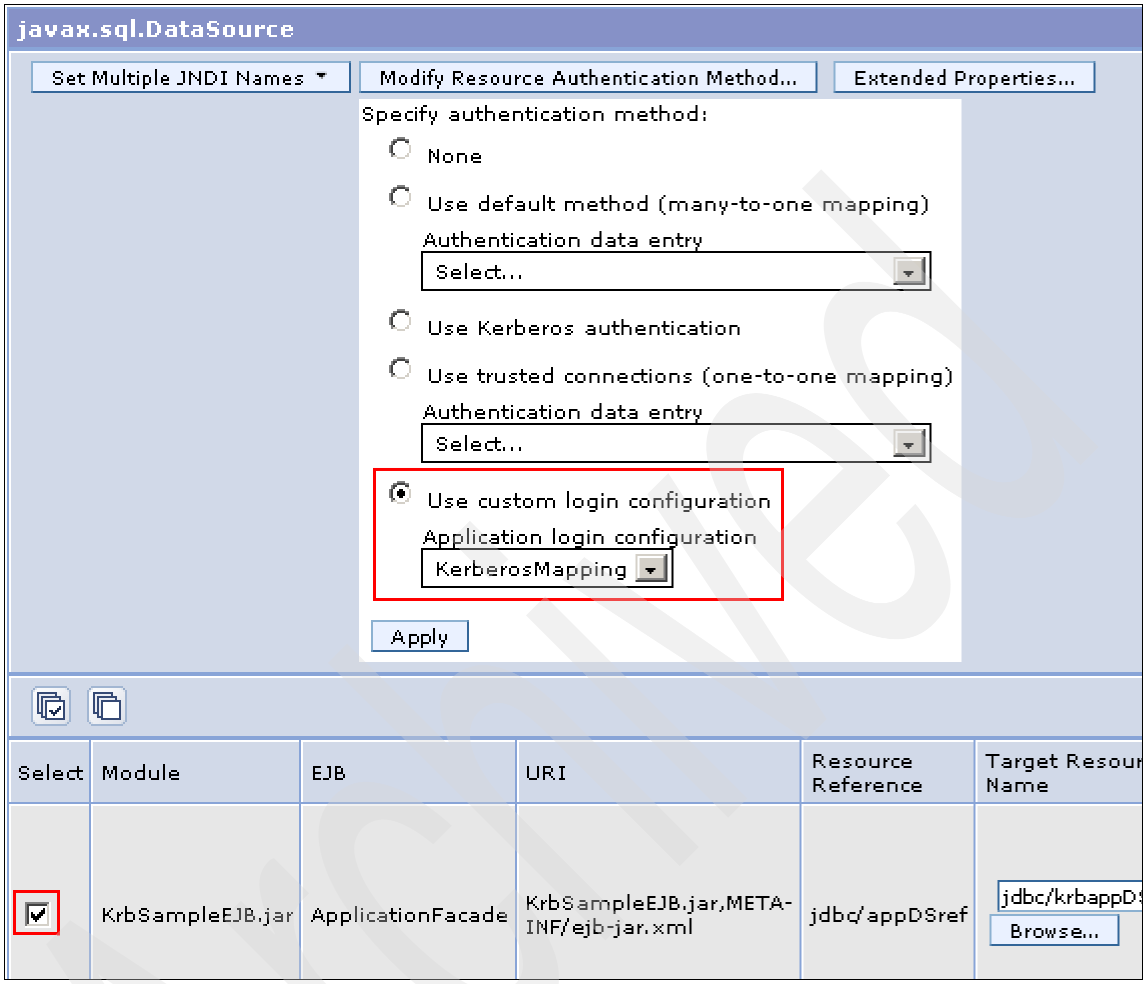
How do you want to install the application?

[ ] Fast Path - Prompt only when additional information is required.
[✓ Detailed - Show all installation options and parameters.]

⊞ Choose to generate default bindings and mappings

Figure 13-12 Selecting installation option

5. Complete the appropriate panels for your application installation. When you reach the “Map resource references to resources” panel, shown in Figure 13-13, complete the following steps:
   a. Click **Browse**, and select the data source that was defined earlier (Kerberos Test app Data source). Then, click **Apply**.
   b. Click **Modify Resource Authentication Method**.
   c. Click **Use custom Login configuration**, and then select **KerberosMapping** as the application login configuration.
   d. Select the check box in the resource reference list.
   e. Click **Apply**.


javax.sql.DataSource

Set Multiple JNDI Names | Modify Resource Authentication Method... | Extended Properties...

Specify authentication method:

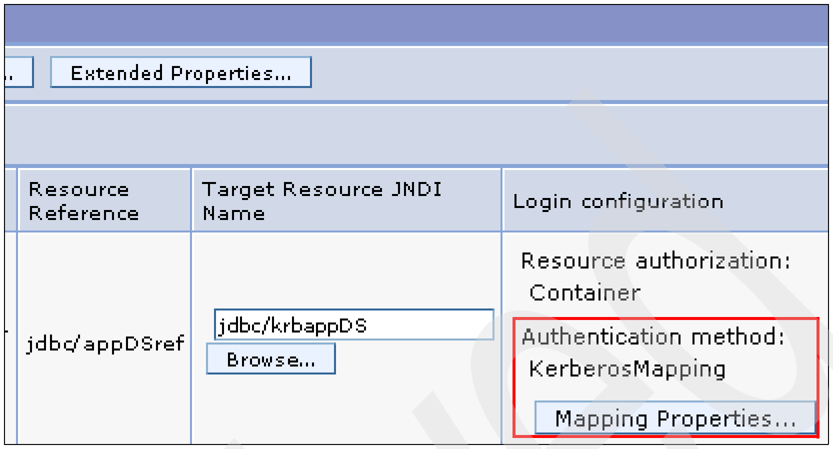
- ○ None
- ○ Use default method (many-to-one mapping)
  Authentication data entry: Select...
- ○ Use Kerberos authentication
- ○ Use trusted connections (one-to-one mapping)
  Authentication data entry: Select...
- ● Use custom login configuration
  Application login configuration: KerberosMapping

Apply

Select	Module	EJB	URI	Resource Reference	Target Resour Name
✓	KrbSampleEJB.jar	ApplicationFacade	KrbSampleEJB.jar,META-INF/ejb-jar.xml	jdbc/appDSref	jdbc/krbappD

Figure 13-13 Selecting authentication method

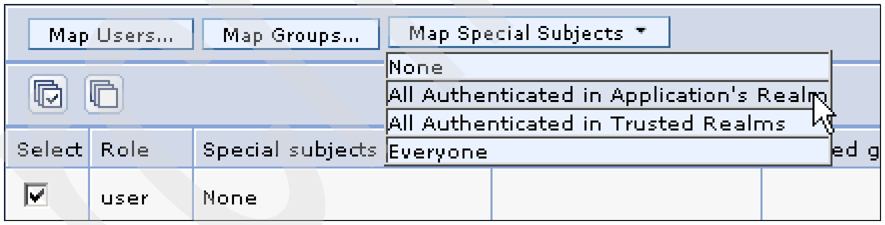
The “Login configuration section” of the reference is updated as shown in Figure 13-14.



Resource Reference	Target Resource JNDI Name	Login configuration
jdbc/appDSref	jdbc/krbappDS	Resource authorization: Container Authentication method: KerberosMapping

Figure 13-14 Updated Login configuration

6. Continue with the application installation until you reach the “Map security roles to users or groups” panel. Select the user role, and click **Map Special Subjects** → **All Authenticated in Application’s Realm**, as shown in Figure 13-15.



Select	Role	Special subjects
✓	user	None

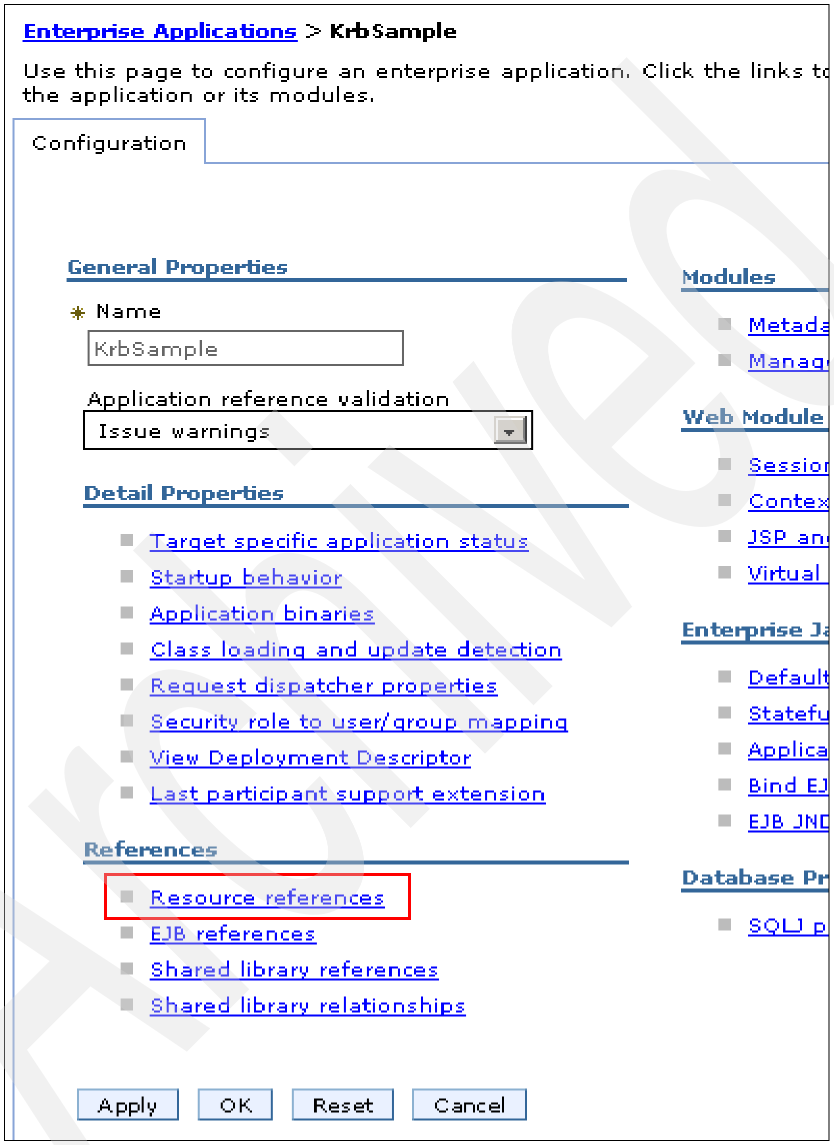
Map Users...
Map Groups...
Map Special Subjects
None
All Authenticated in Application's Realm
All Authenticated in Trusted Realms
Everyone

Figure 13-15 Mapping roles to users

7. Continue with the application installation panels until the Summary window. Click **Finish**.
8. Save the changes.

Next, you have to start the application. Click **Applications** → **Application Types** → **WebSphere enterprise applications**, select the application, and click **Start**.

You can always change the resource reference mapping later using the Resource references link on the Application properties page, as shown in Figure 13-16.



Enterprise Applications > KrbSample

Use this page to configure an enterprise application. Click the links to the application or its modules.

Configuration

General Properties

* Name

KrbSample

Application reference validation

Issue warnings

Detail Properties

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

References

- Resource references
- EJB references
- Shared library references
- Shared library relationships

Modules

- Metada
- Manag

Web Module

- Sessior
- Contex
- JSP an
- Virtual

Enterprise Ja

- Default
- Statefu
- Applica
- Bind EJ
- EJB JND

Database Pr

- SQLJ p

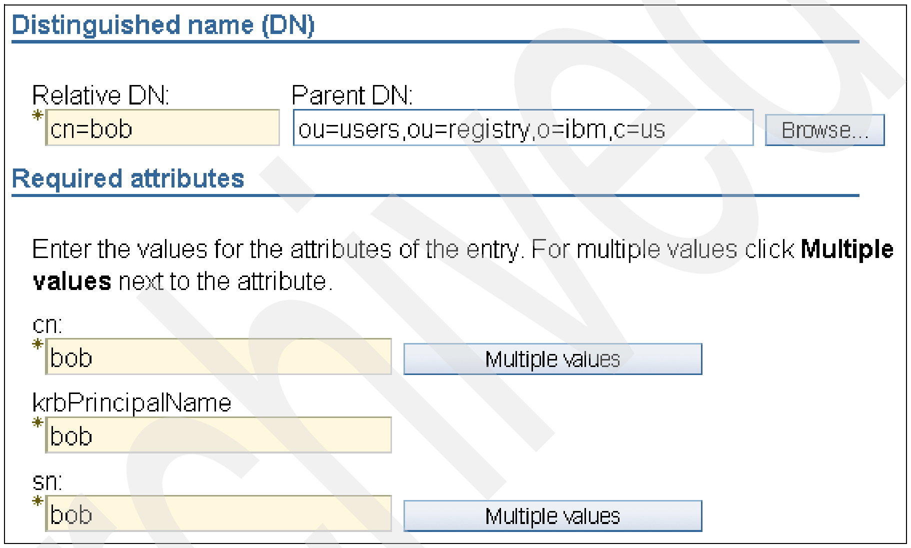
Apply OK Reset Cancel

Figure 13-16 Changing reference after installation

13.4.5 Configuring the user registry

WebSphere Application Server uses Kerberos to authenticate the user. However, for authorization it still queries the configured user registry. For this query to be successful, there must be a mapping between the Kerberos principal name and some searchable user attribute in the WebSphere Application Server user registry.

In this scenario, we used IBM Tivoli Directory Server to hold business users. The user entry has an additional `krbPrincipalName` attribute that stores trimmed Kerberos principal name, as shown in Figure 13-17.



Relative DN: * cn=bob
Parent DN: ou=users,ou=registry,o=ibm,c=us

Enter the values for the attributes of the entry. For multiple values click **Multiple values** next to the attribute.

cn: * bob
krbPrincipalName * bob
sn: * bob

Figure 13-17 User entry configuration

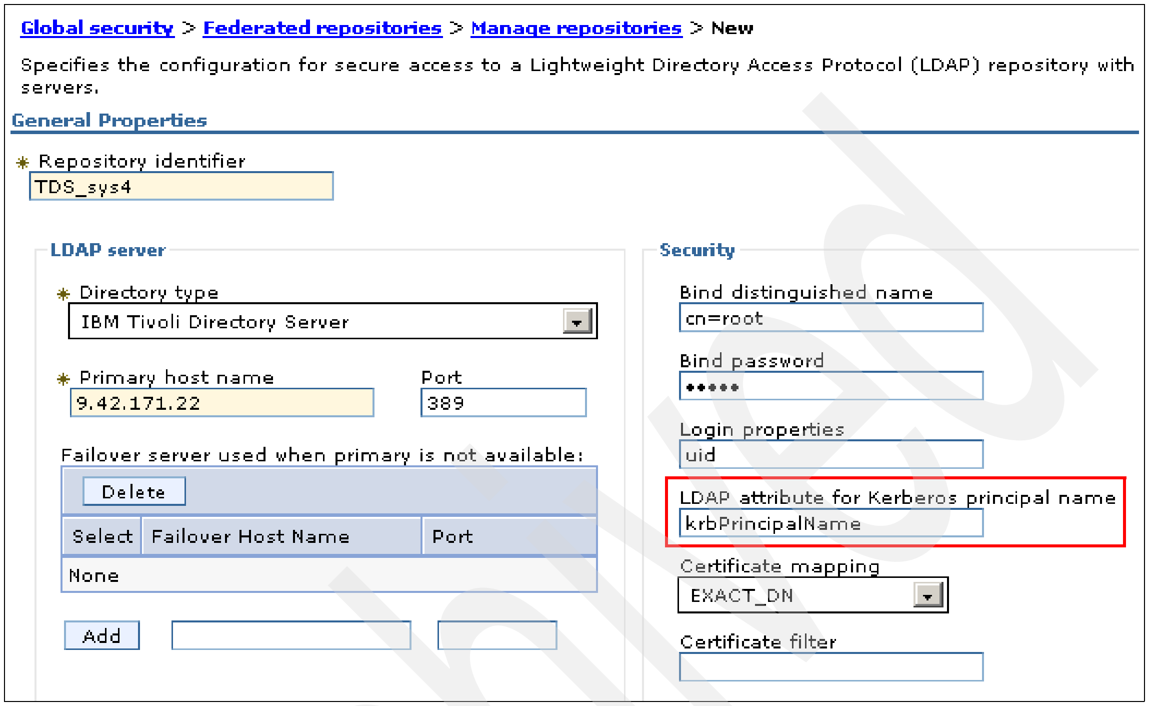
Configuring the federated repositories

The IBM Tivoli Directory Server will be added as LDAP registry, with business users to the Federated repository configuration.

Perform the following steps to LDAP registry to Federated repository:

1. Open the WebSphere administrative console. Then, in the navigation tree, click **Security** → **Global Security**.
2. In the User Account Repository section, click **Configure**.
3. Click the **Manage repositories** link, and then click **Add**.

4. Define the repository configuration, as shown in Figure 13-18. Specify the correct LDAP attribute for Kerberos principal name.


Global security > Federated repositories > Manage repositories > New

Specifies the configuration for secure access to a Lightweight Directory Access Protocol (LDAP) repository with servers.

General Properties

* Repository identifier: TDS_sys4

LDAP server

* Directory type: IBM Tivoli Directory Server

* Primary host name: 9.42.171.22

Port: 389

Failover server used when primary is not available:

Select	Failover Host Name	Port
None		

Security

Bind distinguished name: cn=root

Bind password: •••••

Login properties: uid

LDAP attribute for Kerberos principal name: krbPrincipalName

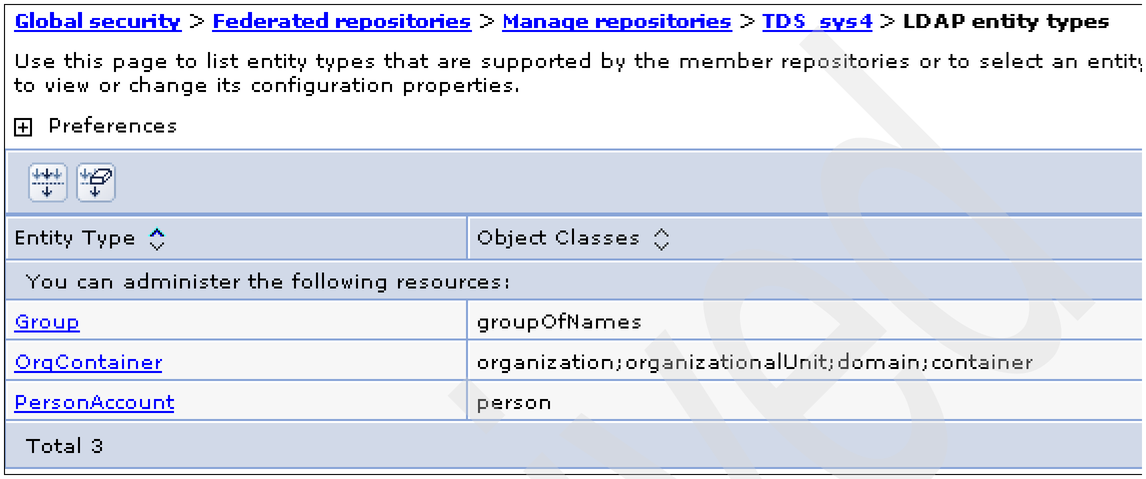
Certificate mapping: EXACT_DN

Certificate filter:

Figure 13-18 Defining the LDAP repository

5. Click **Apply**, and save the changes.

6. Click the **LDAP entity types** link, and verify that the entity Object classes are correct for your LDAP configuration (see Figure 13-19). Click the appropriate links to make corrections. Click **Federated repositories** in the bread crumb to go back to repository configuration.



Global security > Federated repositories > Manage repositories > TDS_sys4 > LDAP entity types

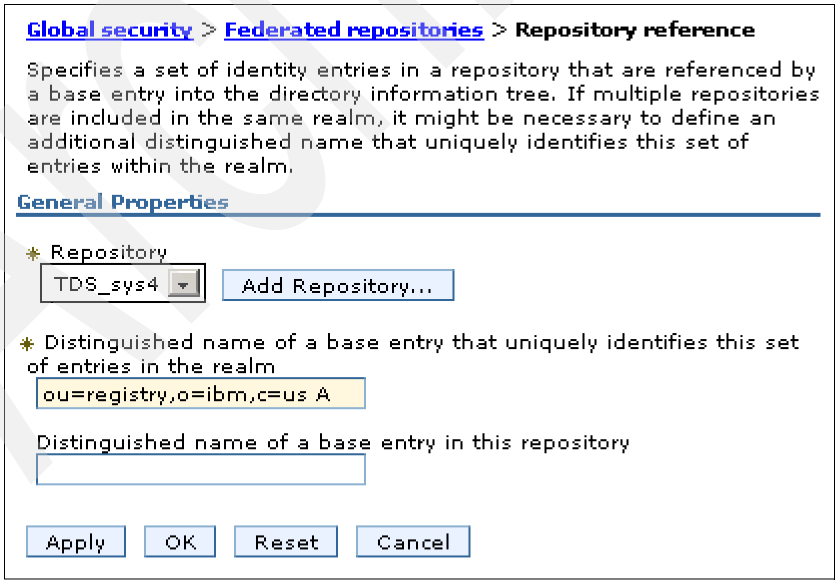
Use this page to list entity types that are supported by the member repositories or to select an entity to view or change its configuration properties.

Preferences

Entity Type	Object Classes
You can administer the following resources:	
Group	groupOfNames
OrgContainer	organization;organizationalUnit;domain;container
PersonAccount	person
Total 3	

Figure 13-19 Checking the object classes

7. Click **Add Base entry to Realm**, and define repository reference as shown in Figure 13-20.



Global security > Federated repositories > Repository reference

Specifies a set of identity entries in a repository that are referenced by a base entry into the directory information tree. If multiple repositories are included in the same realm, it might be necessary to define an additional distinguished name that uniquely identifies this set of entries within the realm.

General Properties

* Repository
TDS_sys4 [Add Repository...]

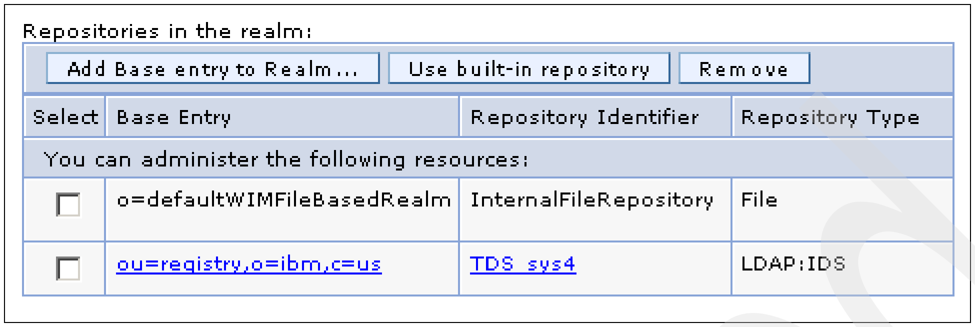
* Distinguished name of a base entry that uniquely identifies this set of entries in the realm
ou=registry,o=ibm,c=us A

Distinguished name of a base entry in this repository

[Apply] [OK] [Reset] [Cancel]

Figure 13-20 Adding repository to the realm

8. Click **OK**, and save the changes. The repository window looks similar to that shown in Figure 13-21.


Repositories in the realm:

Select	Base Entry	Repository Identifier	Repository Type
You can administer the following resources:			
☐	o=defaultWIMFileBasedRealm	InternalFileRepository	File
☐	ou=registry,o=ibm,c=us	TDS_sys4	LDAP:IDS

Figure 13-21 Repositories list in the realm

9. Restart the server, node agent, and deployment manager.

Now the WebSphere Application Server security runtime can find the dbclient user in the LDAP using its Kerberos principal name as the search attribute.

13.4.6 Testing authentication using the sample application Web client

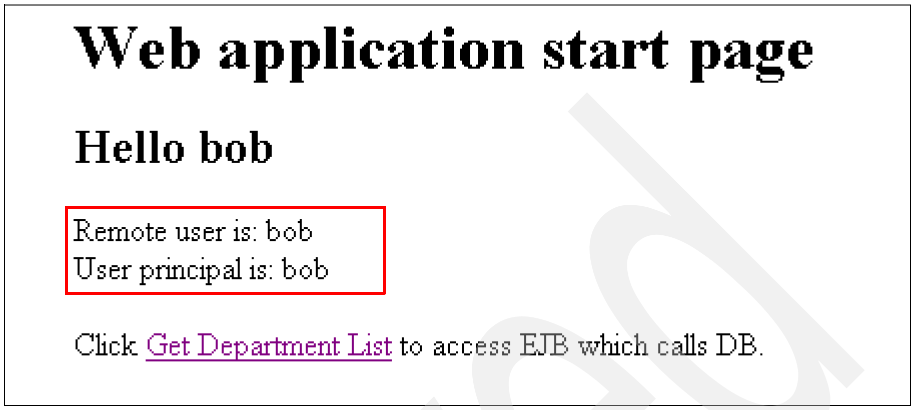
The sample application used to test this configuration contains a Web application that allowed us to show the client Kerberos delegation credential is used to authenticate to the database. Perform the following steps to run the application:

1. Open a new browser and type the following URL:

```
http://yourHostName:9080/KrbSampleWeb/index.jsp
```

2. In the authentication window, provide the Kerberos principal name *bob*, whose credentials are passed to DB2, and the Kerberos password. (You configured the user *bob* in 13.2.1, “Creating principals” on page 309.)

3. The first application page deploys, showing the user credentials, as shown in Figure 13-22.



Web application start page

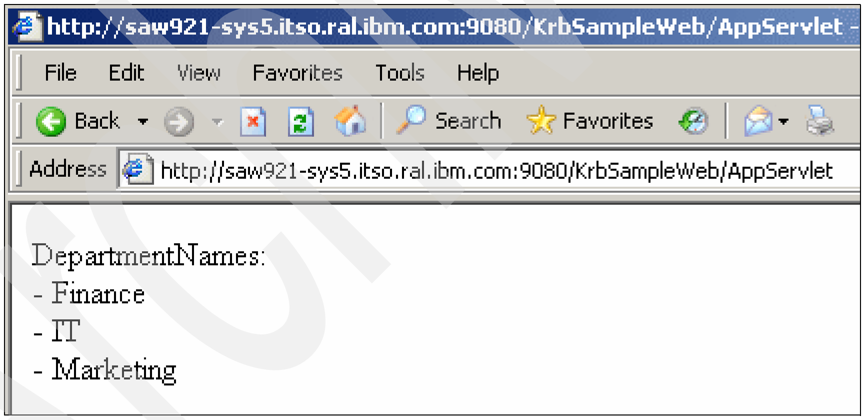
Hello bob

Remote user is: bob
User principal is: bob

Click Get Department List to access EJB which calls DB.

Figure 13-22 Accessing sample Web application

4. Click the **Get Department List** link to call the EJB, which calls DB2 using the reference defined during the installation. Figure 13-23 shows the result.


http://saw921-sys5.itso.ral.ibm.com:9080/KrbSampleWeb/AppServlet -
File Edit View Favorites Tools Help
Back Search Favorites
Address http://saw921-sys5.itso.ral.ibm.com:9080/KrbSampleWeb/AppServlet

DepartmentNames:
- Finance
- IT
- Marketing

Figure 13-23 Department list page

5. To confirm that the client Kerberos credential is used to authenticate to DB2, execute the following command as db2inst1 on the database server.

   ```
   $db2 list applications
   ```

   Example 13-11 shows the output. You can see that the user who is connected to the database is identified as `BOB`.

Example 13-11 Listing connected applications

```
$ db2 list applications

Auth Id  Application    Appl.      Application Id
          Name           Handle

-------- -------------- ---------- ----------------------------
BOB      db2jcc_applica 39112      9.42.170.219.45502.090814104203
```

In this example, we showed that client Kerberos credential is propagated from the Web login to the database. The next section shows how to configure the application client.

13.5 Configuring the JEE application client

In this section, we explain how to configure the JEE application client to use Kerberos authentication to WebSphere Application Server. We use two scenarios to show how the user can authenticate to server:

- Using a Kerberos principal name and password
- Using cached Kerberos credentials (krb5Ccache)

This scenario assumes that WebSphere Application Client is already installed.

For more information about configuring Kerberos authentication, the information center:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tsec_kerb_auth_client.html

13.5.1 Preparing common files

Both scenarios need the Kerberos configuration file (`krb5.conf`). Instead of creating a new configuration file, you can use the file created for the WebSphere Application Server configuration (see “Creating the Kerberos configuration file” on page 317). Copy the file to the `$APPCLIENT_ROOT/krb5` directory, for example:

```
/opt/IBM/WebSphere/AppClient/krb5
```

Copy the `KrbSample.ear` file to the `$APPCLIENT_ROOT/bin` directory:

```
/opt/IBM/WebSphere/AppClient/bin
```

13.5.2 Authenticating using a Kerberos principal name and password

To authenticate using a Kerberos principal name and password, follow these steps:

1. Go to the `/opt/IBM/WebSphere/AppClient/properties` folder.
2. Open the `sas.client.props` file and edit the following properties:

   ```
   com.ibm.CORBA.authenticationTarget=KRB5
   com.ibm.CORBA.loginSource=stdin
   com.ibm.CORBA.krb5ConfigFile=/opt/IBM/WebSphere/AppClient/krb5/kr
   b5.conf
   ```

3. Go to the `/opt/IBM/WebSphere/AppClient/bin` folder, and execute the following command (all in one line) to start the client:

   ```
   [root@saw921-sys4 bin]#./launchClient.sh KrbSample.ear
   -CCBootstrapHost=saw921-sys5.itso.ral.ibm.com
   -CCBootstrapPort=2809
   ```

   In this command, the parameters are as follows:

`-CCBootstrapHost`	The hostname of the application server that runs the JEE application
`-CCBootstrapPort`	The bootstrap port of the application server that runs the JEE application

4. At the login prompt provide the user name `bob` (the Kerberos principal) and the Kerberos password as shown in Example 13-12.

Example 13-12 Logging in using Java Application Client

```
WSCL0035I: Initialization of the Java EE Application Client
Environment has completed.
WSCL0014I: Invoking the Application Client class
com.ibm.itso.kerberos.app.KrbClient
```

```
Strating client
Realm/Cell Name: <default>
Username: bob
Password:
```

5. After the user has been authenticated, the application user interface displays, as shown in Example 13-13.

Example 13-13 Java Application Client user interface

```
Options:
1 - Get Department Names
2 - Exit
:
```

6. Type 1 and press the Enter key to invoke the EJB method. Example 13-14 shows the output of the method.

Example 13-14 Results

```
Printing Department List:
Finance
IT
Marketing
Options:
1 - Get Department Names
2 - Exit
:
```

7. To confirm that the client Kerberos credential is used to authenticate to DB2, execute the following command as db2inst1 on the database server.

```
$db2 list applications
```

Example 13-15 shows that the user who connected to the database is identified as BOB.

Example 13-15 Listing connected applications

```
$ db2 list applications

Auth Id  Application    Appl.      Application Id
          Name           Handle
-------- -------------- ---------- ----------------------------
BOB      db2jcc_applica 39214 9.42.170.219.45502.090814123223
```

8. Type 2 and press the Enter key to exit the application.

This example showed that the user Kerberos credential was used to authenticate to WebSphere Application Server and to the DB2 server.

13.5.3 Authenticating using Kerberos credentials cache

To authenticate using Kerberos credential cache, follow these steps:

1. Go to the `/opt/IBM/WebSphere/AppClient/properties` folder.
2. Open the `sas.client.props` file and edit the following properties:

   ```
   com.ibm.CORBA.authenticationTarget=KRB5
   com.ibm.CORBA.loginSource=krb5Ccache
   com.ibm.CORBA.krb5ConfigFile=/opt/IBM/WebSphere/AppClient/krb5/kr
   b5.conf
   ```

   If credential cache is not in the default location, specify it using the `com.ibm.CORBA.krb5CcacheFile` property.
3. Open `wsjaas_client.conf`, and edit the following entry with the values highlighted in bold font:

   ```
   WSKRB5Login{

   com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapperClient
   required credsType=INITIATOR useFirstPass=false forwardable=false
   renewable=false noAddress=false;
   };
   ```
4. Create the Kerberos credential cache by executing the following command (all in one line):

   ```
   ../java/jre/bin/java
   -Djava.security.krb5.conf=/opt/IBM/WebSphere/AppClient/krb5/krb5.
   conf com.ibm.security.krb5.internal.tools.Kinit bob
   ```

   Example 13-16 shows the output.

   Example 13-16 Creating credential cache

   ```
   [root@saw921-sys4 bin]# ../java/jre/bin/java
   -Djava.security.krb5.conf=/opt/IBM/WebSphere/AppClient/krb5/krb5.con
   f com.ibm.security.krb5.internal.tools.Kinit bob
   Password for bob@ITSO.RAL.IBM.COM:

   Done!
   New ticket is stored in cache file /root/krb5cc_root
   ```

5. Go to the /opt/IBM/WebSphere/AppClient/bin folder and execute the following command (all in one line):

   ```
   [root@saw921-sys4 bin]#./launchClient.sh KrbSample.ear
   -CCBootstrapHost=saw921-sys5.itso.ral.ibm.com
   -CCBootstrapPort=2809
   ```

6. You should get the application user interface *without* providing a user name and password as shown in Example 13-17.

Example 13-17 Java Application Client user interface

```
Strating client
[7/17/09 19:39:40:997 EDT] 00000000  W
UOW=1-36883688-25994897:saw921-sys4.itso.ral.ibm.com
source=com.ibm.ws.wssecurity.util.KRBSPNList org=IBM prod=WebSphere
component=Application Server thread=[P=978614:O=0:CT]
          No Keytab information was processed.
Options:
1 - Get Department Names
2 - Exit
:
```

7. Type 1 and press Enter to invoke the EJB method. Example 13-18 shows the output of the method.

Example 13-18 Results

```
Printing Department List:
Finance
IT
Marketing
Options:
1 - Get Department Names
2 - Exit
:
```

8. To confirm that the user Kerberos credential was used to authenticate to DB2, execute the following command as db2inst1 on the database server:

   ```
   $db2 list applications
   ```

   Example 13-19 shows that user who connected to the database is identified as `BOB`.

Example 13-19 Listing connected applications

```
$ db2 list applications

Auth Id  Application    Appl.      Application Id
          Name           Handle
-------- -------------- ---------- ----------------------------
BOB      db2jcc_applica 39214 9.42.170.219.45502.090814123223
```

14

Single sign-on from a Java thin client with AIX and z/OS Kerberos trusted realms

This chapter discusses a scenario where two Java thin application clients authenticate to WebSphere Application Server with AIX and z/OS Kerberos trusted realms. Because the Kerberos client principal names do not exist in RACF, the scenario configures a custom Java Authentication and Authorization Service (JAAS) mapping login module to map Kerberos client principal names to an RACF ID.

The first client uses two options for authentication:

- Using the Kerberos credential cache to authenticate to the AIX KDC and create the SPNEGO token
- Using the Kerberos principal name and password to authenticate to the AIX KDC and create the Kerberos token.

The second client uses these two options and additionally shows other options to fall back to using the user name and password if the credential cache fails.

This chapter includes the following topics:

- Scenario overview
- Configuring WebSphere Application Server
- Configuring client1
- Installing and configuring client2
- Validating the scenario

14.1 Scenario overview

The primary goal of this scenario allows a Java thin application client to participate in a single sign-on (SSO) using either the Kerberos or the SPNEGO token to authenticate to the WebSphere Application Server. The Java thin clients are configured to use an AIX Kerberos realm. WebSphere Application Server will use the z/OS Kerberos realm which is configured to trust the AIX Kerberos realm. After the WebSphere Application Server validates the Java thin client requests, the server maps the client Kerberos principal name to the RACF ID using the JAAS custom mapping login module.

The secondary goals of this scenario are to show the sample code for creating the SPNEGO and Kerberos token and the sample code for a JAAS custom mapping login module. The JAAS custom mapping login module maps a Kerberos principal name to a user in the WebSphere user registry. The login module is described in detail in 14.2.4, “Configuring a JAAS custom mapping login module” on page 367.

Note: This scenario features the following components:

- SSO from a Java thin application client to WebSphere Application Server on z/OS
- Two key distribution centers (KDCs), a KDC on AIX and a KDC on z/OS, with trust established between them
- Login with user ID and password or with a credential cache on the client
- WebSphere Application Client configuration for client 1
  - Login using credential cache and a SPNEGO token
  - Sample application: SpnegoClientSample and `rspnego.sh` script
- WebSphere Application Client configuration for client 2
  - Login with user ID and password
  - Login using credential cache
  - Login using a Kerberos token
  - Sample application: basicCalculator script
- WebSphere Application Server configuration:
  - User registry: Microsoft Active Directory
  - CSIv2 trusted realms
  - Authentication mechanism: Kerberos and LTPA
  - SPNEGO
  - Kerberos authentication and configuration file
  - Custom JAAS mapping login module to map Kerberos principal names to the WebSphere user registry (RACF)
  - Sample applications: Calculator application in the TechnologySamples, DefaultApplication (snoop servlet)

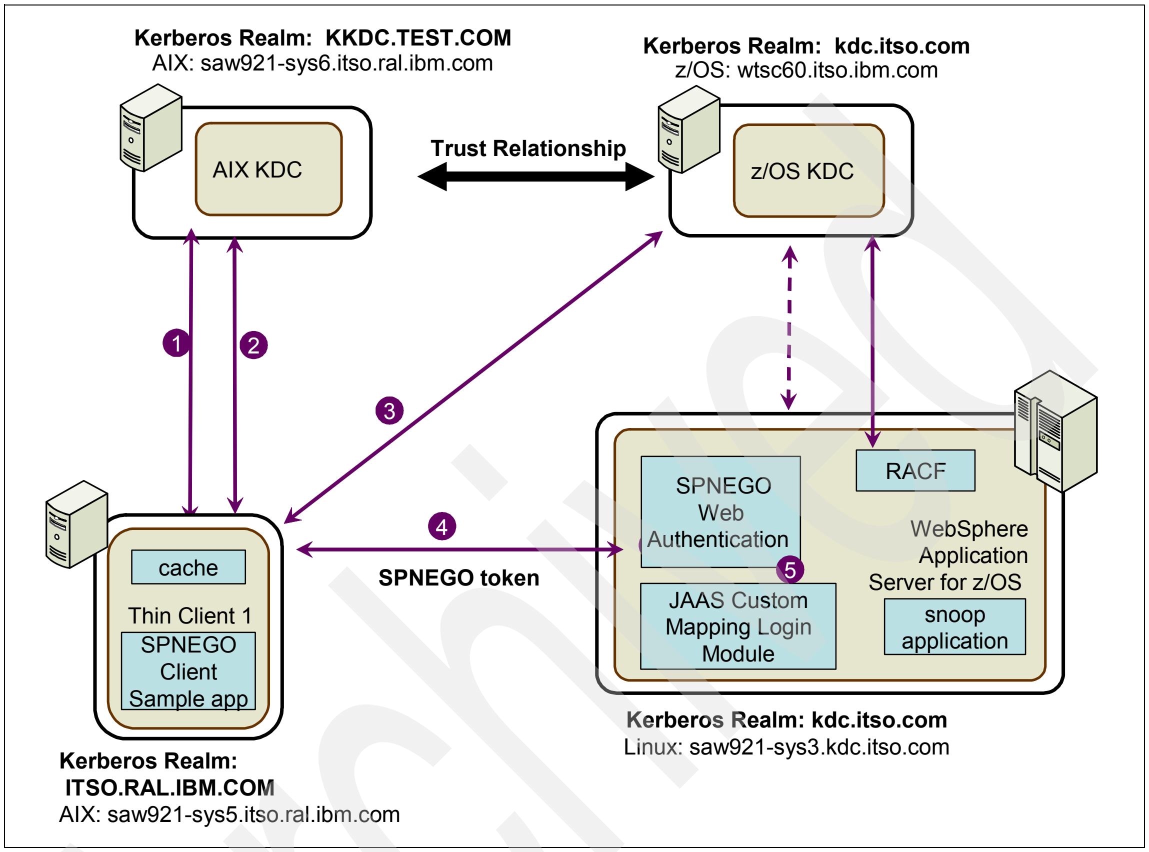
14.1.1 Step-by-step flow of the scenario

This scenario illustrates multiple options for a thin client to access a WebSphere application.

Thin application client1

In this scenario, client1 user logs in to the AIX KDC with the Kerberos principal name `krb_utle` and password using the Java **`kinit`** command. The Kerberos credential cache received from the AIX KDC is saved on the client at the default location of `/home/krb_utle/krb5cc_krb_utle`. The Kerberos credential cache has a default lifetime of 8 hours. The lifetime is configurable by the AIX KDC.

The user accesses the snoop servlet (part of the DefaultApplication application) using the sample SPNEGO Java client. The sample client creates the SPNEGO token using the Kerberos credential cache without having to re-enter the Kerberos principal name and password, as long as the Kerberos credential cache is still valid. The client then inserts the SPNEGO token into the HTTP header and sends the HTTP request to WebSphere Application Server for z/OS.



Kerberos Realm: KKDC.TEST.COM
AIX: saw921-sys6.itso.ral.ibm.com
AIX KDC
Trust Relationship
Kerberos Realm: kdc.itso.com
z/OS: wtsc60.itso.ibm.com
z/OS KDC
1
2
3
4
5
cache
Thin Client 1
SPNEGO Client Sample app
SPNEGO token
SPNEGO Web Authentication
RACF
WebSphere Application Server for z/OS
JAAS Custom Mapping Login Module
snoop application
Kerberos Realm: kdc.itso.com
Linux: saw921-sys3.kdc.itso.com
Kerberos Realm: ITSO.RAL.IBM.COM
AIX: saw921-sys5.itso.ral.ibm.com

Figure 14-1 Step-by-step flow for thin application client1

When client1 attempts to access the snoop application on WebSphere Application Server using the SPNEGO Client Sample application, the following steps occur:

1. The thin application client obtains the Kerberos ticket-granting ticket (TGT) from the AIX KDC using the Java **`kinit`** command. The cached Kerberos credential is saved to the default location of `/home/krb_utle/krb5cc_krb_utle`.

Note: This step usually needs to be done once a day because the default Kerberos ticket lifetime is 8 hours.

2. In the Java SPNEGO sample program running in the thin application client, client1, the following steps take place:
   a. The client obtains the Kerberos TGT from the AIX KDC using the Kerberos credential cache.
   b. The Kerberos client code requests a cross realm Kerberos ticket (TGS_REQ) for the z/OS Kerberos realm from the AIX KDC using the Kerberos TGT obtained in the previous step.
3. The Kerberos client uses a cross realm ticket to request a Kerberos service ticket for WebSphere Application Server for z/OS (TGS_REQ) from the z/OS KDC. The z/OS KDC returns the service ticket and eventually the thin client receives the SPNEGO token.
4. The sample client code inserts the SPNEGO token into the HTTP header and sends the HTTP request to WebSphere Application Server for z/OS for authentication.
5. WebSphere Application Server for z/OS has SPNEGO Web authentication configured. The SPNEGO filter is defined to force the snoop servlet to use SPNEGO Web authentication.

   When the server receives a Web request from the thin application client1, the SPNEGO Web authentication component processes the SPNEGO token. After the server validates the SPNEGO token, the server calls the JAAS custom mapping login module to map the Kerberos principal name to the RACF ID. Thus, the Kerberos principal name of `krb_utle` is mapped to the RACF ID of UTLE

Thin application client2

The thin application client2 is configured to use the AIX Kerberos realm. In this scenario, the thin application client uses the `basicCalculator` script that ships as part of the TechnologySamplesThinClient application. The thin client uses the following login sources to authenticate to WebSphere Application Server for z/OS:

- krb5Ccache
- prompt
- stdin
- properties

The `sas.client.props` file in the `/opt/IBM/WebSphere/AppClient/properties` directory is updated with the following property:

```
com.ibm.CORBA.authenticationTarget=KRB5
```

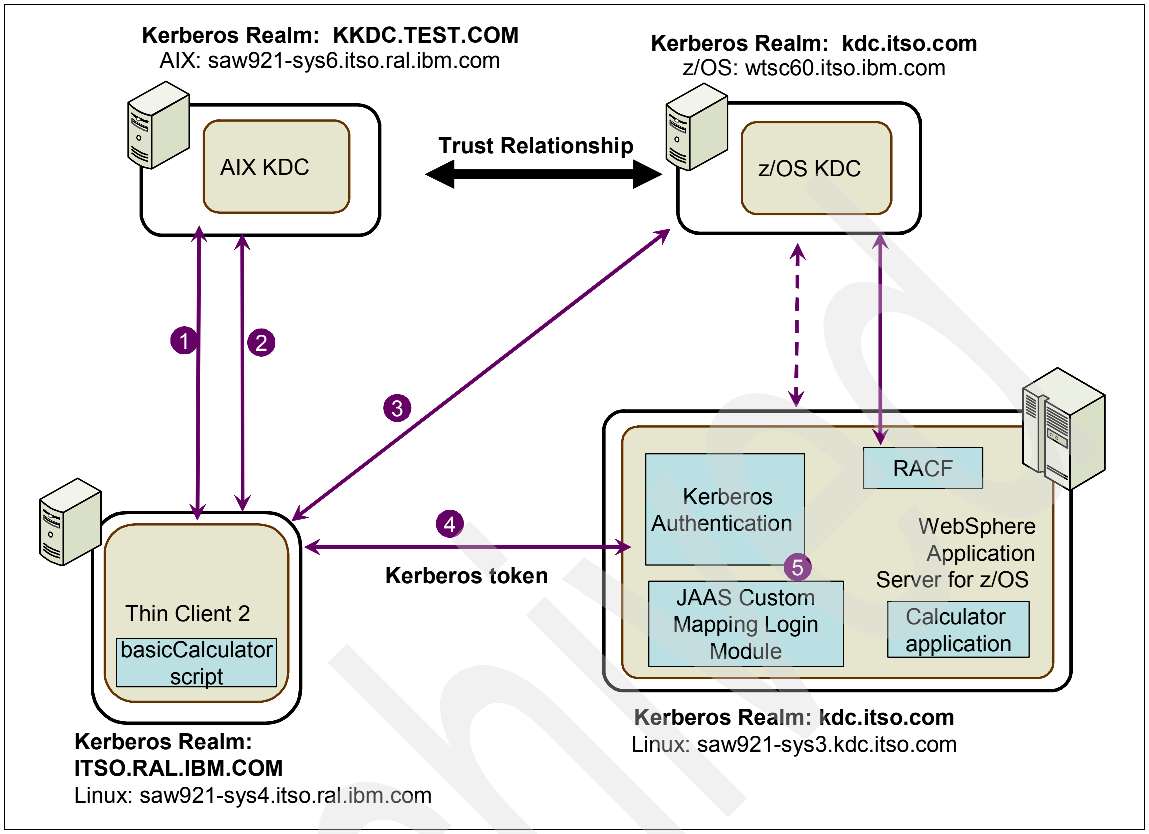

Kerberos Realm: KKDC.TEST.COM
AIX: saw921-sys6.itso.ral.ibm.com
Kerberos Realm: kdc.itso.com
z/OS: wtsc60.itso.ibm.com
AIX KDC
Trust Relationship
z/OS KDC
1
2
3
RACF
Kerberos Authentication
WebSphere Application Server for z/OS
4
Kerberos token
5
JAAS Custom Mapping Login Module
Calculator application
Thin Client 2
basicCalculator script
Kerberos Realm: kdc.itso.com
Linux: saw921-sys3.kdc.itso.com
Kerberos Realm: ITSO.RAL.IBM.COM
Linux: saw921-sys4.itso.ral.ibm.com

Figure 14-2 Step-by-step flow for thin application client2

When client2 attempts to access the calculator application on WebSphere Application Server, the following steps occur:

1. If the `loginSource` property in the `sas.client.props` file is set to `prompt`, then the client prompts for a Kerberos principal name and password.

   The Kerberos client runtime code uses the Kerberos principal name and password to obtain the Kerberos TGT from the AIX KDC.

2. The Kerberos client runtime code then requests a cross realm Kerberos ticket (TGS_REQ) for the z/OS Kerberos realm from the AIX KDC using the TGT.

3. The Kerberos client code uses a cross-realm ticket to request a Kerberos service ticket for the WebSphere Application Server for z/OS (TGS_REQ) from the z/OS KDC. The z/OS KDC returns the Kerberos token.

4. The Kerberos token returned from the previous step is added to the CSIv2 message authentication token and sent to WebSphere Application Server for z/OS for authentication.

5. WebSphere Application Server for z/OS has Kerberos authentication configured. The TechnologySamples application is installed on the WebSphere Application Server, and the basicCalculator application is parted of the TechnologySamples application.

   When the server receives an RMI request from the thin application client to invoke the basicCalculator application, the Kerberos login module validates the client2 Kerberos token for authentication. After the server validates the client Kerberos token, the server calls the JAAS custom mapping login module to map the Kerberos principal name to the RACF ID. Thus, the Kerberos principal name of `krb_utle` is mapped to the RACF id of UTLE.

14.1.2 Scenario components

This scenario requires the following basic systems and configurations:

- z/OS Z1, which is a z/OS V1R9 system
  - WebSphere Application Server Network Deployment installation of WebSphere Application Server for z/OS V7.0.0.5 on the z/OS Z1 system
  - z/OS KDC, which is a z/OS V1R9 KDC on the z/OS Z1 system
- An AIX Kerberos KDC installed on an AIX system (AIX A1) that is configured to use an LDAP server
- The zOS Kerberos realm and AIX Kerberos realm configured to trust each other
- An application client for WebSphere Application Server and thin client V7.0.0.5 (Thin client1) installed on an AIX system, AIX A2
- An application client for WebSphere Application Server and thin client V7.0.0.5 (Thin client2) installed on a Linux system, Linux L2

14.1.3 Basic requirements for this scenario

This scenario assumes that the following systems and configurations are in place:

- A Network Deployment environment of WebSphere Application Server for z/OS V7.0.0.5
- The z/OS KDC V1R9 is configured and set up

  For more information, refer to Chapter 2, “Setting up a KDC on a z/OS system” on page 13.

- The AIX KDC is configured and set up

  For more information, refer to Chapter 3, “Configuring IBM Network Authentication Service KDC on AIX” on page 29.
- The trust is set up between the AIX Kerberos realm and the z/OS Kerberos realm

  For more information, refer to Chapter 5, “Setting up trust between an AIX KDC and a z/OS KDC” on page 89.
- The user krb_utle exists in the local operating system registries of the two thin client systems, AIX A1 and Linux L2
- The Kerberos principal name krb_utle exists in the AIX Kerberos realm
- The user UTLE exists in the RACF registry and is authorized to access the snoop servlet
- A timing service is used on the thin client, AIX KDC, and on the z/OS KDC systems to ensure that the times are synchronized

14.1.4 Summary of the implementation steps for this scenario

To implement single sign-on (SSO) for a Java thin client to a WebSphere Application Server for z/OS environment requires the following high-level steps:

1. Set up a trust between the AIX and z/OS Kerberos realms.
2. Configure RACF as the user registry for WebSphere Application Server for z/OS.
3. Enable the SPNEGO Web and Kerberos authentication mechanism for WebSphere Application Server for z/OS.
4. Create a JAAS custom mapping login module to map the Kerberos principal name to an RACF ID.
5. Configure the JAAS custom mapping login module on WebSphere Application Server for z/OS.
6. Install the TechnologySamples application on the WebSphere Application Server for z/OS server. The calculator application is included in the sample application.
7. Install the Application Client for WebSphere Application Server, including the thin client software, on client1.
8. Configure the Java thin application client (client1) to use the Kerberos credential cache, user ID, and password.
9. Install the Application Client for WebSphere Application Server, including the thin client software, client2.

10. Configure a Java thin application client (client2) to use the Kerberos credential cache and principal name and password.

Note: All the Kerberos clients use the encryption type of des3-cbc-sha1. The thin client1 must use an encryption type that is supported by the Kerberos configuration of the WebSphere Application Server for z/OS, the AIX Kerberos realm, and the z/OS Kerberos realm.

14.2 Configuring WebSphere Application Server

This section describes how to configure the security settings on WebSphere Application Server for z/OS.

14.2.1 Configuring Kerberos authentication

The following sections describe how to configure Kerberos authentication on WebSphere Application Server for z/OS. This process has two steps:

1. First, you create the Kerberos configuration file, `krb5.conf`, which includes information about the z/OS Kerberos realm. This file is then updated to include the configuration information for the AIX Kerberos realm to which the two thin clients belong.
2. Next, you create the Kerberos keytab file, which contains pairs of Kerberos service principal names and encryption keys.

Creating the Kerberos configuration file (krb5.conf)

The `krb5.conf` file contains the Kerberos configuration information, including the location of the KDC, the Kerberos realm name, and the location of the Kerberos keytab file. You can create the `krb5.conf` file using the administrative scripting task `createKrbConfigFile` as follows:

1. Start the command line utility by running the **`wsadmin.sh -lang jython`** command from the *`app_server_root`*`/bin` directory.
2. At the **`wsadmin`** prompt, enter the following command:

```
AdminTask.createKrbConfigFile('[-krbPath
/wasmvconfig/mvcell/mvdmnode/myKerberos/krb5.conf -realm
ITSO.IBM.COM -kdcHost wtsc60.itso.ibm.com -dns itso.ibm.com
-keytabPath
/wasmvconfig/mvcell/mvdmnode/myKerberos/scenario4b.keytab
-encryption des3-cbc-sha1]')
```

This command uses the following parameters:

`-krbPath`	Specifies the fully qualified file system location of the Kerberos configuration file on the z/OS system.

> **Note:** You do not have to use the default location of `/etc/krb5` when specifying the `krb5.conf` file.

`-realm`	The Kerberos realm name of the z/OS Kerberos realm. You can specify only one realm here.
`-kdcHost`	The host name of the z/OS KDC system.
`-dns`	The domain name service of the z/OS KDC system.
`-keytab`	Provides the fully qualified file system location of the Kerberos keytab file on the z/OS system.
`-encryption`	Identifies the list of supported encryption types on the z/OS Kerberos realm.

3. After this command executes, a message similar to the following displays:

```
/wasmvconfig/mvcell/mvdmnode/myKerberos/krb5.conf has been
created.
```

> **Note:** The task sets the file permission bits of the `krb5.conf` file to `644` to allow everyone to read the file and to allow only the administrator to update it. Be careful not to change these permissions to allow more control.

Example 14-1 illustrates the contents of the generated `krb5.conf` file.

Example 14-1 The generated krb5.conf file

```
[libdefaults]
   default_realm = ITSO.IBM.COM
   default_keytab_name =
FILE:/wasmvconfig/mvcell/mvdmnode/myKerberos/scenario4b.keytab
   default_tkt_enctypes =des3-cbc-sha1
   default_tgs_enctypes = des3-cbc-sha1
   forwardable  = true
   renewable  = true
   noaddresses = true
   clockskew  = 300
[realms]
        ITSO.IBM.COM = {
                kdc = wtsc60.itso.ibm.com:88
                default_domain = itso.ibm.com
   }
[domain_realm]
   .itso.ibm.com = ITSO.IBM.COM
```

4. Because you can specify only one realm using the scripting task `createKrb5ConfigFile`, you must update the `krb5.conf` file to add the trusted AIX Kerberos realm information as follows:

   a. Add the following information to the `[realms]` section of the `krb5.conf` to define the AIX Kerberos realm:

   ```
   ITSO.RAL.IBM.COM = {
                   kdc = saw921-sys6.itso.ral.ibm.com:88
                   default_domain = itso.ral.ibm.com
      }
   ```

   This command uses the following parameters:

`ITSO.RAL.IBM.COM`	Specifies the Kerberos realm name of the AIX Kerberos realm
`kdc`	Specifies the hostname:port of the AIX KDC
`default_domain`	Specifies the default domain for the hosts in the AIX Kerberos realm

   b. Add the following information to the `[domain_realm]` section of the `krb5.conf` file to map the two application thin client systems to the AIX Kerberos realm. Because the host names of the two client systems are `aw921-sys4.itso.ral.ibm.com` and `saw921-sys5.itso.ral.ibm.com`, it is sufficient to add just `itso.ral.ibm.com` as follows:

   ```
   .itso.ral.ibm.com = ITSO.RAL.IBM.COM
   ```

Example 14-2 shows the updated `krb5.conf` file.

Example 14-2 The krb5.conf with cross-realm trust

```
[libdefaults]
   default_realm = ITSO.IBM.COM
   default_keytab_name =
FILE:/wasmvconfig/mvcell/mvdmnode/myKerberos/scena
rio4b.keytab
   default_tkt_enctypes = des3-cbc-sha1
   default_tgs_enctypes = des3-cbc-sha1
   forwardable  = true
   renewable  = true
   noaddresses = true
   clockskew  = 300
[realms]
        ITSO.IBM.COM = {
```

```
                kdc = wtsc60.itso.ibm.com:88
                default_domain = itso.ibm.com
   }
        ITSO.RAL.IBM.COM = {
                kdc = saw921-sys6.itso.ral.ibm.com:88
                default_domain = itso.ral.ibm.com
   }

[domain_realm]
   .itso.ral.ibm.com = ITSO.RAL.IBM.COM
   .itso.ibm.com = ITSO.IBM.COM
```

To verify that the `krb5.conf` file is valid, use the Java command **`kinit`** to authenticate a Kerberos principal to the z/OS KDC as follows:

```
app_server_root/java/bin/java
-Djava.security.krb5.conf=wasmvconfig/mvcell/mvdmnode/myKerberos/krb
5.conf com.ibm.security.krb5.internal.tools.Kinit krb_myPrincipal
```

In this command, `krb_myPrincipal` is the Kerberos principal name that exists in the z/OS KDC. Example 14-3 shows the expected output of this command. (You can ignore the IOException.)

Example 14-3 The output of the krb_myPrincipal command

```
java.io.IOException: Primary principals do not match
Done!
New ticket is stored in cache file
/var/WebSphere/home/MVCFG/krb5cc_krb_myPrincipal
```

Creating the Kerberos keytab file

The Kerberos keytab file contains pairs of Kerberos SPNs and encryption keys that were derived from the Kerberos password. The keytab file is used by the WebSphere Application Server for z/OS server to process incoming requests that have a Kerberos or SPNEGO token. These steps assume that the z/OS KDC is already set up. The following Kerberos SPNs should already exist in the z/OS Kerberos realm:

- `HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM`
- `WAS104/wtsc60.itso.ibm.com@ITSO.IBM.COM`

Refer to 2.3.1, “Creating the Kerberos service principal name” on page 19 for information about how to create these SPNs.

Note: A Kerberos keytab file includes a list of keys that are analogous to user passwords. It is important for hosts to protect their Kerberos keytab files by storing them on the local disk.

To create the Kerberos keytab file, follow these steps:

1. Generate the keytab file that contains the Kerberos SPNs for SPNEGO Web and Kerberos authentication using the **ktab** utility that is provided by Java as follows:

   a. Add the SPN to the keytab file for SPNEGO Web authentication using the following command:

   ```
   app_server_root/java/bin/java
   -Djava.security.krb5.conf=wasmvconfig/mvcell/mvdmnode/myKerberos/
   krb5.conf com.ibm.security.krb5.internal.tools.Ktab -a
   HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM
   ```

   Note: Because this scenario does not use the default location for the `krb5.conf` file (`/etc/krb5`), the file location must be specified using the Java argument:

   `-Djava.security.krb5.conf`

   b. Add the SPN to the keytab file for Kerberos authentication with the following command:

   ```
   app_server_root/java/bin/java
   -Djava.security.krb5.conf=/wasmvconfig/mvcell/mvdmnode/myKerberos
   /krb5.conf com.ibm.security.krb5.internal.tools.Ktab -a
   WAS104/wtsc60.itso.ibm.com@ITSO.IBM.COM
   ```

   Note: The first component for the Kerberos WebSphere Application Server SPN (`WAS104` in this scenario) can be any string. However, the initial component for SPNEGO must be `HTTP`.

2. Verify that the Kerberos keytab file was created correctly using the **klist** command to display the entries in the keytab file:

   ```
   app_server_root/java/bin/java
   -Djava.security.krb5.conf=/wasmvconfig/mvcell/mvdmnode/myKerberos
   /krb5.conf com.ibm.security.krb5.internal.tools.Klist -k -e
   /wasmvconfig/mvcell/mvdmnode/myKerberos/scenario4b.keytab
   ```

   In this command, **-k** specifies that the key tab entries are to be listed and **-e** shows the encryption type.

Example 14-4 illustrates the output of the **klist** command.

Example 14-4 Output of the klist command

```
Key table: scenario4b.keytab
Number of entries: 2

[1] principal: HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM
   KVNO: 1

   Encryption type: DES3 CBC mode with SHA1-KD

[2] principal: WAS104/wtsc60.itso.ibm.com@ITSO.IBM.COM
   KVNO: 1

   Encryption type: DES3 CBC mode with SHA1-KD
```

After completing these steps, you should have a Kerberos configuration file, `krb5.conf`, with information about both the z/OS Kerberos realm and the AIX Kerberos realm. You should have a Kerberos keytab file, `scenario4b.keytab`, with entries for the WebSphere Application Server SPN and the SPNEGO SPN.

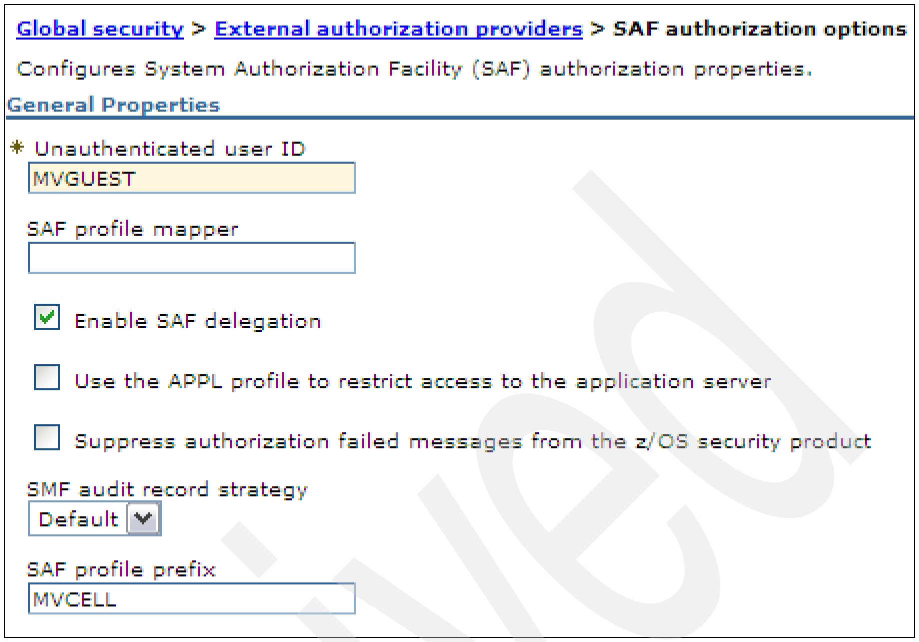
14.2.2 Configuring the WebSphere global security features

The following steps might have already been done in your environment. The WebSphere cell is configured to use SAF authorization, and the Local operating system is the active user registry.

To configure the WebSphere global security features, follow these steps:

1. Log on to the WebSphere Application Server administrative console and navigate to **Security** → **Global security**.
2. Select the “External authorization providers” option. Then, click **System Authorization Facility (SAF) authorization**.
3. (Optional) The z/OS KDC setup activates the APPL class, which implies that WebSphere Application Server will also use this use class for restricting access to the server, and requires additional setup in the SAF database.

   If you do not want WebSphere to use the APPL class, then you can disable the option on the administrative console. On the External authorization providers page, click **Configure**. Then, clear the “Use the APPL profile to restrict access to the application server” option. Figure 14-3 shows this option.


Global security > External authorization providers > SAF authorization options

Configures System Authorization Facility (SAF) authorization properties.

General Properties

* Unauthenticated user ID: MVGUEST

SAF profile mapper:

[✓] Enable SAF delegation

[ ] Use the APPL profile to restrict access to the application server

[ ] Suppress authorization failed messages from the z/OS security product

SMF audit record strategy: Default

SAF profile prefix: MVCELL

Figure 14-3 The SAF authorization options page

4. (Optional) If you chose to use the APPL profile to restrict access to the application server, then you must complete this step. Otherwise, skip this step.

   Issue the following RACF commands to give the SAF user id permission to the APPL profile.

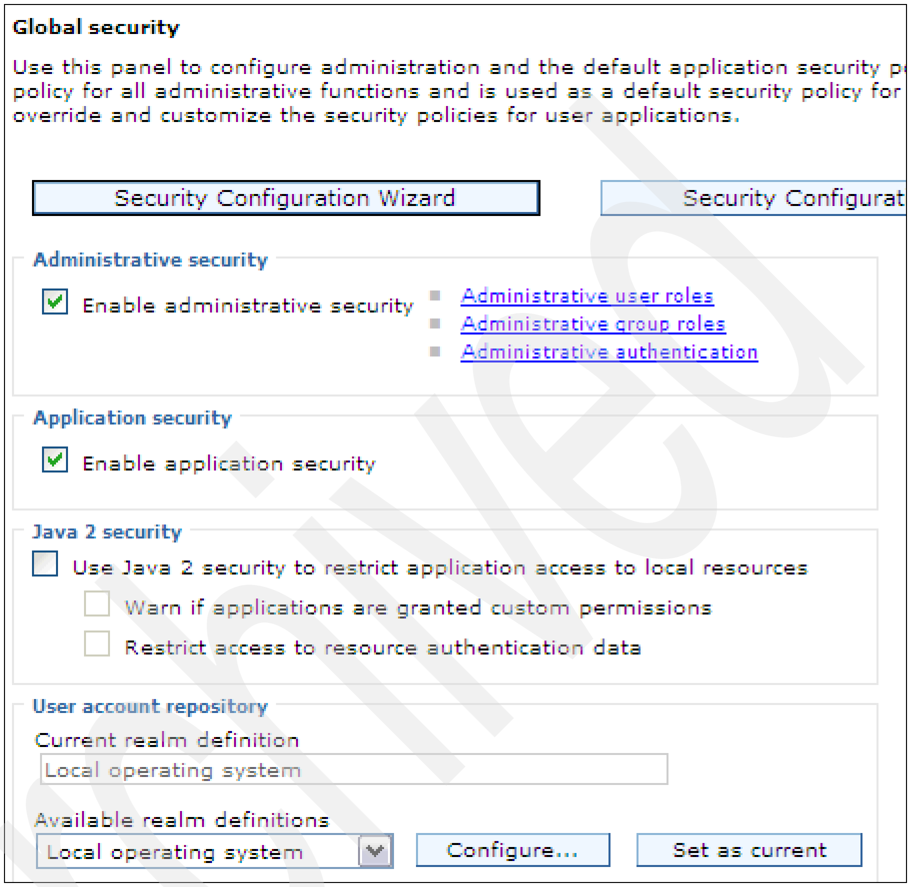
   ```
   PERMIT MVCELL ACCESS(READ) CLASS(APPL) ID(UTLE)
   ```

   In this command, `MVCELL` is the SAF profile prefix configured for the application server. If no SAF profile prefix is defined, then use `CBS390`.

   `UTLE` is the SAF user ID that will access the application.

5. Click **OK**, and then click **OK** again to return to the Global security page.
6. Select **Local operating system** in the “Available realm definitions” field, and click **Configure**. For Server user identity, select **Automatically generated server identity**. Then, click **OK** to return to the Global security page.
7. Select **Local operating system** in the Available realm definitions field, and click **Set as current**.

8. Verify that you have selected **Enabled** for both administrative and application security (as shown in Figure 14-4).



Global security

Use this panel to configure administration and the default application security p
policy for all administrative functions and is used as a default security policy for
override and customize the security policies for user applications.

Security Configuration Wizard
Security Configurat

Administrative security

[✓] Enable administrative security

Administrative user roles
Administrative group roles
Administrative authentication

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: Local operating system

Available realm definitions: Local operating system

Figure 14-4 The Global security page

9. Click **Apply** to apply all the settings in the Global security page.

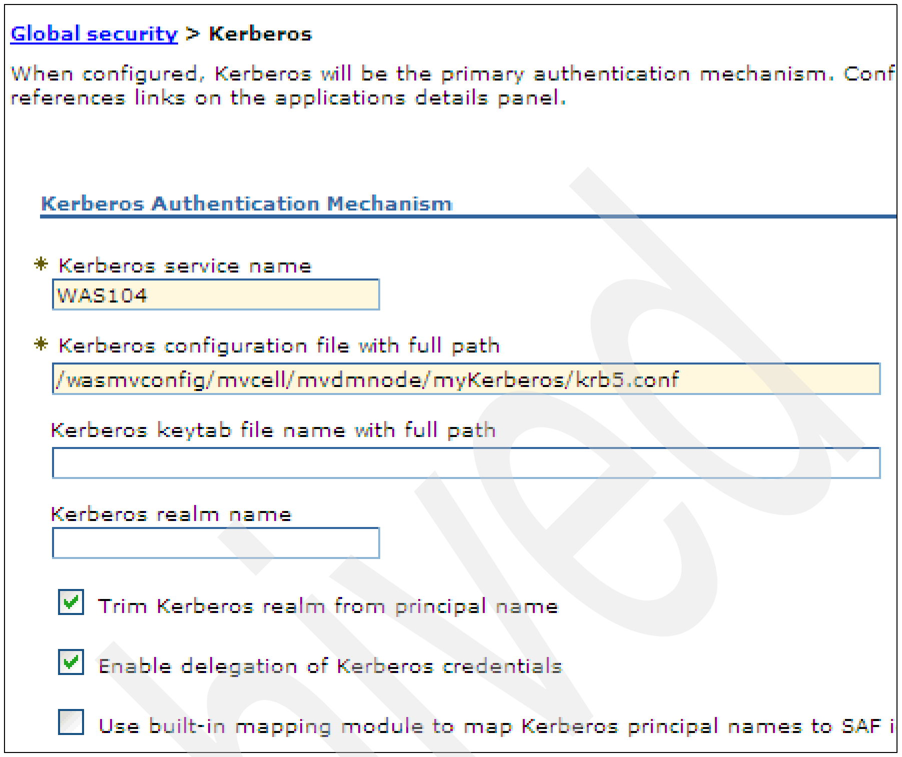
10. Save and synchronize configuration changes to the nodes.

You have now configured the global security features on the WebSphere Application Server for z/OS cell.

Configuring the Kerberos authentication mechanism

To configure Kerberos authentication in WebSphere Application Server for z/OS, follow these steps:

1. Log on to the WebSphere Application Server administrative console, and navigate to **Security** → **Global security**.
2. Select the **Kerberos configuration** link inside the Authentication box to configure Kerberos Authentication. Then, in the Kerberos Authentication Mechanism window, shown in Figure 14-5 on page 364, follow these steps:
   a. Enter `WAS104`, the Kerberos WebSphere Application Server SPN initial component name, as the Kerberos service name. This value must match the first component of the WebSphere Application Server SPN.
   b. Enter `/wasmvconfig/mvcell/mvdmnode/myKerberos/kr5.conf`, the location of the Kerberos configuration file, in the "Kerberos configuration file with full path" field.
   c. Select **Trim Kerberos realm from principal name** to remove the suffix of the principal user name.
   d. Select **Enable delegation of Kerberos credentials** to allow the client credentials to flow to other processes.


Global security > Kerberos

When configured, Kerberos will be the primary authentication mechanism. Conf
references links on the applications details panel.

Kerberos Authentication Mechanism

* Kerberos service name
WAS104

* Kerberos configuration file with full path
/wasmvconfig/mvcell/mvdmnode/myKerberos/krb5.conf

Kerberos keytab file name with full path

Kerberos realm name

[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

[ ] Use built-in mapping module to map Kerberos principal names to SAF i

Figure 14-5 Configuring Kerberos on the administrative console

3. Save and synchronize configuration changes to the nodes.

This completes the setup of Kerberos authentication.

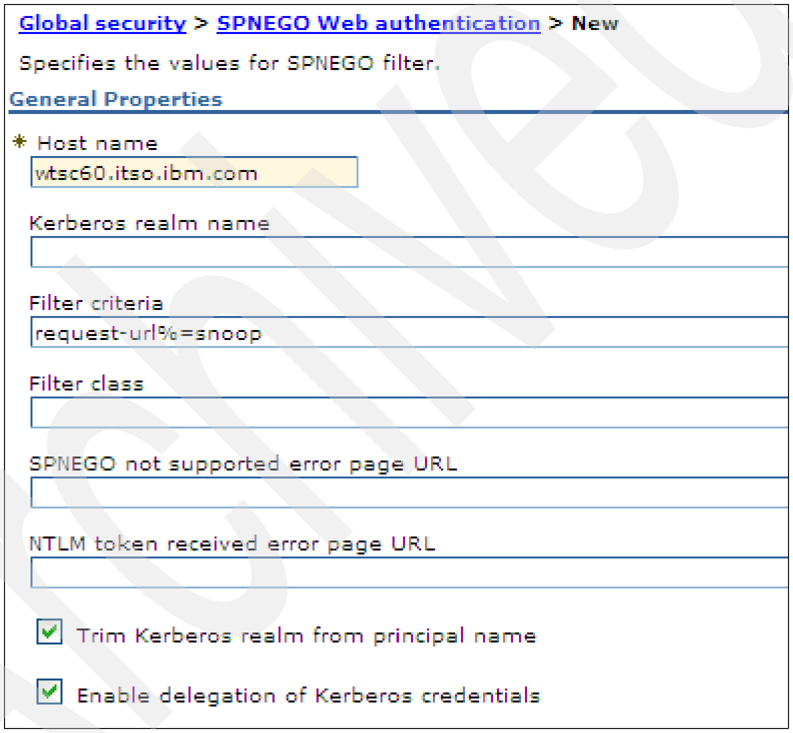
14.2.3 Configuring SPNEGO Web authentication

This section explains how to configure the SPNEGO Web authentication in WebSphere Application Server to allow clients to pass a SPNEGO token to authenticate with WebSphere Application Server.

To configure the DefaultApplication (sample application provided by WebSphere Application Server) as a SPNEGO service, follow these steps:

1. Log on to the WebSphere Application Server administrative console and navigate to **Security** → **Global security**.
2. Expand **Web and SIP security**, and then click **SPNEGO Web authentication**.

3. Each application that participates in SSO must have a filter defined. On SPNEGO Filters list, click **New**. Then, specify the following properties, as shown in Figure 14-6:
   a. Enter the fully qualified host name of WebSphere server in the Host name field.
   b. Enter `request-url%=snoop` in the Filter criteria field.
   c. Select **Trim Kerberos realm from principal name** to remove the suffix of the principal user name.
   d. Select **Enable delegation of Kerberos credentials** to allow the client credentials to flow to other processes.


Global security > SPNEGO Web authentication > New

Specifies the values for SPNEGO filter.

General Properties

* Host name: wtsc60.itso.ibm.com

Kerberos realm name:

Filter criteria: request-url%=snoop

Filter class:

SPNEGO not supported error page URL:

NTLM token received error page URL:

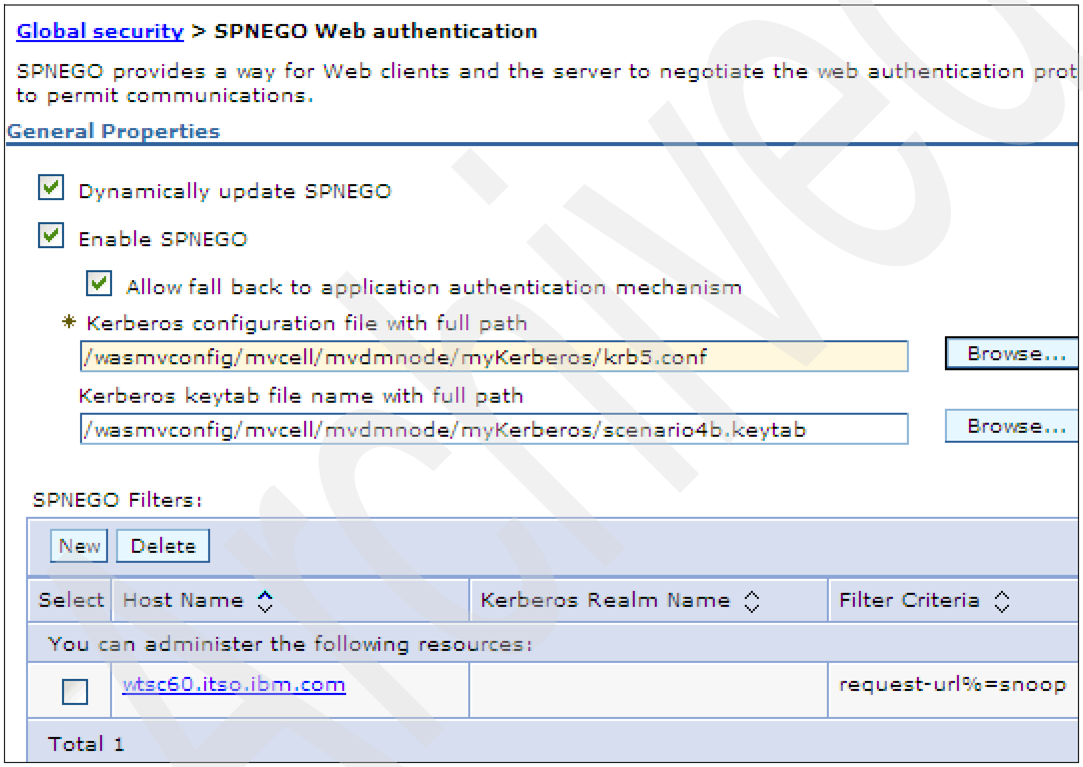
[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

Figure 14-6 SPNEGO Web authentication filter

4. Click **OK** to return you the SPNEGO Web authentication page.

5. Enable the "Dynamically update SPNEGO" option so that changes to the SPNEGO configuration are loaded automatically without having to restart the servers.
6. Click **Enable SPNEGO**.
7. Enable the "Allow fallback to application authentication mechanism" option to specify that SPNEGO is used to log in to WebSphere Application Server first. If SPNEGO authentication fails, the authentication mechanism that is defined during application assembly time is used.
8. Select the Kerberos configuration and keytab file if they are not already selected, as shown in Figure 14-7. Then, click **OK**.


Global security > SPNEGO Web authentication

SPNEGO provides a way for Web clients and the server to negotiate the web authentication prot
to permit communications.

General Properties

✓ Dynamically update SPNEGO

✓ Enable SPNEGO

✓ Allow fall back to application authentication mechanism

* Kerberos configuration file with full path
/wasmvconfig/mvcell/mvdmnode/myKerberos/krb5.conf

Kerberos keytab file name with full path
/wasmvconfig/mvcell/mvdmnode/myKerberos/scenario4b.keytab

SPNEGO Filters:

Select	Host Name	Kerberos Realm Name	Filter Criteria
You can administer the following resources:			
☐	wtsc60.itso.ibm.com		request-url%=snoop
Total 1			

Figure 14-7 Enabling SPNEGO Web authentication

9. Save and synchronize configuration changes to the nodes.

This completes the SPNEGO Web Authentication setup.

14.2.4 Configuring a JAAS custom mapping login module

The intent of configuring a JAAS custom login module is to map the Kerberos principal name to a user in the RACF registry. The WebSphere Information Center includes a sample mapping login module. You can use this sample as a starting point.

The final code for the login module is listed in Example A-1 on page 491. For this example, it is assumed that the Kerberos principal name has the format of `krb_userName`. The mapping code of the sample login module is updated to remove the `krb_` prefix so that the principal is mapped to the RACF user `userName`.

The JAAS custom login module is updated to retrieve the Kerberos principal name using the `getPrivateCredentials(KRBAuthnToken.class)` method instead of the `KerberosPrincipal` class.

Note: You can use the `KerberosPrincipal` class only if you are using SPNEGO Web authentication. However, you *must* use the `KRBAuthnToken` class if you are using both SPNEGO Web and Kerberos authentication. Furthermore, the `KRBAuthnToken` class is recommended because this token can regenerate the `GSSCredential`. Otherwise, the `GSSCredential` is lost during propagation because it is not serializable.

The custom login module maps the Kerberos principal name to the SAF user registry identity, and inserts the mapped identity in the hashtable property `com.ibm.wsspi.security.cred.userId (WSCREDENTIAL_USERID)`. The `wsMapDefaultInboundLoginModule` then uses the mapped identity to create a `WSCredential`.

Creating a sample JAAS custom mapping login module

These steps describe how to write, compile, and create the `.jar` file for the JAAS custom mapping login module. If you do not want to start with the sample file, you can skip steps 1-4 and just copy the final login module listed in Example A-1 on page 491.

1. Copy the sample mapping module provided in the following information center article:

   `http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/tsec_kerb_map.html`

   Then, save the file as `ITSOSpnegoMappingLoginModule.java`.

2. Edit this file and change all occurrences of `sampleSpnegoMappingLoginModule` to `ITSOSpnegoMappingLoginModule`.

3. Now, update the custom login module to retrieve the Kerberos principal using the `KRBAuthnToken` class. Add the following statement to the import section to include the `KRBAuthnToken` class:

   ```
   import com.ibm.wsspi.wssecurity.platform.token.KRBAuthnToken;
   ```

   Remove this line in the `login()` method:

   ```
   java.util.Set krb5Principals=
   subject.getPrincipals(KerberosPrincipal.class);
   ```

   After this line, add the following code:

   ```
   java.util.Set krb5Principals=
   subject.getPrivateCredentials(KRBAuthnToken.class);
   ```

4. Update the custom login module to use the `KRBAuthnToken` class and modify the mapping logic. Add the following lines at the beginning of the `login()` method:

   ```
   KRBAuthnToken krbAuthnToken = null;
   String kerbPrefix = "krb_";
   ```

   Remove the following section:

   ```
   Object princObj = krb5PrincIter.next();
   debugOut("Kerberos principal name: "+ princObj.toString());

   if (princObj != null &&
   princObj.toString().equals("utle@WSSEC.AUSTIN.IBM.COM")){
   mapUid = "user1";
   ```

   Then, after this line, add the code shown in Example 14-5.

   Example 14-5 New mapping logic for custom JAAS login module

   ```
   krbAuthnToken = (KRBAuthnToken)krb5PrincIter.next();
   String kerbPrincipal = (String) krbAuthnToken.getTokenPrincipal() +
   "@" + krbAuthnToken.getTokenRealm();
   debugOut("Kerberos principal name: "+ kerbPrincipal.toString());
   String principal = null;
   int index = kerbPrincipal.indexOf("@");
   if (index > -1)
   {
      principal =
      kerbPrincipal.substring(0,kerbPrincipal.indexOf("@"));
   }
   else
   {
      principal = kerbPrincipal;
   }
   ```

```
if (principal.startsWith(kerbPrefix))
{
   mapUid =
   principal.substring(kerbPrefix.length(),principal.length());
```

The entire contents of the updated JAAS login module are provided in Example A-1 on page 491.

5. Create the directory structure for the Java package. First, open a Telnet session to the z/OS system and cd to the *app_server_root*/lib/ext directory. Then create the directory com/ibm/websphere/security using the following commands:

   ```
   mkdir com
   mkdir com/ibm/
   mkdir com/ibm/websphere
   mkdir com/ibm/websphere/security
   ```

6. FTP, in ASCII, the ITSOSpnegoMappingModule.java file to the z/OS system to the newly created directory structure:

   ```
   app_server_root/lib/ext/com/ibm/websphere/security
   ```

7. Compile the Java file using the following command:

   ```
   app_server_root/java/bin/javac -classpath
   app_server_root/plugins/com.ibm.ws.runtime.jar:app_server_root/pl
   ugins/com.ibm.wsfp.main.jar
   com/ibm/websphere/security/ITSOSpnegoMappingLoginModule.java
   ```

   This command creates the ITSOSpnegoMappingLoginModule class file.

8. Create a .jar file that includes the newly created class file using the following command:

   ```
   app_server_root/java/bin/jar cvf ITSOSpnegoMappingLoginModule.jar
   com/ibm/websphere/security/ITSOSpnegoMappingLoginModule.class
   ```

   Here is the expected output of this command:

   ```
   added manifest
   adding:
   com/ibm/websphere/security/ITSOSpnegoMappingLoginModule.class(in
   = 4486) (out= 2251)(deflated 49%)
   ```

9. Copy the ITSOSpnegoMappingLoginModule.jar file to the *app_server_root*/lib/ext directory of all the other nodes in the WebSphere cell.

You have now completed writing, compiling, and creating the .jar file for the JAAS custom mapping login module. The next step is to configure WebSphere Application Server for z/OS to use this login module.

Configuring a sample JAAS custom mapping login module

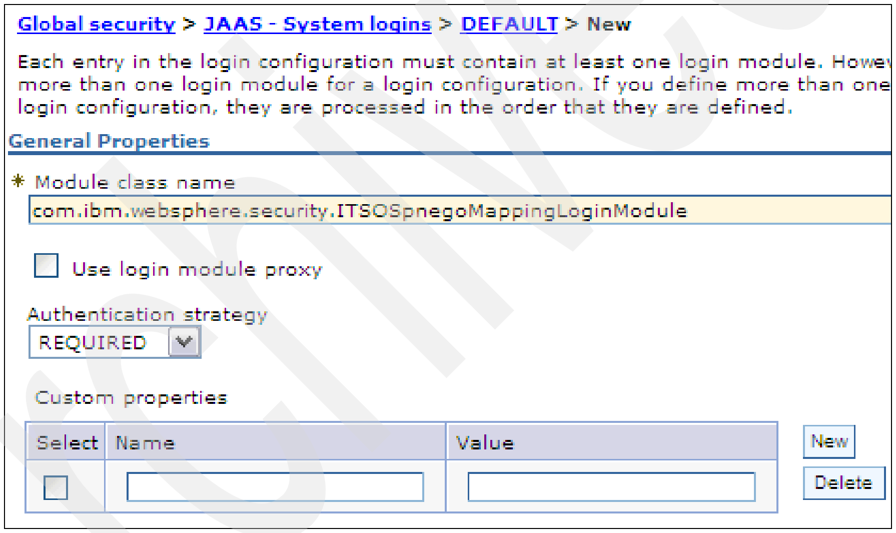
To configure WebSphere Application Server for z/OS to use the JAAS custom mapping login module created in the previous section, follow these steps:

1. Log in to the administrative console and select **Security** → **Global security**. Expand **Java Authentication and Authorization Service**, and click **System logins.**
2. Click **DEFAULT** to open the default login configuration.
3. Click **New** to create a new JAAS login module. For the module class name, enter the following name:

   ```
   com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   ```

   Leave the defaults for the other fields. Figure 14-8 illustrates these settings.

   Click **OK**.



Global security > JAAS - System logins > DEFAULT > New

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

General Properties

* Module class name
com.ibm.websphere.security.ITSOSpnegoMappingLoginModule

☐ Use login module proxy

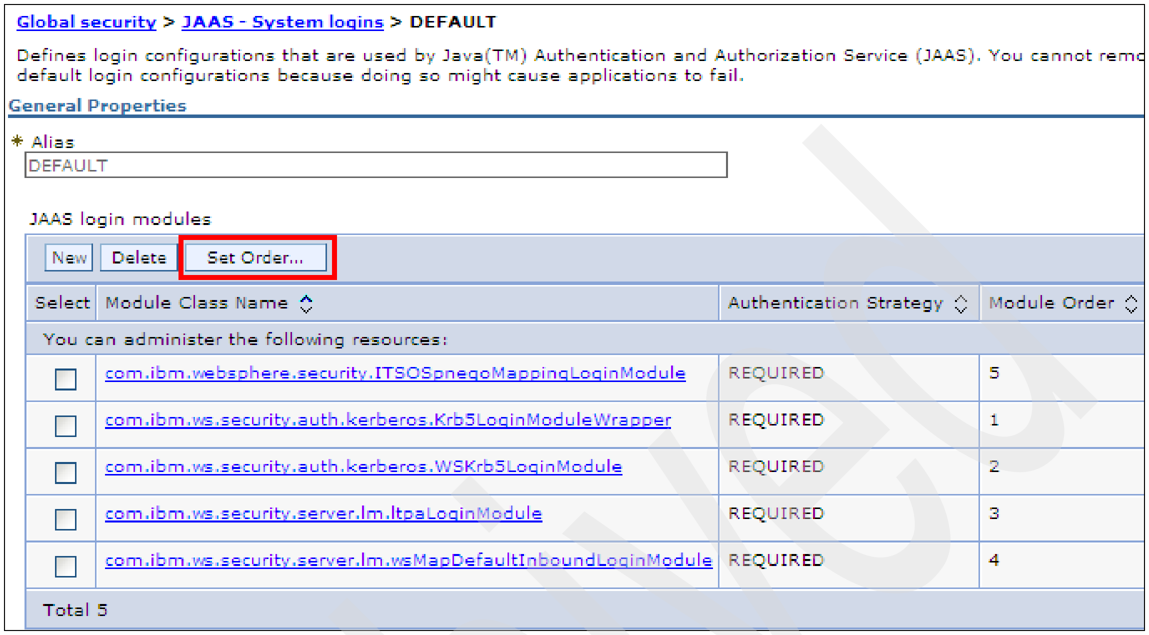
Authentication strategy
REQUIRED

Custom properties

Select	Name	Value
☐		

Figure 14-8 Creating the new JAAS login module

4. Click **Set Order**, as shown in Figure 14-9.



Global security > JAAS - System logins > DEFAULT

Defines login configurations that are used by Java(TM) Authentication and Authorization Service (JAAS). You cannot remo
default login configurations because doing so might cause applications to fail.

General Properties

* Alias

DEFAULT

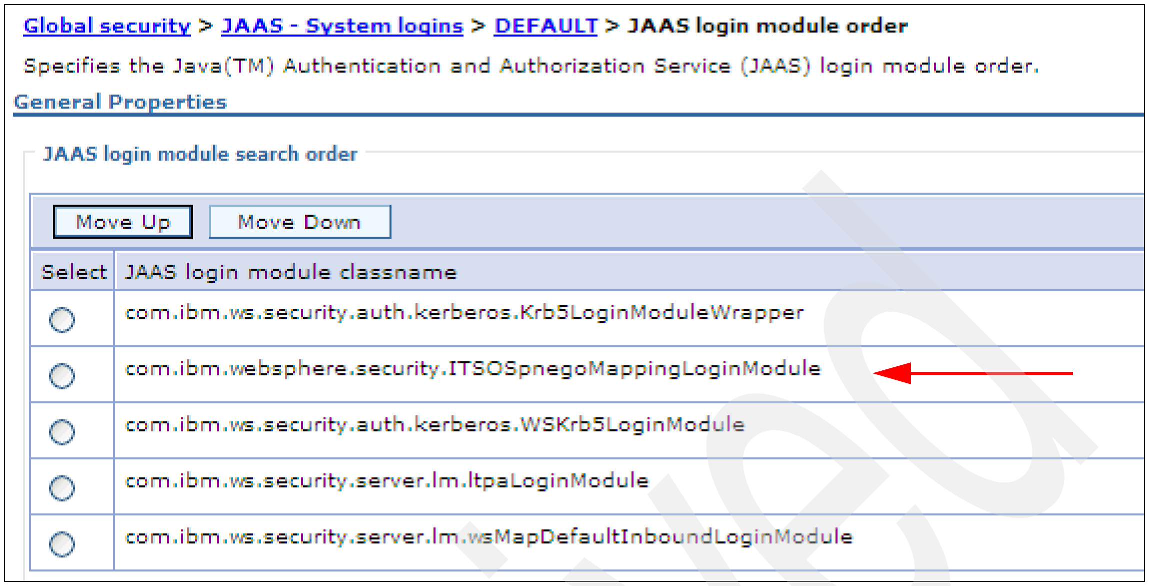
JAAS login modules

Select	Module Class Name	Authentication Strategy	Module Order
You can administer the following resources:			
☐	com.ibm.websphere.security.ITSOSpnegoMappingLoginModule	REQUIRED	5
☐	com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapper	REQUIRED	1
☐	com.ibm.ws.security.auth.kerberos.WSKrb5LoginModule	REQUIRED	2
☐	com.ibm.ws.security.server.lm.ltpaLoginModule	REQUIRED	3
☐	com.ibm.ws.security.server.lm.wsMapDefaultInboundLoginModule	REQUIRED	4
Total 5			

Figure 14-9 Set the order of the login modules

5. Select **com.ibm.websphere.security.ITSOSpnegoMappingLoginModule**, and click **Move up**.

   Repeat this process until the `ITSOSpnegoMappingLoginModule` is immediately before the `com.ibm.ws.security.auth.kerberos.WSKrb5LoginModule` as shown in Figure 14-10.


Global security > JAAS - System logins > DEFAULT > JAAS login module order

Specifies the Java(TM) Authentication and Authorization Service (JAAS) login module order.

General Properties

JAAS login module search order

Select	JAAS login module classname
○	com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapper
○	com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
○	com.ibm.ws.security.auth.kerberos.WSKrb5LoginModule
○	com.ibm.ws.security.server.lm.ltpaLoginModule
○	com.ibm.ws.security.server.lm.wsMapDefaultInboundLoginModule

Figure 14-10 The JAAS login module order

6. Click **OK**.
7. Repeat steps 1-6 for the RMI_INBOUND and WEB_INBOUND system logins.
8. Save and synchronize configuration changes to the nodes.

The configuration of the JAAS custom mapping login module is now complete. Note that this login module is defined at the cell level, so it applies to logins on the deployment manager. Thus, the login module is called and the mapping is performed when you log in to the administrative console.

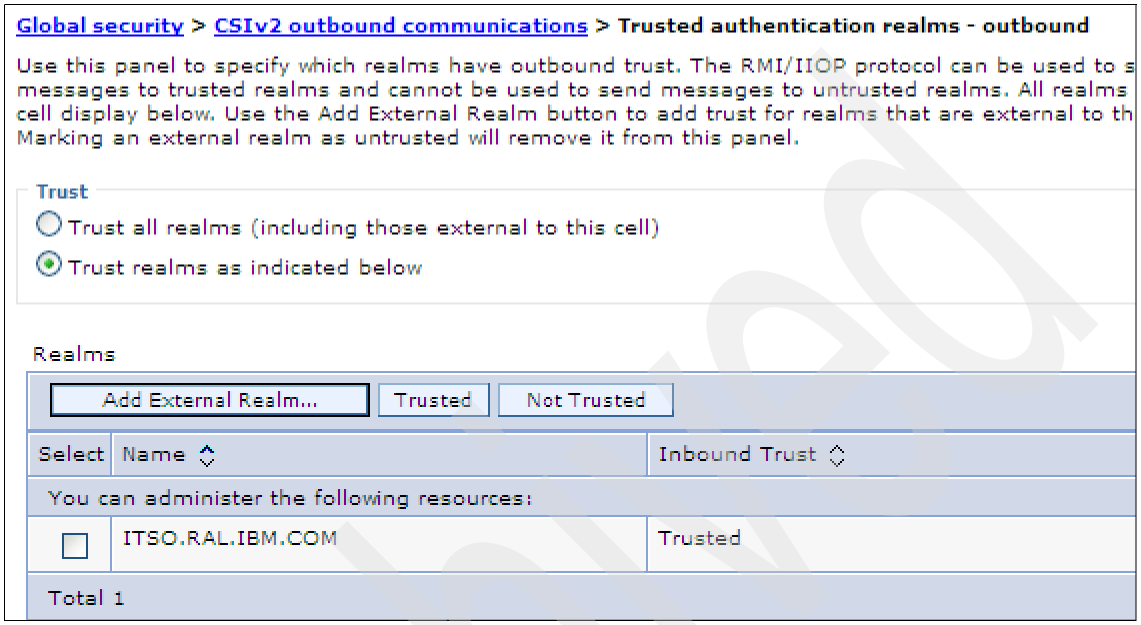
14.2.5 Adding the foreign Kerberos realm to the CSIv2 trusted realms

The AIX Kerberos realm is considered a foreign realm and is not trusted. Therefore, you must add this realm name to the list of trusted authentication realms in WebSphere as follows:

1. Log on to the WebSphere Application Server administrative console and navigate to **Security** → **Global security.** Expand **RMI/IIOP security**, and click **CSIv2 outbound communications**.
2. Click **Trusted authentication realms - outbound**.
3. Click **Add External Realm** and enter the AIX Kerberos realm name (`ITSO.RAL.IBM.COM`).

The AIX Kerberos realm name is now listed as trusted in the list of realms, as shown in Figure 14-11.

Click **OK.**


Global security > CSIv2 outbound communications > Trusted authentication realms - outbound

Use this panel to specify which realms have outbound trust. The RMI/IIOP protocol can be used to s messages to trusted realms and cannot be used to send messages to untrusted realms. All realms cell display below. Use the Add External Realm button to add trust for realms that are external to th Marking an external realm as untrusted will remove it from this panel.

Trust

Trust all realms (including those external to this cell) [✓ Trust realms as indicated below]

Realms

Select	Name	Inbound Trust
You can administer the following resources:		
☐	ITSO.RAL.IBM.COM	Trusted
Total 1		

Figure 14-11 The trusted authentication realms

This completes the configuration for adding the foreign Kerberos realm name to the list of trusted authentication realms.

14.2.6 Installing the TechnologySamples application

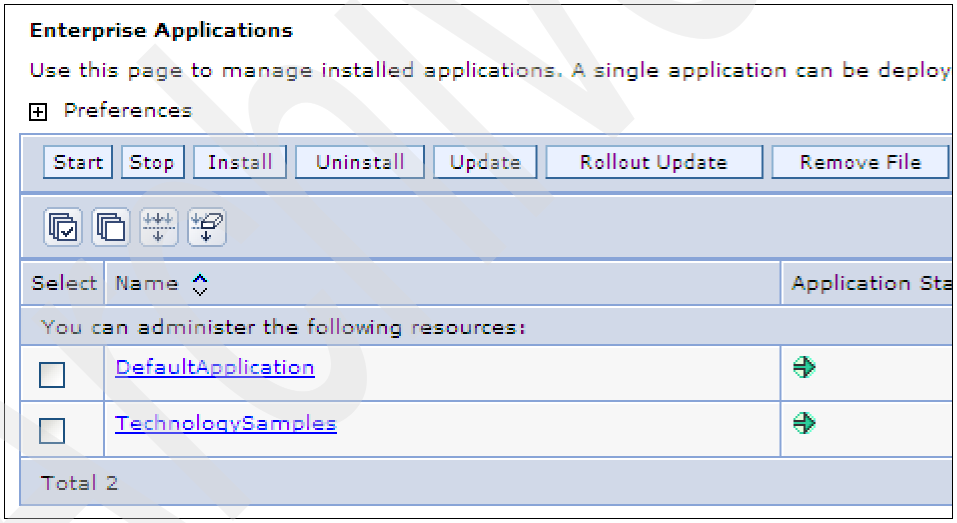
The TechnologySamples application is a sample application that ships with WebSphere Application Server. This application is used to test the cross-realm trust between the thin clients and the WebSphere server.

To install the TechnologySamples application, follow these steps:

1. Log on to the WebSphere Application Server administrative console and navigate to **Applications** → **Application Types** → **WebSphere enterprise applications**.
2. Click **Install**, select **Remote file system**, and enter the following path to the `TechnologySamples.ear` file:

   `app_server_root``/samples/lib/TechSamp/TechnologySamples.ear`

3. Click **Next**.
4. Now, complete these steps:
   a. On the “Preparing for the application installation” page, leave the default option of **Fast Path**, and then click **Next**.
   b. On the “Select installation options” page, leave all the defaults, and click **Next**.
   c. On the “Map modules to servers” page, take the defaults, and click **Next**.
   d. On the “Map virtual hosts for web modules” page, take the defaults, and click **Next**.
5. Click **Finish**.
6. Save and synchronize configuration changes to the nodes.
7. Verify that the application was installed and started successfully by navigating to **Applications** → **Application Types** → **WebSphere enterprise applications**. The TechnologySamples application is listed with a green arrow, as shown in Figure 14-12.


Enterprise Applications

Use this page to manage installed applications. A single application can be deploy

Select	Name	Application Sta
You can administer the following resources:		
☐	DefaultApplication	➔
☐	TechnologySamples	➔
Total 2		

Figure 14-12 Installing the TechnologySamples application

This completes the installation of the TechnologySamples application. The subsequent sections review in more detail how this application works.

14.3 Configuring client1

This section shows how to configure the Application Client for WebSphere Application Server, which includes the thin client software, for client1.

Installing the software: You can find instructions for installing the IBM Application Client for WebSphere Application Server in Appendix D, “Installing the Application Client for WebSphere Application Server” on page 519.

14.3.1 Creating a Kerberos configuration file (krb5.conf) for client1

The `krb5.conf` file contains the Kerberos configuration information, including the location of the KDC, the KDC realm name, and the location of the Kerberos keytab file. You can use an existing `krb5.conf` file, or you can use the administrative scripting task **`createKrbConfigFile`** command on the WebSphere Application Server to create the `krb5.conf` file. For more information about creating the `krb5.conf` file, see 14.2.1, “Configuring Kerberos authentication” on page 355.

For this client, copy the `krb5.conf` file created in 14.2.1, “Configuring Kerberos authentication” on page 355 from the WebSphere Application Server system to the application client. Then make the following changes:

- Set the `default_realm` to `ITSO.RAL.IBM.COM`.
- Set the `default_keytab_name` to `FILE:/etc/krb5.keytab`.

Place `krb5.conf` in the default directory of `/etc/krb5.conf`, with the following exceptions:

- On a Windows operating system, the default location is `c:\winnt\krb5.ini`.
- On a Linux operating system, the default location is `/etc/krb5.conf`.
- On other UNIX-based operating systems, the default location is `/etc/krb5/krb5.conf`.
- On the z/OS operating system, the default location is `/etc/krb5/krb5.conf`.
- On the IBM i operating system, the default location is `/QIBM/UserData/OS400/NetworkAuthentication/krb5.conf`.

Note: If the `krb5.conf` file is not located in the default location, you must set the JVM system property `java.security.krb5.conf` with the correct path and Kerberos configuration file name. For example:

```
-Djava.security.krb5.conf=/etc/test/krb5.conf
```

Example 14-6 shows the `krb.conf` file that we used in this chapter.

Example 14-6 The krb5.conf file used for this chapter

```
[libdefaults]
   default_realm = ITSO.RAL.IBM.COM
   default_keytab_name = FILE:/etc/krb5.keytab
   default_tkt_enctypes = des3-cbc-sha1
   default_tgs_enctypes = des3-cbc-sha1
   forwardable  = true
   renewable  = true
   noaddresses = true
   clockskew  = 300
[realms]
   ITSO.RAL.IBM.COM = {
      kdc = saw921-sys6.itso.ral.ibm.com:88
      default_domain = itso.ral.ibm.com
   }
   ITSO.IBM.COM = {
      kdc = wtsc60.itso.ibm.com:88
      default_domain = itso.ibm.com
   }

[domain_realm]
   .itso.ral.ibm.com = ITSO.RAL.IBM.COM
   .itso.ibm.com = ITSO.IBM.COM
```

14.3.2 Creating a Java SPNEGO client sample

The `SpnegoClientSample.java` application that we use in this chapter has the following options:

- Use the Kerberos credential cache.
- Use the Kerberos principal name and password.
- Use the Windows Kerberos native credential cache. Because the two clients are not Windows operating systems, we did not use this option in this chapter.

You can find the source code for `SpnegoClientSample.java` in “Java code for the SPNEGO client sample” on page 504.

You need to compile the `SpnegoClientSample.java` file, create the `spnego.jar` file that contains the `SpnegoClientSample` class, and then place the `spnego.jar` file in the `/usr/IBM/WebSphere/AppClient/SPNEGO` directory. Use the following steps to accomplish these tasks:

1. Create the directory structure for the Java package using the following command:

   ```
   mkdir
   /IBM/WebSphere/thinClients/SPNEGO/com/ibm/ws/security/auth/kerberos
   ```

2. Change the current directory to the new directory using the following command:

   ```
   cd
   /IBM/WebSphere/thinClients/SPNEGO/com/ibm/ws/security/auth/kerberos
   ```

3. Place the `SpnegoClientSample.java` file in the current directory.
4. Compile the Java file using the following command:

   ```
   /usr/IBM/WebSphere/AppClient/java/bin/javac -classpath
   /usr/IBM/WebSphere/thinClients/com.ibm.ws.ejb.thinclient_7.0.0.jar
   SpnegoClientSample.java
   ```

   This command creates the `SpnegoClientSample` class file.

5. Change the current directory to the SPNEGO directory using the following command:

   ```
   cd /IBM/WebSphere/thinClients/SPNEGO
   ```

6. Create a `.jar` file that includes the newly created class file using the following command:

   ```
   /usr/IBM/WebSphere/AppClient/java/bin/jar -cvf spnego.jar
   com/ibm/ws/security/auth/kerberos/*.class
   ```

   Here is the expected output of this command:

   ```
   added manifest
   adding:
   com/ibm/ws/security/auth/kerberos/SpnegoClientSample.class(in =
   4062) (out= 2146)(deflated 47%)
   ```

14.3.3 Launching the client application using the principal name

Use the sample `rspnego.sh` script to launch SpnegoClientSample as follows:

1. Create the following directory for the thin client code:

   `/usr/IBM/WebSphere/thinClients`

2. Copy the contents of `/usr/IBM/WebSphere/AppClient/runtimes/*` to `/usr/IBM/WebSphere/thinClients`.
3. Create the `rspnego.sh` script. Copy the sample provided in Example 14-7 on page 378, and modify it as necessary for your environment.
4. Place the `rspnego.sh` file in the `/opt/IBM/WebSphere/thinClients` directory.
5. Pass in the Kerberos principal name and password when running SpnegoClientSample.

Example 14-7 Sample rspnego.sh

```
#!/bin/sh
DEFAULTSERVERNAME=wtsc60.itso.ibm.com
SERVERPORT=9637
JAVA=/usr/IBM/WebSphere/AppClient/java
CRBHOME=/usr/IBM/WebSphere/thinClients
ARGS="-host $DEFAULTSERVERNAME -port $SERVERPORT -realm ITSO.IBM.COM
-app snoop -user krb_utle -pwd utlepwd"
APP=com.ibm.ws.security.auth.kerberos.SpnegoClientSample
APP_LOCATION=$CRBHOME/SPNEGO
APP_CP=$APP_LOCATION/spnego.jar
KRB="-Djava.security.krb5.conf=/etc/krb5.conf
-Djava.security.auth.login.config=
$CRBHOME/properties/wsjaas_client.conf"
KRBTRACE="-Dcom.ibm.security.jgss.debug=all
-Dcom.ibm.security.krb5.Krb5Debug=all"
$JAVA/bin/java -classpath "$CRBHOME/*:$APP_CP" $KRBTRACE $KRB $APP
$ARGS
```

When you run the `rspnego.sh`, the client uses the Kerberos principal name and password to authenticate and obtain the SPNEGO token from the KDC. The following steps take place:

1. SpnegoClientSample uses the Kerberos principal name and password to obtain the Kerberos TGT from the AIX KDC. The Kerberos client runtime code in the Application Client software places the Kerberos TGT into the client subject.
2. The Kerberos client code requests a cross realm Kerberos ticket (TGS_REQ) for the z/OS Kerberos realm from the AIX KDC using the Kerberos credential in the client subject.
3. The Kerberos client uses a cross realm ticket to request a Kerberos service ticket for WebSphere Application Server for z/OS (TGS_REQ) from the z/OS KDC. The z/OS KDC returns the Kerberos service ticket and the Kerberos client forms the SPNEGO token.
4. The SpnegoClientSample inserts the SPNEGO token into the HTTP header and sends the HTTP request to WebSphere Application Server for z/OS server for authentication.

If the client authenticates successfully to WebSphere Application Server for z/OS, the client displays the following message:

```
Successfully connect to the URL
http://wtsc60.itso.ibm.com:9637/snoop
```

Note that if you try to access the snoop servlet from a Web browser that does not support SPNEGO, the following message displays in the browser:

```
SPNEGO authentication is not supported on this client.
```

14.3.4 Launching the sample using the Kerberos credential cache

Update the `rspnego.sh` script to use the Kerberos credential cache option by removing the password (as shown in Example 14-8) so that the client can participate in the SSO.

Example 14-8 Sample rspnego.sh

```
#!/bin/sh
DEFAULTSERVERNAME=wtsc60.itso.ibm.com
SERVERPORT=9637
JAVA=/usr/IBM/WebSphere/AppClient/java
CRBHOME=/usr/IBM/WebSphere/thinClients
ARGS="-host $DEFAULTSERVERNAME -port $SERVERPORT -realm ITSO.IBM.COM
-app snoop -user krb_utle"
APP=com.ibm.ws.security.auth.kerberos.SpnegoClientSample
```

```
APP_LOCATION=$CRBHOME/SPNEGO
APP_CP=$APP_LOCATION/spnego.jar
KRB="-Djava.security.krb5.conf=/etc/krb5.conf
-Djava.security.auth.login.config=
$CRBHOME/properties/wsjaas_client.conf"
KRBTRACE="-Dcom.ibm.security.jgss.debug=all
-Dcom.ibm.security.krb5.Krb5Debug=all"
$JAVA/bin/java -classpath "$CRBHOME/*:$APP_CP" $KRBTRACE $KRB $APP
$ARGS
```

First, use the Java **kinit** command to obtain the Kerberos TGT from the AIX KDC for user `krb_utle` as shown in Example 14-9.

Example 14-9 Using the kinit command to obtain the Kerberos TGT

```
$ . /usr/IBM/WebSphere/AppClient/java/jre/bin/kinit krb_utle
Password for krb_utle@ITSO.RAL.IBM.COM:

Done!
New ticket is stored in cache file /home/krb_utle/krb5cc_krb_utle
```

The **kinit** command places the Kerberos credential cache in the default location. You can use the Java **klist** command to check the Kerberos ticket lifetime, the Kerberos credential cache file name and location, as shown in Example 14-10.

Example 14-10 Using the klist command to check the Kerberos ticket lifetime

```
$ . /usr/IBM/WebSphere/AppClient/java/jre/bin/klist

Credentials cache: /home/krb_utle/krb5cc_krb_utle
Default principal: krb_utle@ITSO.RAL.IBM.COM
Number of entries: 1

[1] Service principal: krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
        Valid starting: Tuesday, July 21, 2009 at 5:12:37 PM
        Expires: Wednesday, July 22, 2009 at 5:12:37 PM
```

Note: The default location of the Kerberos credential cache file depends on which operating system you use. The user credential cache is located by searching for the following files in the following order:

1. The file referenced by the Java property `KRB5CCNAME`
2. The `<`*`user.home`*`>/krb5cc_<`*`user.name`*`>` file
3. The `<`*`user.home`*`>/krb5cc` file (if `<`*`user.name`*`>` cannot be obtained)

Note: If the Kerberos credential cache is not in the default location, set the Java property KRB5CCNAME as a URL, for example:

```
export KRB5CCNAME=FILE:/home/utle/krb5cc_krb_utle
```

When you run `rspnego.sh`, the client uses the Kerberos credential cache to authenticate and obtain the SPNEGO token from the KDC as follows:

1. The SpnegoClientSample uses the Kerberos credential cache to obtain the Kerberos TGT from the AIX KDC. The Kerberos client runtime code places the Kerberos TGT in the client subject.
2. The Kerberos client code requests a cross realm Kerberos ticket (TGS_REQ) for the z/OS Kerberos realm from the AIX KDC using the Kerberos TGT obtained in the previous step.
3. The Kerberos client uses a cross realm ticket to request a Kerberos service ticket for WebSphere Application Server for z/OS (TGS_REQ) from the z/OS KDC. The z/OS KDC returns the Kerberos service ticket, and the Kerberos client forms the SPNEGO token.
4. The SPNEGO token returned in the previous step is inserted into the HTTP header.
5. The SpnegoClientSample sends the HTTP request to WebSphere Application server for accessing the snoop application.

If the client authenticates successfully to the WebSphere Application Server for z/OS server, the following message is output to the command line:

```
Successfully connect to the URL
http://wtsc60.itso.ibm.com:9637/snoop
```

14.4 Installing and configuring client2

This section shows how to install and configure the Application Client for WebSphere Application Server, which includes the thin client software, for client2.

14.4.1 Installing the Application Client and thin client

Use the instructions in Appendix D, “Installing the Application Client for WebSphere Application Server” on page 519 to install the Application Client and thin client software. Install this software at `/opt/IBM/WebSphere/AppClient`.

Use the `basicCalculator` script that is shipped as part of the thin client technology samples in the following directory:

```
/opt/IBM/WebSphere/AppClient/samples/bin/TechnologySamplesThinClient
/BasicCalculator
```

For a Java client or thin client application to use Kerberos authentication, it must reference the `wsjaas_client.conf` file. The `basicCalculator` script does not have a property for referencing this file. Therefore, you need to add this property to the `basicCalculator` script so that it can be used with Kerberos authentication.

To update the `basicCalculator` script, follow these steps:

1. Edit the `basicCalculator` script, and search for `com.ibm.websphere.samples`.
2. Add **“`$JAASSOAP`”** in front of the `com.ibm.websphere.samples` string, as follows:

```
"$CLIENTSAS" "$CLIENTSSL" "$JAASSOAP"
com.ibm.websphere.samples.technologysamples.basiccalcthinclient.B
asicCalculatorClientThinMain add 1 2
```

14.4.2 Creating a Kerberos configuration file (krb5.conf) for client2

The `krb5.conf` file contains the Kerberos configuration information, including the location of the KDC, the KDC realm name, and the location of the Kerberos keytab file. You can use an existing `krb5.conf` file, or you can use the administrative scripting task **`createKrbConfigFile`** command on WebSphere Application Server to create the `krb5.conf` file. For more information about how to create the `krb5.conf` file, see “Creating the Kerberos configuration file (krb5.conf)” on page 355.

For this client, use the `krb5.conf` file created in “Creating the Kerberos configuration file (krb5.conf)” on page 355 but change the following settings:

- Set the `default_realm` to `ITSO.RAL.IBM.COM`
- Set the `default_keytab_name` to `FILE:/etc/krb5.keytab`

Place the `krb5.conf` at the default local `/etc/krb5.conf`, with the following exceptions:

- On a Windows operating system, the default location is `c:\winnt\krb5.ini`.
- On a Linux operating system, the default location is `/etc/krb5.conf`.
- On other UNIX-based operating systems, the default location is `/etc/krb5/krb5.conf`.
- On the z/OS operating system, the default location is `/etc/krb5/krb5.conf`.
- On the IBM i operating system, the default location is `/QIBM/UserData/OS400/NetworkAuthentication/krb5.conf`.

Note: If the `krb5.conf` file is not located in the default location, you must set `com.ibm.COBRA.krb5ConfigFile` in the `sas.client.props` file with the correct path and Kerberos configuration file name.

Example 14-11 shows the sample `krb5.conf` that we used for this chapter.

Example 14-11 Sample krb5.conf used for this chapter

```
[libdefaults]
   default_realm = ITSO.RAL.IBM.COM
   default_keytab_name = FILE:/etc/krb5.keytab
   default_tkt_enctypes = des3-cbc-sha1
   default_tgs_enctypes = des3-cbc-sha1
   forwardable  = true
   renewable  = true
   noaddresses = true
   clockskew  = 300
[realms]
   ITSO.RAL.IBM.COM = {
      kdc = saw921-sys6.itso.ral.ibm.com:88
      default_domain = itso.ral.ibm.com
   }
   ITSO.IBM.COM = {
      kdc = wtsc60.itso.ibm.com:88
      default_domain = itso.ibm.com
   }

[domain_realm]
   .itso.ral.ibm.com = ITSO.RAL.IBM.COM
   .itso.ibm.com = ITSO.IBM.COM
```

14.4.3 Updating the sas.client.props file

Update the `sas.client.props` file to use the Kerberos token to authenticate to the WebSphere Application Server for z/OS server. In the `/opt/IBM/WebSphere/AppClient/properties/sas.client.props` file, set the `com.ibm.CORBA.authenticationTarget` property to `KRB5`. When Kerberos authentication is used, the client sends the Kerberos token to WebSphere Application Server for authentication. The client's clear text password never leaves the client system.

Note: The Java client or thin client application must have the `wsjaas_client.conf` file. If you do not use the **`launchClient`** command, you must set the Java system property in your script as follows:

```
-Djava.security.auth.login.config=/usr/IBM/WebSphere/AppClient/pr
operties/wsjaas_client.config
```

14.4.4 Running the basicCalculator script

This section describes the different methods for running the `basicCalculator` script, depending on your environment, which include:

- Using Kerberos credential cache as a login source
- Using prompt as a login source
- Using stdin as a login source
- Using properties as a login source
- Using krb5Ccache:prompt as a login source

You can update the `sas.client.props` file with the login source that works for your environment.

Using Kerberos credential cache as a login source

Using the Kerberos credential cache, `krb5Ccache`, as the login source allows the client to participate in SSO. The user runs the **`kinit`** command once to cache the credentials, thus eliminating the need to enter the user ID and password again for authentication to WebSphere Application Server for as long as the Kerberos credential cache is still valid.

To use the Kerberos credential cache as the login source, follow these steps;

1. Get the Kerberos TGT using the Java **`kinit`** command. The **`kinit`** command places the Kerberos credential cache in the default location as follows:

   ```
   #. /opt/IBM/WebSphere/AppClient/java/jre/bin/kinit krb_utle
   Password for krb_utle@ITSO.RAL.IBM.COM:

   Done!
   New ticket is stored in cache file /home/krb_utle/krb5cc_krb_utle
   ```

2. Use the Java **`klist`** command to check the Kerberos ticket lifetime, as well as the Kerberos credential cache file name and location as follows:

   ```
   #. /opt/IBM/WebSphere/AppClient/java/jre/bin/klist

   Credentials cache: /home/krb_utle/krb5cc_krb_utle
   Default principal: krb_utle@ITSO.RAL.IBM.COM
   ```

```
Number of entries: 1

[1] Service principal: krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
        Valid starting: Tuesday, July 21, 2009 at 5:12:37 PM
        Expires: Wednesday, July 22, 2009 at 5:12:37 PM
```

3. In the `sas.client.props` file, set the `com.ibm.CORBA.loginSource` property to `krb5Ccache`.

   The default location of the Kerberos credential cache file depends on which operating system you use. The user credential cache is located in the following files in the following order:

   a. The file referenced by the Java property `KRB5CCNAME`
   b. The <*user.home*>/`krb5cc_`<*user.name*> file
   c. The <*user.home*>/`krb5cc file` (if <*user.name*> cannot be obtained)

   Note: If the Kerberos credential cache is not at the default location, set the `com.ibm.CORBA.krb5CcacheFile` property as a URL, for example:

   ```
   com.ibm.CORBA.krb5CcacheFile=FILE:/home/krb_utle/krb5cc_krb_utle
   ```

4. In the `wsjaas_client.conf` file, update the `WSKRB5Login` entry to use the Kerberos credential cache as follows:

   ```
   WSKRB5Login{

   com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapperClient
   required credsType=INITIATOR useFirstPass=false
   forwardable=false renewable=false noAddress=false;
   }
   ```

5. Now, you can run the `basicCalculator` script, and if it authenticates successfully to the WebSphere Application Server for z/OS server, the following output displays:

   ```
   --Narrowing... Done.
   --Creating Home... Done.
   Result: 1+2 = 3
   ```

Using prompt as a login source

The default `com.ibm.CORBA.loginSource` property is set to `prompt`. When you run the `basicCalculator` using `prompt` as a login source, it prompts you for the user ID and password.

Using stdin as a login source

When you set the `com.ibm.CORBA.loginSource` property to `stdin` and run the `basicCalculator` script, you can enter the user ID and password using `stdin`.

Using properties as a login source

Set the `com.ibm.CORBA.loginSource` property to `properties` and set the `com.ibm.CORBA.loginUserid` and `com.ibm.CORBA.loginPassword` properties to the Kerberos principal name and password. Then, when you run the `basicCalculator` script, it uses the user ID and password in the `sas.client.props` file to authenticate to the WebSphere Application Server for z/OS server.

Using krb5Ccache:prompt as a login source

Set the `com.ibm.CORBA.loginSource` property to `krb5Ccache:prompt`. When you run the `basicCalculator` script, it uses the Kerberos credential cache to authenticate to the WebSphere Application Server for z/OS first. If that authentication fails, then it falls back to prompt for a user ID and password.

14.5 Validating the scenario

You need to verify the scenario by examining the following information as described in this section:

- Messages displayed in the WebSphere Application Server for z/OS job logs when the scenario completes successfully
- The output on the clients

14.5.1 Validating the scenario on the server

This section describes how to verify that the scenario was successful on the server. After executing the client, examine the job logs of the servant region for the application server on z/OS. You should see the debug messages from the JAAS custom mapping login module.

The constructor for the login module is called to create an instance of the `ITSOSpnegoMappingLoginModule` class. The following debug statements should display in the servant region log:

```
Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   ITSOSpnegoMappingLoginModule() entry
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   ITSOSpnegoMappingLoginModule() exit
```

The `initialize()` method of the login module is called to initialize the relevant parameters, resulting in the debug statements shown in Example 14-12.

Example 14-12 The debug statements for the initialize() method of the login module

```
Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   initialize(subject = "Subject:
   Principal: krb_utle@ITSO.RAL.IBM.COM
   Private Credential:
 --- GSSCredential ---
   Number of mehanism credentials: 2
O[1] SPNEGOCredential
 Owner:krb_utle@ITSO.RAL.IBM.COM
 Usage:Initiate Only
 Start Time:7/22/09 5:06 PM
 InitLifetime:86349 Seconds
 AcceptLifetime:0Seconds
O[2] Kerberos credential, mechanism: 1.2.840.113554.1.2.2
   Owner:krb_utle@ITSO.RAL.IBM.COM
   Usage:initiate only
   StartTime:7/22/09 5:06 PM
   InitLifeTime:86,349 seconds
   AcceptLifeTime:unknown
   Krb5Client:krb_utle@ITSO.RAL.IBM.COM
   Krb5Server:krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
 --- End of GSSCredential ---
O  Private Credential:
Otoken name:         com.ibm.wsspi.wssecurity.token.krbAuthnToken
 version:            1
 hashCode:           1676089970
 uniqueId:           krb_utle1260979952
 kerberos principal: krb_utle
 realm:              ITSO.RAL.IBM.COM
 expiration:         Thu Jul 23 17:05:32 GMT 2009
 renew until:        Wed Jul 29 17:05:32 GMT 2009
 isReadOnly:         false
 isAddressless:      true
 isForwardable:      true
 isRenewable:        true
 KerberosTicket:     Ticket (hex) =
 0000: 61 82 01 2e 30 82 01 2a  a0 03 02 01 05 a1 12 1b
a...0...........
 0010: 10 49 54 53 4f 2e 52 41  4c 2e 49 42 4d 2e 43 4f
.ITSO.RAL.IBM.CO
```

```
 0020: 4d a2 25 30 23 a0 03 02  01 02 a1 1c 30 1a 1b 06
M..0.......0...
 0030: 6b 72 62 74 67 74 1b 10  49 54 53 4f 2e 52 41 4c
krbtgt..ITSO.RAL
 0040: 2e 49 42 4d 2e 43 4f 4d  a3 81 e7 30 81 e4 a0 03
.IBM.COM...0....
 0050: 02 01 10 a1 03 02 01 01  a2 81 d7 04 81 d4 34 50
..............4P
 0060: 32 2c 96 4c 20 fa 51 d2  be 81 bf ce 09 7d ed 23
2..L..Q.........
 0070: 64 2d ba 00 38 66 a7 6d  c3 b0 db 6f f5 1f de 66
d...8f.m...o...f
 0080: d5 9d 78 19 af ec 1a cb  07 21 48 53 30 b4 84 48
..x.......HS0..H
 0090: bf f5 3c 13 b8 be 6e f7  81 03 9d 4d 83 22 cb 6b
......n....M...k
 00a0: 95 03 54 2f e8 46 a9 60  93 70 27 76 86 0a 2d ee
..T..F...p.v....
 00b0: 83 4e f3 01 f1 40 80 71  01 d6 51 86 13 50 da e3
.N.....q..Q..P..
 00c0: e0 9e 56 33 a3 0e 71 af  f9 87 9f 5c 66 d6 e3 87
..V3..q.....f...
 00d0: 0d d9 b8 50 cf fa f5 ad  37 ee 71 e2 0f 9f 63 bc
...P....7.q...c.
 00e0: 9b ac 77 e4 ed d2 1b 02  01 8c ba 3d 3a 21 b6 af
..w.............
 00f0: dc fb fc 1a 69 f8 09 fe  ed fe 93 c9 89 06 23 ef
....i...........
 0100: 7f a5 6f ef 49 d5 3b 65  51 fe 24 b9 9c bb c0 c9
..o.I..eQ.......
 0110: 77 27 85 0a 83 26 60 0b  33 24 b6 e9 2b 6f 67 3e
w.......3....og.
 0120: c1 82 e5 c8 45 68 a1 1c  18 4f ab 05 2f e1 8f 55
....Eh...O.....U
 0130: f4 01
 Client Principal = krb_utle@ITSO.RAL.IBM.COM
 Server Principal = krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
 Session Key = EncryptionKey: keyType=16 keyBytes (hex dump)=
 0000: 8f 6b fd f7 d9 1a ea 01  13 b3 80 76 e5 e0 a4 86
.k.........v....
 0010: e5 e6 86 2f d9 ef c2 5e
0Forwardable Ticket true
 Forwarded Ticket true
 Proxiable Ticket false
 Proxy Ticket false
 Postdated Ticket false
 Renewable Ticket true
 Initial Ticket true
 Auth Time = Wed Jul 22 17:05:32 GMT 2009
 Start Time = Wed Jul 22 17:05:34 GMT 2009
 End Time = Thu Jul 23 17:05:32 GMT 2009
 Renew Till = Wed Jul 29 17:05:32 GMT 2009
 Client Addresses  Null
 GSSCredential:
 --- GSSCredential ---
   Number of mehanism credentials: 2
O[1] SPNEGOCredential
 Owner:krb_utle@ITSO.RAL.IBM.COM
 Usage:Initiate Only
 Start Time:7/22/09 5:06 PM
 InitLifetime:86349 Seconds
 AcceptLifetime:0Seconds
O[2] Kerberos credential, mechanism: 1.2.840.113554.1.2.2
   Owner:krb_utle@ITSO.RAL.IBM.COM
   Usage:initiate only
   StartTime:7/22/09 5:06 PM
   InitLifeTime:86,349 seconds
   AcceptLifeTime:unknown
   Krb5Client:krb_utle@ITSO.RAL.IBM.COM
   Krb5Server:krbtgt/ITSO.RAL.IBM.COM@ITSO.RAL.IBM.COM
 --- End of GSSCredential ---
0", callbackHandler =
"javax.security.auth.login.LoginContext$SecureCallbackHandler@200e200e"
, sharedState = "{Callback=[Ljavax.secur
 ity.auth.callback.Callback;@2fa72fa7}", options = "{}")
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   initialize() exit
```

The `login()` method of the login module is then called to authenticate the subject. These debug statements show that the Kerberos principal name, `krb_utle@ITSO.RAL.IBM.COM`, is mapped to the identity of `utle`, as shown in Example 14-13.

Example 14-13 The debug statements for the login() method of the login module

```
Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   ITSOSpnegoMappingLoginModule.login() entry
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   Kerberos principal name: krb_utle@ITSO.RAL.IBM.COM
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
```

```
   mapUid: utle
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   Add a mapping user ID to Hashtable, mapping ID = utle
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   login() custom properties =
{com.ibm.wsspi.security.cred.userId=utle}
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   ITSOSpnegoMappingLoginModule.login() exit
```

After the `login()` method completes successfully, the `commit()` method is invoked to commit the changes to the subject. The following debug statements show this call:

```
Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   commit()
 Debug: com.ibm.websphere.security.ITSOSpnegoMappingLoginModule
   commit()
```

14.5.2 Validating the scenario on the client

For thin application client 1, the following message displays upon success:

```
Successfully connect to the URL
http://wtsc60.itso.ibm.com:9637/snoop
```

For thin application client 2, the following output displays upon success:

```
--Narrowing... Done.
--Creating Home... Done.
Result: 1+2 = 3
```

15

SSO to WebSphere Application Server for z/OS and DB2 using Microsoft Kerberos KDC and z/OS KDC trust

This chapter discusses a an end-to-end single sign-on (SSO) scenario in which a client that is running in a Windows client workstation accesses an application that is running in WebSphere Application Server for z/OS, which then in turn accesses DB2. The scenario in this chapter uses the client Kerberos delegate credentials to authenticate to all products so that all systems know the client user. This chapter includes the following topics:

- Scenario overview
- Configuring a cross-trust realm between two Kerberos realms
- Configuring DB2 for Kerberos
- Configuring WebSphere Application Server
- Configuring SPNGEO on the client browser
- Validating the scenario

15.1 Scenario overview

This scenario allows a client to access a protected Web page and the back-end resource that it touches (DB2) without being prompted for additional user information. This scenario also illustrates trust between key distribution centers (KDCs).

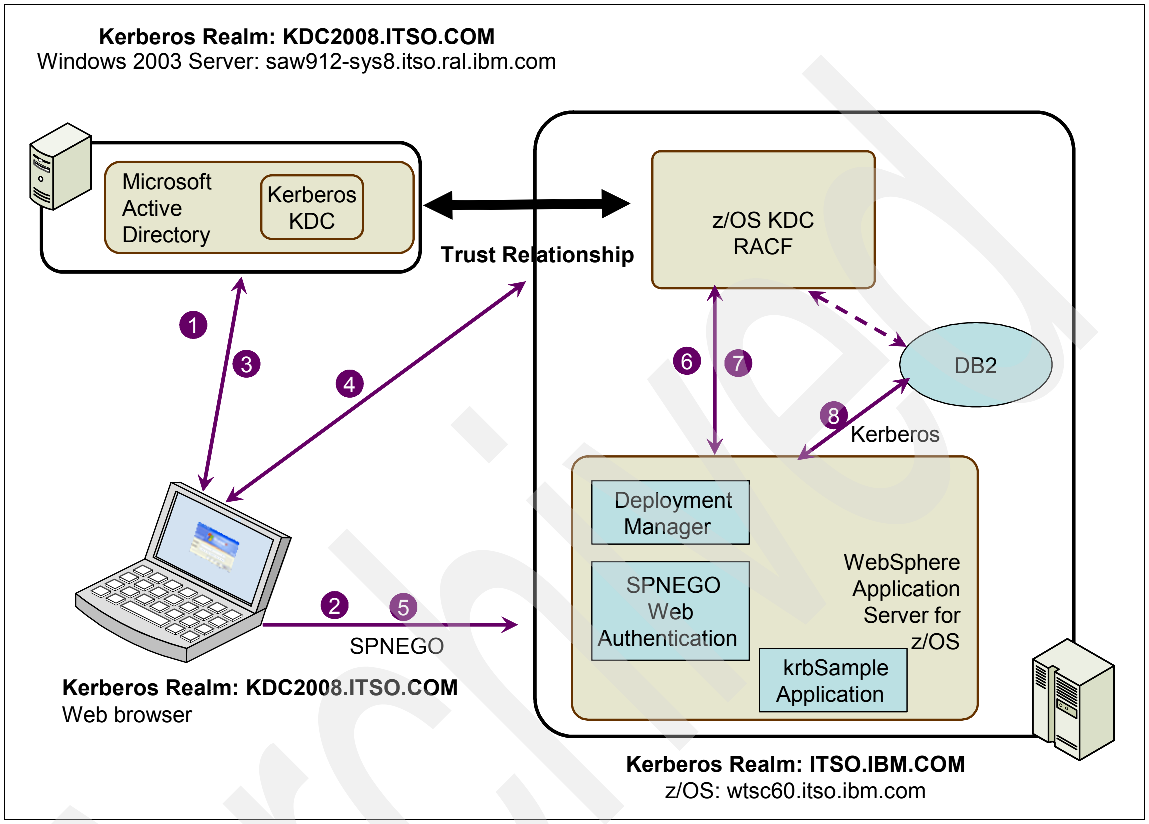
When the user accesses the Web application, the user’s Kerberos delegation credentials are sent to WebSphere Application Server. WebSphere Application Server is configured to use only the z/OS KDC, but the z/OS KDC trusts the Microsoft Kerberos KDC. So, the z/OS KDC can validate the client’s SPNEGO token. This identity is mapped to a RACF ID minus the Kerberos realm. Then, the WebSphere application accesses the back-end resource, DB2, providing the resource with the original client’s identity providing the solution that allows the client user’s credentials to flow to all the processes that are involved to validate the user’s identity.

Note: This scenario features the following components:

- SSO from a Web browser client to WebSphere Application Server for z/OS and then to DB2
- Two KDCs, one KDC on Microsoft Active Directory and the other KDC on z/OS, with trust established between them
- A DB2 data source in WebSphere Application Server to use Kerberos credentials
- Web browser client to use SPNEGO token for authentication
- WebSphere Application Server configuration:
  - User registry: RACF
  - Authentication mechanism: Kerberos and LTPA
  - SPNEGO Web Authentication
  - Kerberos authentication and configuration file
  - Sample application: DefaultApplication (snoop servlet), krbSample

15.1.1 Step-by-step flow of this scenario

Figure 15-1 illustrates the flow of this scenario.



Kerberos Realm: KDC2008.ITSO.COM
Windows 2003 Server: saw912-sys8.itso.ral.ibm.com
Microsoft Active Directory
Kerberos KDC
Trust Relationship
z/OS KDC RACF
DB2
1
3
4
6
7
8
Kerberos
Deployment Manager
SPNEGO Web Authentication
WebSphere Application Server for z/OS
krbSample Application
2
5
SPNEGO
Kerberos Realm: KDC2008.ITSO.COM
Web browser
Kerberos Realm: ITSO.IBM.COM
z/OS: wtsc60.itso.ibm.com

Figure 15-1 Cross-trust scenario

In this scenario, the following actions occur:

1. To begin, the user logs on to the Microsoft Domain controller from the workstation. After the user enters a user name (TAMT) and password, Windows logon sends the information to the workstation local security authority, which passes the request to the Kerberos authentication package. The client sends an initial authentication request (AS_REQ), which includes the user's credentials and an encrypted time stamp to the KDC. This is a request for authentication and obtains a ticket-granting ticket (TGT). The first domain controller in the domain generates the `krbtgt/`*`domain_name`* ticket. The KDC uses the secret key to decrypt the time stamp and issues a TGT to the client. The AS_REP is encrypted with the user's key and returned to the user.

2. Next, the user attempts to access a Web application from a Web browser. This action sends an HTTP get request to WebSphere Application Server.

   WebSphere Application Server, using SPNEGO Web authentication, responds to the client browser with an HTTP challenge header 401 containing the `Authenticate: Negotiate` status.

3. The client browser recognizes the negotiate header because it has been configured to support integrated Windows authentication. The client parses the requested URL for the host name and sees the domain is for the z/OS KDC realm. The client uses the host name as part of the target service principal name (SPN) to request a cross-realm ticket for the z/OS Kerberos realm from the Kerberos ticket-granting service (TGS) in the Microsoft Kerberos KDC (TGS_REQ).
4. The client uses a cross-realm ticket to request a Kerberos service ticket for WebSphere Application Server (TGS_REQ) from the z/OS KDC.
5. The service ticket proves the user’s identity. The client responds to the WebSphere Application Server `Authenticate: Negotiate` challenge with the SPNEGO token in the request HTTP header. The SPNEGO token is passed to WebSphere Application Server for authentication.
6. WebSphere Application Server validates the client SPNEGO token and checks with RACF for the user’s authorization:
   a. WebSphere Application Server, using SPNEGO Web authentication, sees the HTTP header with the SPNEGO token. It extracts the client’s delegated Kerberos credentials and gets the identity (principal) of the user.
   b. WebSphere Application Server sends the user's identity, TAMT at the Microsoft Active Directory realm, to RACF to identify the ID. After the identification process is executed, WebSphere executes the enterprise application code (servlets, JSPs, EJBs, and so on) and checks authorizations.
   c. If all access is granted, WebSphere Application Server executes the enterprise application code.
7. WebSphere Application Server obtains a Kerberos service ticket for DB2 (TGS_REQ) on behalf of the client and uses the client’s delegated Kerberos credentials.
   a. WebSphere Application Server use the client’s delegated Kerberos credentials to request the cross realm ticket on behalf of the client from the Kerberos KDC.
   b. WebSphere Application Server uses the cross realm ticket to request a service ticket from the z/OS KDC for DB2.
8. The Kerberos token returned from the z/OS KDC (TGS_REP) is added to the DB2 authentication request and sent to DB2 for authentication.

15.1.2 Scenario components

This scenarios uses the following main components:

- The application server

  WebSphere Application Server for z/OS v7.0.0.5. The operating system is z/OS v1r9.

- The back-end resource

  DB2 for z/OS v9.1.

- The Kerberos KDC

  Microsoft Windows Server 2003 SP2 with the Microsoft Kerberos KDC and z/OS KDC v1r9 configured to trust each other

- The client workstation

  Microsoft Windows XP Professional 2002 SP2. The Windows Server 2003 Resource Kit Tools are installed. It runs Microsoft Internet Explorer v6.0.2900 and Mozilla Firefox v3.0.6.

Download material: This scenario uses an application called krbSample that is installed on WebSphere Application Server. The application is included in the Web material for this book. See Appendix E, “Additional material” on page 525 for information about obtaining this material.

15.1.3 Basic requirements for this scenario

A working domain controller and at least one client computer in that domain are required for this solution. Using SPNEGO from the domain controller does not work.

This scenario assumes the following components are in place:

- A functioning Microsoft Windows 2000/2003 Active Directory Domain including:
  - Domain controller
  - Client workstation

  Users can log in to the client workstation.

- A z/OS KDC v1r9 is configured and running (see Chapter 2, “Setting up a KDC on a z/OS system” on page 13).
- The users in the RACF registry can access successfully the protected resources for WebSphere Application Server using a native authentication mechanism, such as basic authentication or forms authentication.

- RACF users can access the DB2 server and have authority to read or write to databases, based on enterprise application requirements.
- A DB2 version that supports Kerberos is installed to host the application database. In this scenario, we used DB2 9.1.
- Windows 200x Support Tools are installed on the Microsoft Windows Server. These include the **setspn**, the **ktpass**, and the **adsiedit** tools. These support tools are available on the Windows Server installation CD or from the Microsoft Web site:

  http://www.microsoft.com/downloads/details.aspx?familyid=6EC50B78-8BE1-4E81-B3BE-4E7AC4F0912D

We also recommend that you install the Windows Server 2003 Resource Kit Tools, which provide the **kerbtray.exe** utility that allows you to view Kerberos tickets. You can install this package on the user workstation:

http://www.microsoft.com/downloads/details.aspx?FamilyID=9d467a69-57ff-4ae7-96ee-b18c4790cffd

15.1.4 Summary of implementation steps

To implement this scenario requires the following basic steps:

1. Configuring a cross-trust realm between two Kerberos realms.
2. Configuring DB2 for Kerberos.
3. Configuring WebSphere Application Server.
4. Configuring SPNGEO on the client browser.
5. Validating the scenario.

15.2 Configuring a cross-trust realm between two Kerberos realms

This scenario uses a cross-realm trust between two Kerberos realms. Users from one KDC are accepted automatically because the KDC trusts the tokens that are provided by the first Kerberos realm. You can set up a Microsoft Kerberos realm and z/OS Kerberos cross-realm trust as described in Chapter 6, “Setting up trust between a Microsoft Kerberos KDC and a z/OS KDC” on page 101.

After you set up the Kerberos cross-realm trust, a client workstation user can access Web pages in WebSphere Application Server that are identified by SPNEGO Web filters, even though WebSphere Application Server does not directly trust or know the Microsoft Kerberos realm.

15.3 Configuring DB2 for Kerberos

This section explains how to configure DB2 and WebSphere Application Server so that a WebSphere application can access a DB2 database using Kerberos credentials for authentication.

First, you need to add a Kerberos segment to the DB2's user ID in RACF, which allows Kerberos authentication for the local z/OS DB2. Then, you can configure a data source for the DB2 database in WebSphere that allows a connection to the database using either a user ID and password or a Kerberos token.

For more information about configuring WebSphere Application Server to connect to a database using Kerberos, see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/tdat_db2kerberos.html

For more information about DB2 Kerberos configuration, refer to the DB2 Information Center:

http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/topic/com.ibm.db29.doc/db2prodhome.htm

15.3.1 Configuring the local z/OS DB2 to use Kerberos

To have DB2 on z/OS use Kerberos, you need to do the following tasks:

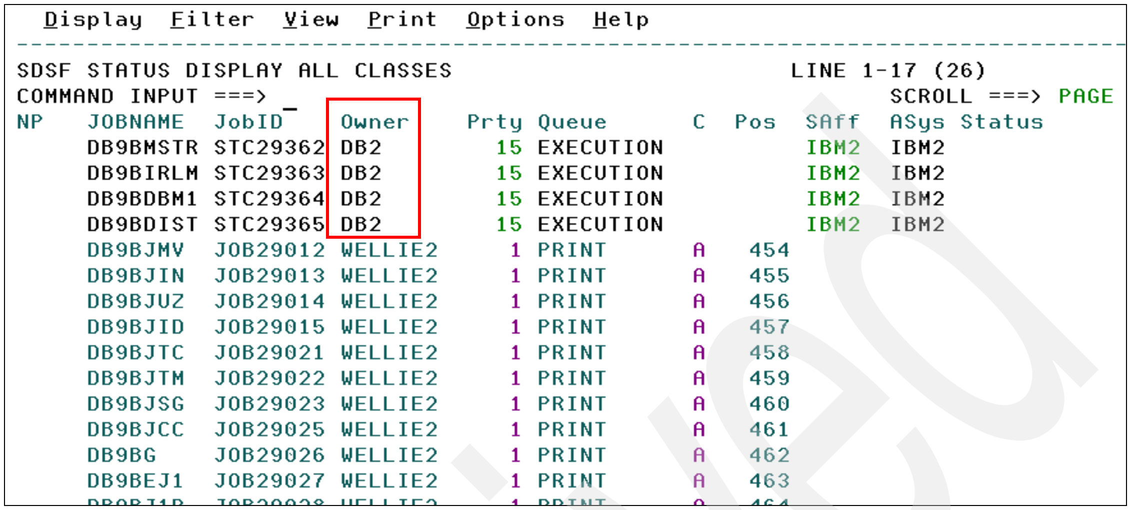
- Make sure the RACF to which you have DB2 configured is configured as a KDC
- Configure the DB2 owner ID to have a Kerberos segment

These two tasks allow DB2 to accept Kerberos tokens and to associate these tokens to the z/OS KDC with which RACF is associated. If you do not complete these configuration tasks, the database program ignores the Kerberos credentials that are provided by the client.

To configure the local z/OS DB2 for Kerberos, complete the following steps:

1. Find the DB2 job owner:
   a. Start a TSO session, and log in to the ISPF program. Choose the option to go to SDSF.
   b. Enter `ST` in the COMMAND INPUT line to see the status of jobs in the system. You can narrow the job list to only the DB2 job by entering `PRE DB9*` in the command input.

c. Find the owner that is listed for the DB2 job names, and note this owner for use in the next step, as shown in Figure 15-2.



```
 Display  Filter  View  Print  Options  Help
-------------------------------------------------------------------------------
SDSF STATUS DISPLAY ALL CLASSES                         LINE 1-17 (26)
COMMAND INPUT ===> _                                           SCROLL ===> PAGE
NP    JOBNAME  JobID    Owner     Prty Queue      C  Pos  SAff  ASys Status
      DB9BMSTR STC29362 DB2         15 EXECUTION          IBM2  IBM2
      DB9BIRLM STC29363 DB2         15 EXECUTION          IBM2  IBM2
      DB9BDBM1 STC29364 DB2         15 EXECUTION          IBM2  IBM2
      DB9BDIST STC29365 DB2         15 EXECUTION          IBM2  IBM2
      DB9BJMV  JOB29012 WELLIE2      1 PRINT      A   454
      DB9BJIN  JOB29013 WELLIE2      1 PRINT      A   455
      DB9BJUZ  JOB29014 WELLIE2      1 PRINT      A   456
      DB9BJID  JOB29015 WELLIE2      1 PRINT      A   457
      DB9BJTC  JOB29021 WELLIE2      1 PRINT      A   458
      DB9BJTM  JOB29022 WELLIE2      1 PRINT      A   459
      DB9BJSG  JOB29023 WELLIE2      1 PRINT      A   460
      DB9BJCC  JOB29025 WELLIE2      1 PRINT      A   461
      DB9BG    JOB29026 WELLIE2      1 PRINT      A   462
      DB9BEJ1  JOB29027 WELLIE2      1 PRINT      A   463
```

Figure 15-2 SDSF DB2 job search

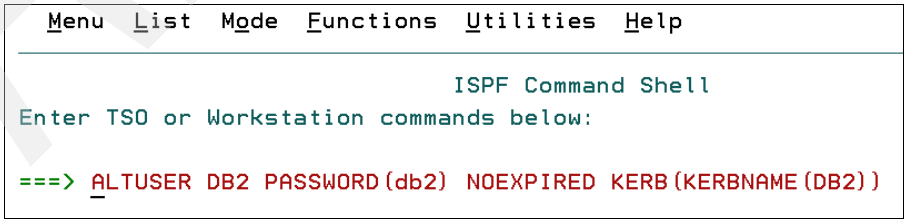
Note: If you do not want to modify the ID that is used by the DB2 instance, you can create a new user ID and modify the STARTED profiles in RACF accordingly.

2. Add a Kerberos segment to the DB2 Job owner ID, using the following command:

   ```
   ALTUSER <ownerID> PASSWORD(<password>) NOEXPIRED
   KERB(KERBNAME(<ownerID>))
   ```

   In this example, the ISPF Command Shell panel was used to enter the following command, as shown in Figure 15-3:

   ```
   ALTUSER DB2 PASSWORD(db2) NOEXPIRED KERB(KERBNAME(DB2))
   ```



```
  Menu  List  Mode  Functions  Utilities  Help
-----------------------------------------------------------------
                              ISPF Command Shell
Enter TSO or Workstation commands below:

===> ALTUSER DB2 PASSWORD(db2) NOEXPIRED KERB(KERBNAME(DB2))
```

Figure 15-3 ISPF Command Shell Kerberos Segment command

Note: The password that is provided in this command is used to generate the Kerberos key. Use the DB2 user ID password for the `PASSWORD` value so that the DB2 password is not changed. If the DB2 user ID does not have a password, then choose any password.

15.3.2 Verifying the DB2 owner ID Kerberos segments

To verify that the Kerberos segment was created successfully, open the “ISPF Command Shell” panel and enter the following command:

```
LISTUSER DB2 KERB NORACF
```

This command lists the Kerberos segment for the owner ID, as shown in Example 15-1.

Example 15-1 Command output for LISTUSER

```
USER=DB2

KERB INFORMATION
----------------
KERBNAME= DB2
KEY FROM= PASSWORD
KEY VERSION= 011
KEY ENCRYPTION TYPE= DES DES3 DESD AES128 AES256

USER=IBMUSER
```

Note: If the command returns the following output, then the RACF ID does not have the Kerberos Segments configured:

```
USER=DB2

NO KERB INFORMATION
```

See 15.3.1, “Configuring the local z/OS DB2 to use Kerberos” on page 397 for information about how to configure the RACF ID’s Kerberos Segment properly.

Note: Be aware of the KEY ENCRYPTION TYPE list that is configured for this RACF ID. This list includes all the encryption types DB2 will accept from users. If you need to add more or remove encryptions types from this user, refer to 2.3.5, “Setting the encryption type” on page 22.

15.4 Configuring WebSphere Application Server

This section provides the steps to configure WebSphere Application Server with Kerberos. You need to modify and verify the global security configuration to work properly with this configuration. Also, you need to modify the Kerberos configuration file to support cross-realm trust.

Note: WebSphere Application Server makes Kerberos RACF service calls using the realm name of the user that is provided to WebSphere Application Server. This scenario uses native mapping for authorization and passes the foreign realm, Microsoft realm, and local z/OS realm with the user. Thus, an extra step is required an to configure the foreign realms on RACF ID.

The following high-level steps are required to configure WebSphere Application server with Kerberos:

1. Configuring the data source
2. Configuring the Kerberos configuration file
3. Verifying global security settings
4. Configuring SPNEGO Web authentication
5. Configuring Kerberos authentication

15.4.1 Configuring the data source

In this step, you create and configure the database reference in WebSphere Application Server to use Kerberos credentials for DB2.

Creating the database for the application

The sample application accesses a database in DB2 z/OS. To create the database, follow these steps:

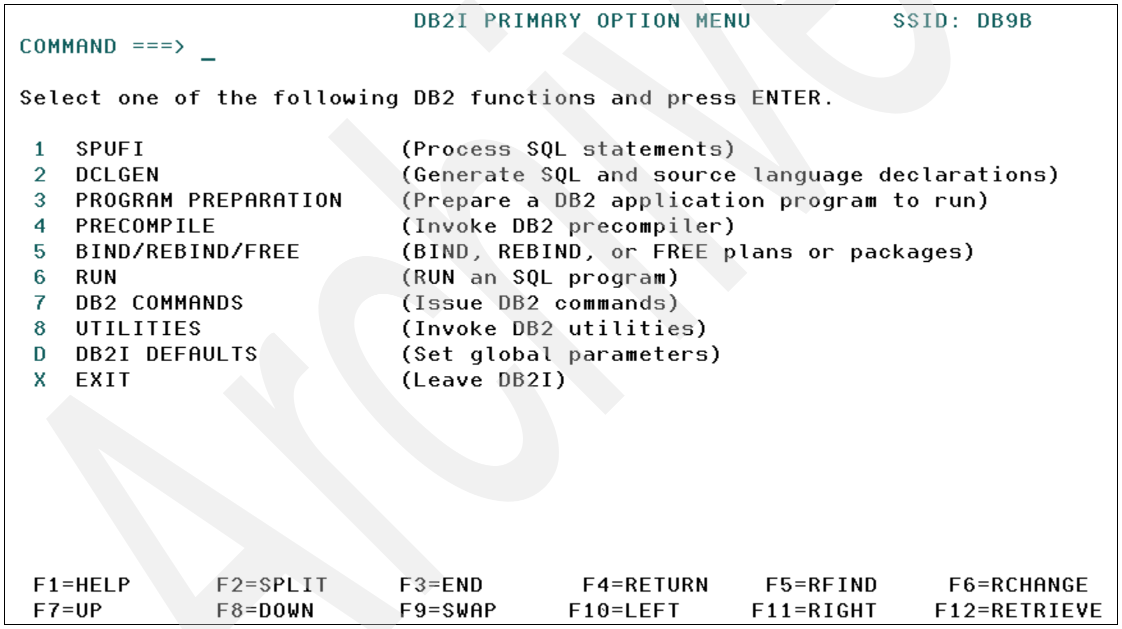
1. Start a TSO session, and log in to the ISPF program.
2. Create a new dataset that contains the SQL statements that are required to build the database, as shown in Example 15-2. In this scenario, the dataset is called `jamesk.spufi.cntl(credb1)`.

Example 15-2 SQL database statements for Kerberos sample

```
--
--
    SET CURRENT SQLID = 'JAMESK';
    DROP   DATABASE SECDB1 ;
--
    COMMIT ;
```

```
--
   CREATE DATABASE SECDB1 STOGROUP SYSDEFLT CCSID EBCDIC;
   CREATE TABLE DEPARTMENT (
     ID INTEGER NOT NULL,
     NAME VARCHAR(50) NOT NULL
    ) IN DATABASE SECDB1;

   ALTER TABLE DEPARTMENT ADD CONSTRAINT DEPARTMENT__UN UNIQUE
    (ID);
   INSERT INTO DEPARTMENT (ID, NAME) VALUES (1, 'IT');
   INSERT INTO DEPARTMENT (ID, NAME) VALUES (2, 'FINANCE');
   INSERT INTO DEPARTMENT (ID, NAME) VALUES (3, 'MARKETING');
   CREATE ALIAS APP.DEPARTMENT FOR JAMESK.DEPARTMENT;
```

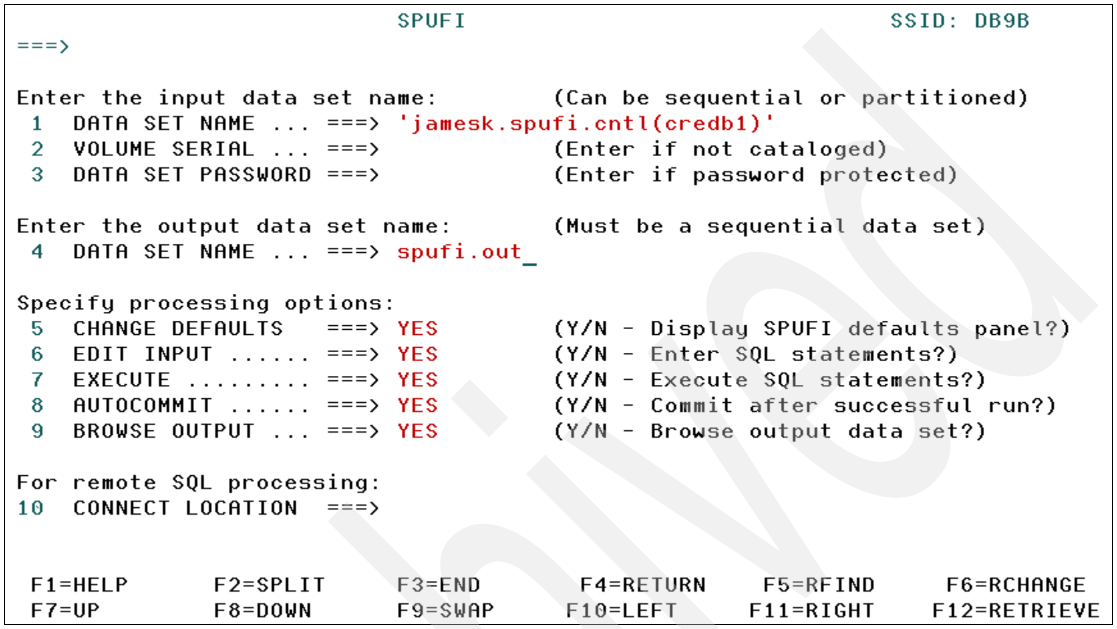
3. Choose the option to open the DB2I PRIMARY OPTION MENU panel, shown in Figure 15-4.



```
                            DB2I PRIMARY OPTION MENU          SSID: DB9B
COMMAND ===> _

Select one of the following DB2 functions and press ENTER.

 1  SPUFI                   (Process SQL statements)
 2  DCLGEN                  (Generate SQL and source language declarations)
 3  PROGRAM PREPARATION     (Prepare a DB2 application program to run)
 4  PRECOMPILE              (Invoke DB2 precompiler)
 5  BIND/REBIND/FREE        (BIND, REBIND, or FREE plans or packages)
 6  RUN                     (RUN an SQL program)
 7  DB2 COMMANDS            (Issue DB2 commands)
 8  UTILITIES               (Invoke DB2 utilities)
 D  DB2I DEFAULTS           (Set global parameters)
 X  EXIT                    (Leave DB2I)








 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 15-4 DB2I PRIMARY OPTION MENU panel

4. Choose the option 1 to open the SPUFI panel. Then, enter the dataset name that contains the SQL statements for the input data set information, and enter the dataset name SPUFI.OUT for the output data set information, as shown in Figure 15-5.



```
                           SPUFI                               SSID: DB9B
===>

Enter the input data set name:        (Can be sequential or partitioned)
 1  DATA SET NAME ... ===> 'jamesk.spufi.cntl(credb1)'
 2  VOLUME SERIAL ... ===>            (Enter if not cataloged)
 3  DATA SET PASSWORD ===>            (Enter if password protected)

Enter the output data set name:       (Must be a sequential data set)
 4  DATA SET NAME ... ===> spufi.out_

Specify processing options:
 5  CHANGE DEFAULTS   ===> YES        (Y/N - Display SPUFI defaults panel?)
 6  EDIT INPUT ...... ===> YES        (Y/N - Enter SQL statements?)
 7  EXECUTE ......... ===> YES        (Y/N - Execute SQL statements?)
 8  AUTOCOMMIT ...... ===> YES        (Y/N - Commit after successful run?)
 9  BROWSE OUTPUT ... ===> YES        (Y/N - Browse output data set?)

For remote SQL processing:
10  CONNECT LOCATION  ===>


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Figure 15-5 SPUFI panel

5. Press ENTER twice to get an EDIT session that displays the SQL statements in the dataset created in Example 15-2 on page 400.
6. Press F3, and then press ENTER to execute the SQL statements.

   For each line of SQL you should see this message:

   ```
   DSNE616I STATEMENT EXECUTION WAS SUCCESSFUL, SQLCODE IS 0
   ```

Now, the database schema JAMESK and table DEPARMENT are created and ready for use. This process also created an alias mapping of APP.DEPARTMENT to JAMESK.DEPARTMENT so that the application SQL statements work.

Creating the J2C authentication data

First, create J2C authentication data that can be used to verify that the database is defined correctly to WebSphere Application Server.

Note: This is not the login data that is used for the Kerberos connection.

To create this data, follow these steps:

1. Log on to the WebSphere Application Server administrative console, and navigate to **Security** → **Global security**.
2. Navigate to **Authentication** → **Java Authentication and Authorization Service**, and select **J2C authentication data** in the global security panel.
3. Click **New**.
4. Enter the alias name for the new J2C authentication data, the user ID, and password information that are required to access the database.

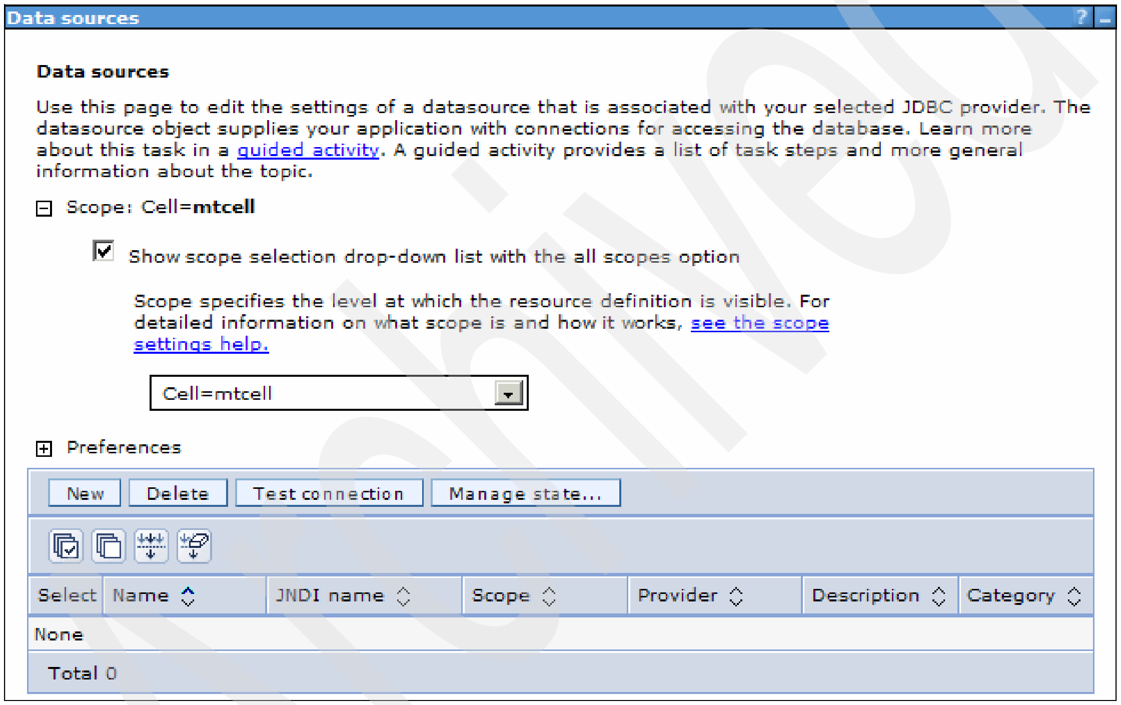
   The new alias is called *DB2*. When this alias is referenced from resource configurations, it is prefaced with the node name for the cell. In this case, the new J2C authentication data is referenced to as *mtdmnode/DB2*.
5. Click **OK**.

Creating the data source

Creating the database: You can find information about creating the database in "Creating the database for the application" on page 400. You can create the database now, or you can wait until you are ready to validate the scenario.

Next, create the data source that defines the connection to the database to WebSphere Application Server as follows:

1. Navigate to **Resources** → **JDBC** → **Data sources** in the left menu. Then, select the appropriate scope, as shown in Figure 15-6. Click **New**.



Data sources

Data sources

Use this page to edit the settings of a datasource that is associated with your selected JDBC provider. The datasource object supplies your application with connections for accessing the database. Learn more about this task in a guided activity. A guided activity provides a list of task steps and more general information about the topic.

Scope: Cell=**mtcell**

[✓] Show scope selection drop-down list with the all scopes option

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.

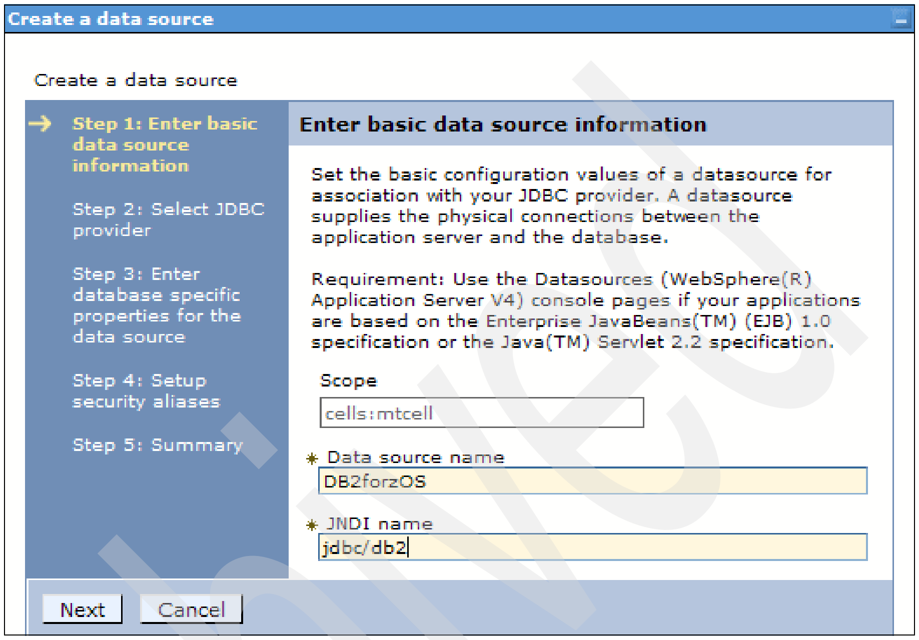
Cell=mtcell

Preferences

Select	Name	JNDI name	Scope	Provider	Description	Category
None						
Total 0						

Figure 15-6 Scoping the data source panel

2. Enter the basic data source information, including the data source name for this configuration and the JNDI name to be used to access it, as shown in Figure 15-7. Click **Next**.



Create a data source

Create a data source

→ Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary

Enter basic data source information

Set the basic configuration values of a datasource for association with your JDBC provider. A datasource supplies the physical connections between the application server and the database.

Requirement: Use the Datasources (WebSphere(R) Application Server V4) console pages if your applications are based on the Enterprise JavaBeans(TM) (EJB) 1.0 specification or the Java(TM) Servlet 2.2 specification.

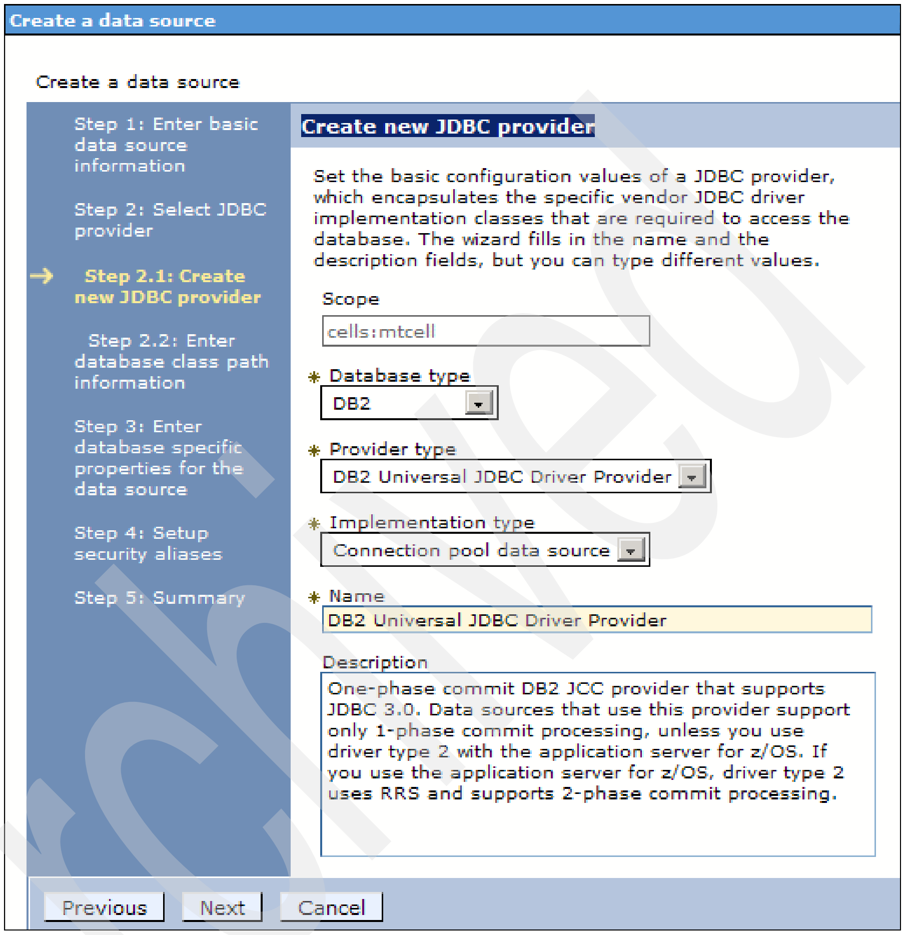
Scope: cells:mtcell

* Data source name: DB2forzOS

* JNDI name: jdbc/db2

Figure 15-7 Enter basic data source information Wizard panel

3. Select an existing JDBC provider or enter the information for a new one, as shown in Figure 15-8. Click **Next**.


Create a data source

Create a data source

Step 1: Enter basic data source information

Step 2: Select JDBC provider

→ Step 2.1: Create new JDBC provider

Step 2.2: Enter database class path information

Step 3: Enter database specific properties for the data source

Step 4: Setup security aliases

Step 5: Summary

Create new JDBC provider

Set the basic configuration values of a JDBC provider, which encapsulates the specific vendor JDBC driver implementation classes that are required to access the database. The wizard fills in the name and the description fields, but you can type different values.

Scope
cells:mtcell

* Database type
DB2

* Provider type
DB2 Universal JDBC Driver Provider

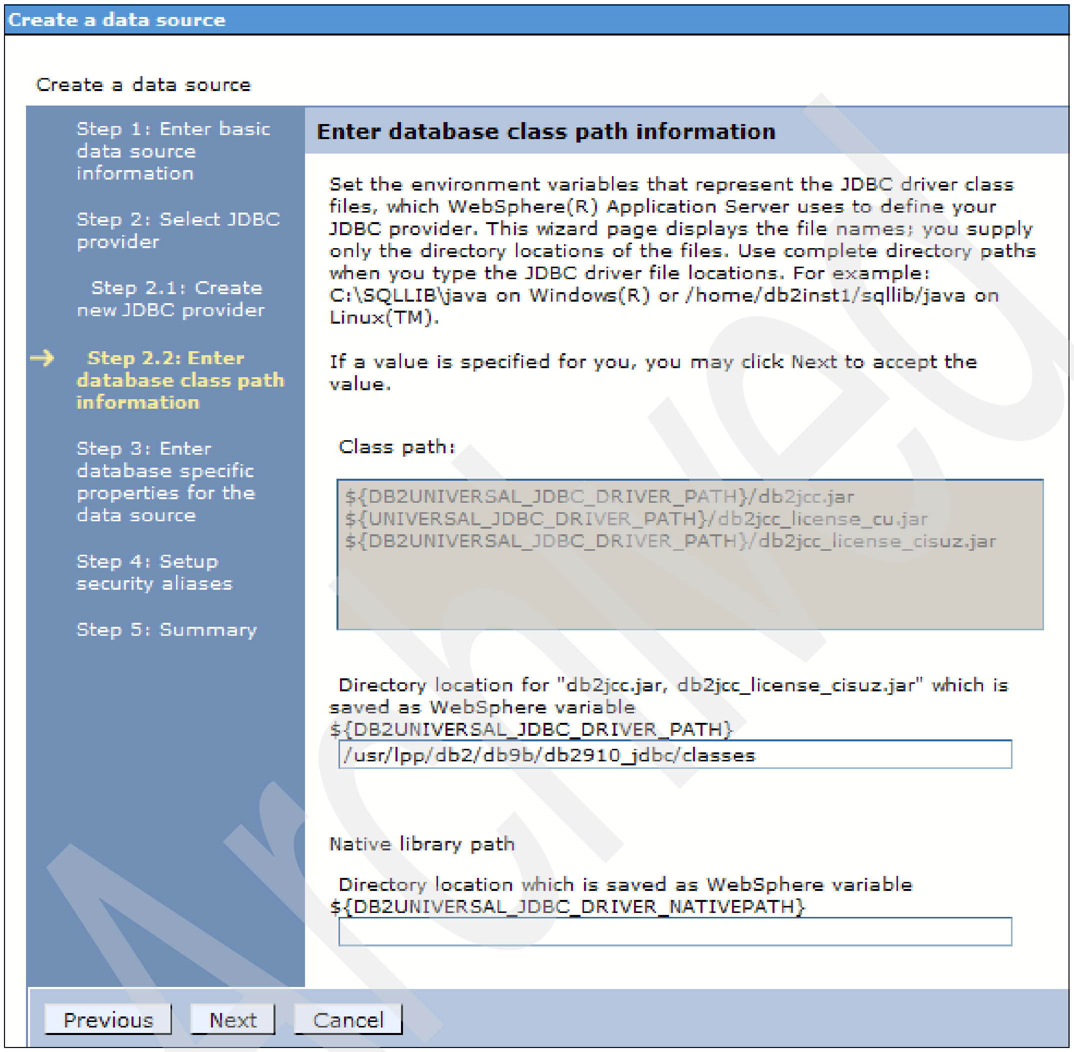
* Implementation type
Connection pool data source

* Name
DB2 Universal JDBC Driver Provider

Description
One-phase commit DB2 JCC provider that supports JDBC 3.0. Data sources that use this provider support only 1-phase commit processing, unless you use driver type 2 with the application server for z/OS. If you use the application server for z/OS, driver type 2 uses RRS and supports 2-phase commit processing.

Figure 15-8 Create new JDBC provider

4. Enter the class path information for the database drivers (Figure 15-9), and click **Next**.


Create a data source
Create a data source
Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 2.1: Create new JDBC provider
Step 2.2: Enter database class path information
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Previous
Next
Cancel

Enter database class path information

Set the environment variables that represent the JDBC driver class files, which WebSphere(R) Application Server uses to define your JDBC provider. This wizard page displays the file names; you supply only the directory locations of the files. Use complete directory paths when you type the JDBC driver file locations. For example: C:\SQLLIB\java on Windows(R) or /home/db2inst1/sqllib/java on Linux(TM).

If a value is specified for you, you may click Next to accept the value.

Class path:

```
${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc.jar
${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar
${DB2UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cisuz.jar
```

Directory location for "db2jcc.jar, db2jcc_license_cisuz.jar" which is saved as WebSphere variable ${DB2UNIVERSAL_JDBC_DRIVER_PATH}

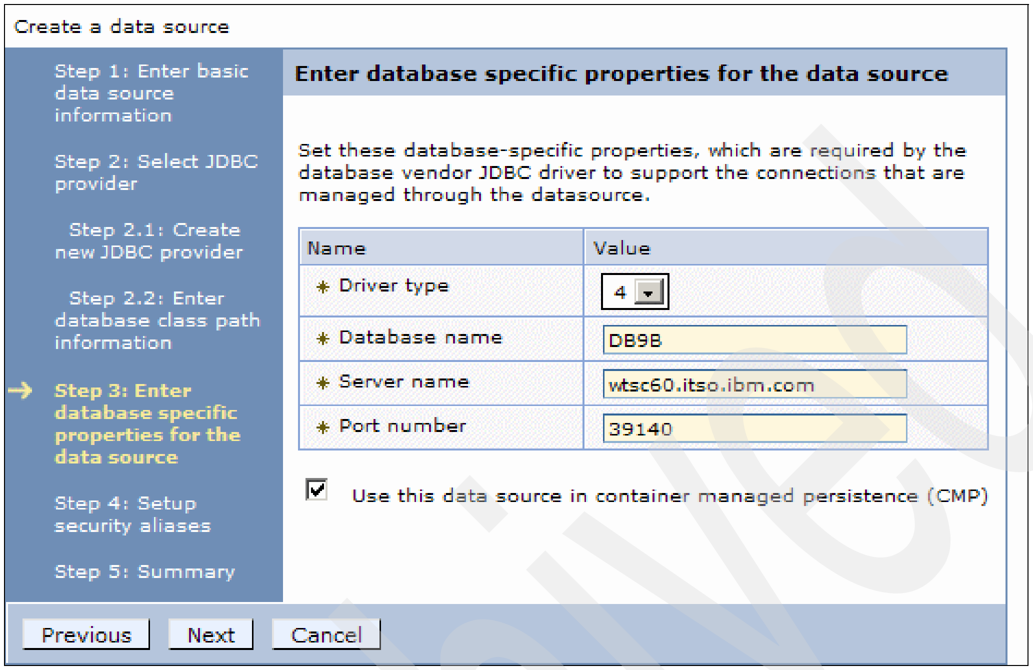
/usr/lpp/db2/db9b/db2910_jdbc/classes

Native library path

Directory location which is saved as WebSphere variable ${DB2UNIVERSAL_JDBC_DRIVER_NATIVEPATH}

Figure 15-9 Enter database class path information

5. Enter the database properties as shown in Figure 15-10, and click **Next**.



Create a data source

Step 1: Enter basic data source information

Step 2: Select JDBC provider

Step 2.1: Create new JDBC provider

Step 2.2: Enter database class path information

→ **Step 3: Enter database specific properties for the data source**

Step 4: Setup security aliases

Step 5: Summary

Enter database specific properties for the data source

Set these database-specific properties, which are required by the database vendor JDBC driver to support the connections that are managed through the datasource.

Name	Value
* Driver type	4
* Database name	DB9B
* Server name	wtsc60.itso.ibm.com
* Port number	39140

[✓] Use this data source in container managed persistence (CMP)

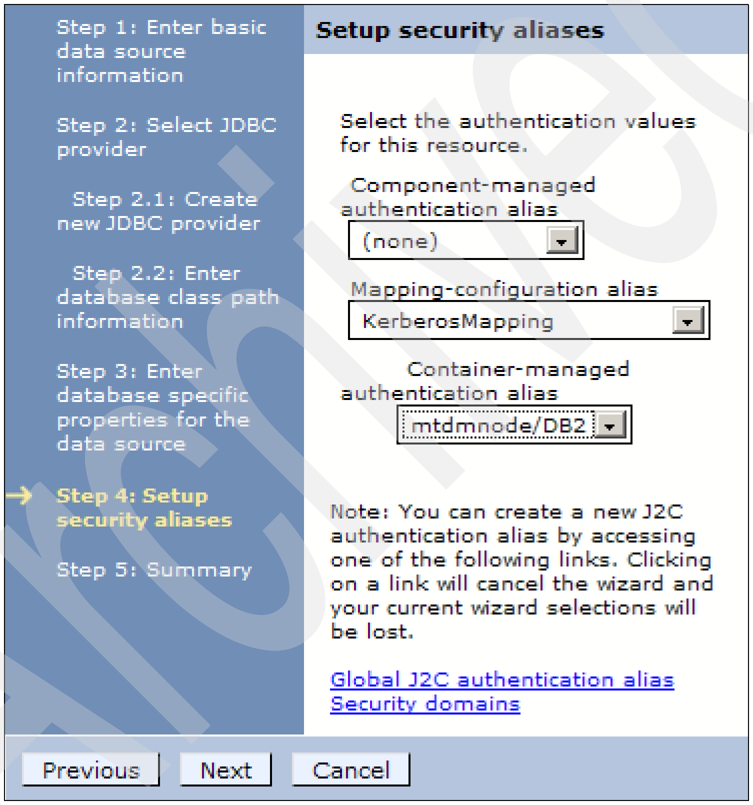
Figure 15-10 Enter database specific properties for the data source

6. Enter the authentication information for the database, as shown in Figure 15-11.

   For this scenario, the sample application uses container-managed authentication. The Mapping-configuration alias field is set to KerberosMapping.

   The container-managed authentication alias field is also set to point to the J2C authentication data defined in "Creating the J2C authentication data" on page 402. This allows you to test the database connection from the administrative console.

   Click **Next**.



Setup security aliases

Select the authentication values for this resource.

Component-managed authentication alias: (none)

Mapping-configuration alias: KerberosMapping

Container-managed authentication alias: mtdmnode/DB2

Note: You can create a new J2C authentication alias by accessing one of the following links. Clicking on a link will cancel the wizard and your current wizard selections will be lost.

Global J2C authentication alias
Security domains

Step 1: Enter basic data source information
Step 2: Select JDBC provider
Step 2.1: Create new JDBC provider
Step 2.2: Enter database class path information
Step 3: Enter database specific properties for the data source
Step 4: Setup security aliases
Step 5: Summary
Previous
Next
Cancel

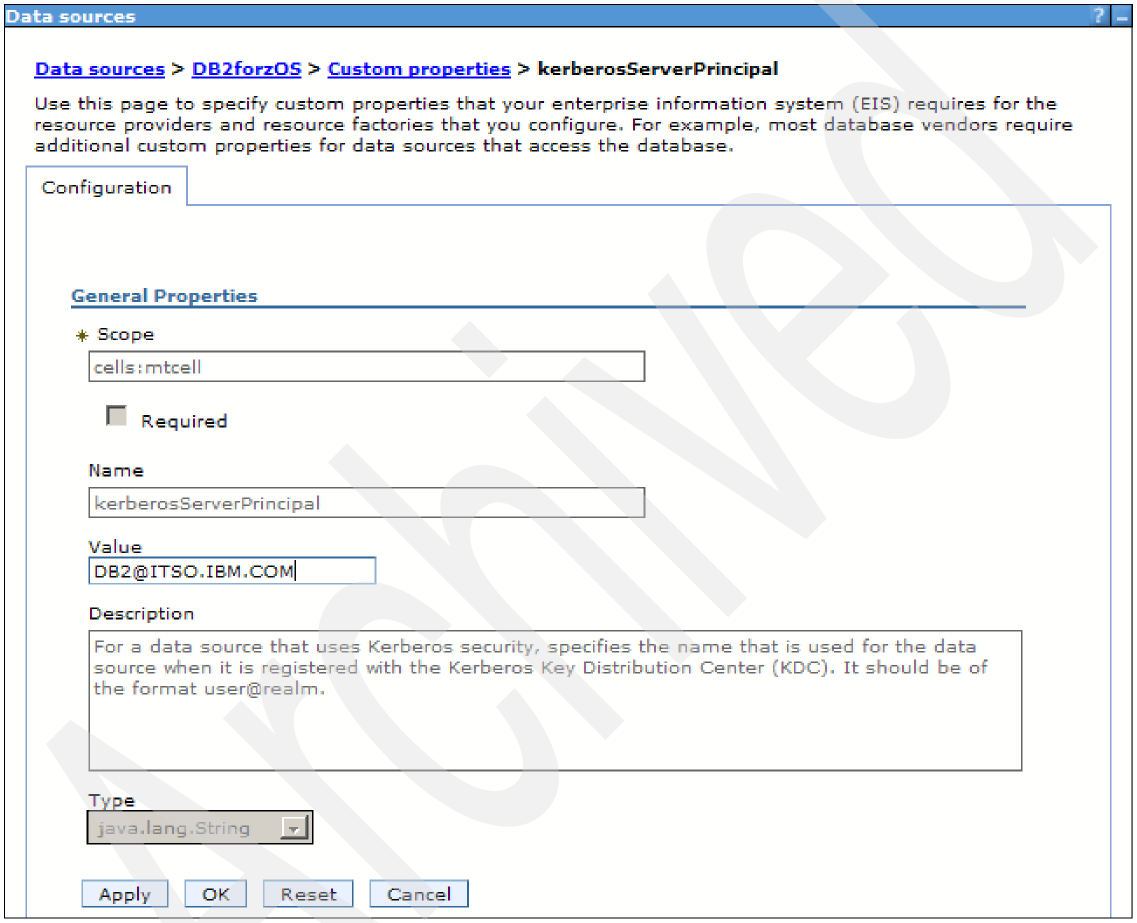
Figure 15-11 Setup security aliases

7. On the Summary wizard panel, click **Finish**.
8. Now, navigate to **Resources** → **JDBC** → **Data sources** in the left menu. Select the new **DB2forzOS** data source from the list.

9. Select **Custom properties** from the Additional Properties section. Then, select the **kerberosServerPrincipal** property, and enter `DB2@ITSO.IBM.COM` in the value field (Figure 15-12), for example:

   `<db2 owner ID>@<z/OS Kerberos REalm>`

   Click **OK**.


Data sources
Data sources > DB2forzOS > Custom properties > kerberosServerPrincipal
Configuration
General Properties
Apply
OK
Reset
Cancel

Use this page to specify custom properties that your enterprise information system (EIS) requires for the resource providers and resource factories that you configure. For example, most database vendors require additional custom properties for data sources that access the database.

* Scope: cells:mtcell

Required

Name: kerberosServerPrincipal

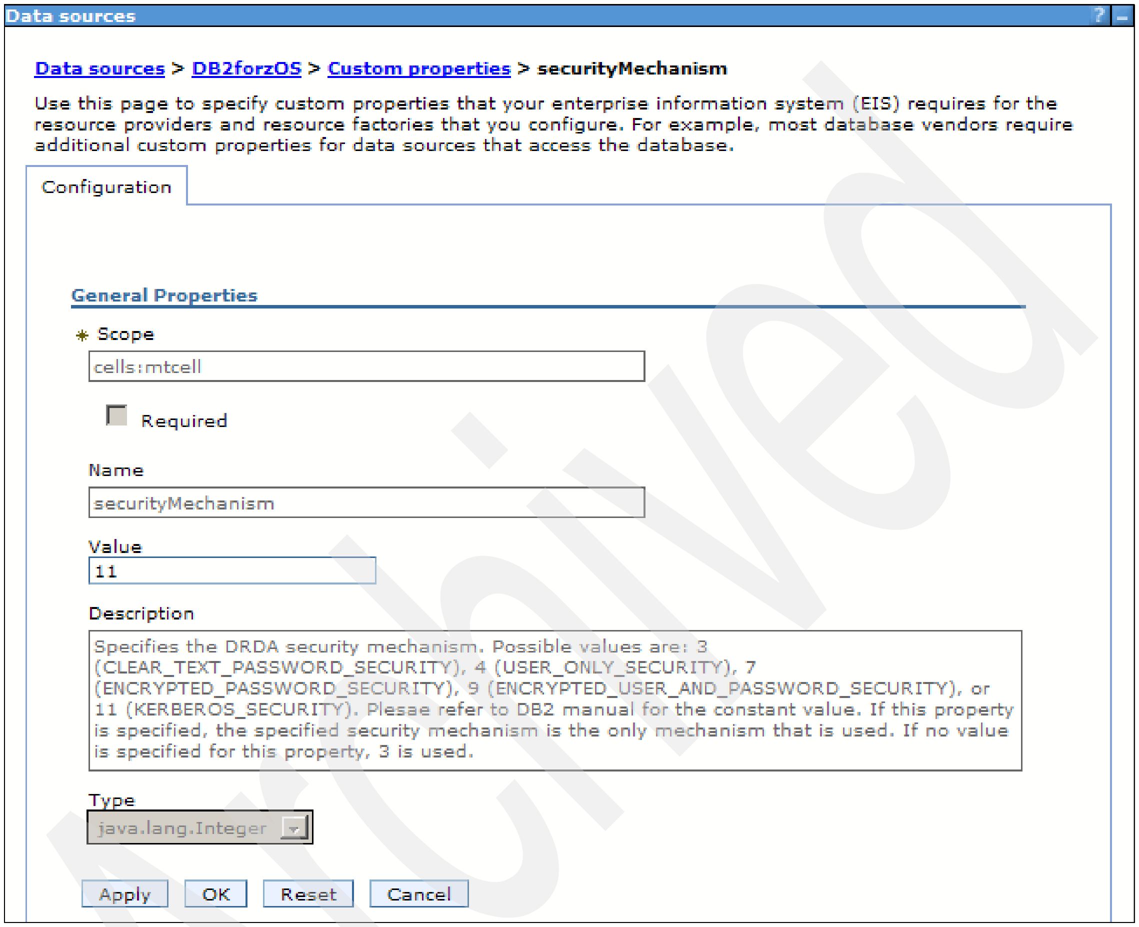
Value: DB2@ITSO.IBM.COM

Description: For a data source that uses Kerberos security, specifies the name that is used for the data source when it is registered with the Kerberos Key Distribution Center (KDC). It should be of the format user@realm.

Type: java.lang.String

Figure 15-12 kerberosServerPrincipal property panel

10. Select **securityMechanism** property and enter 11 in the value field. This value indicates Kerberos security (the values can be seen in the Description field in Figure 15-13). Click **OK**.


Data sources
Data sources > DB2forzOS > Custom properties > securityMechanism

Use this page to specify custom properties that your enterprise information system (EIS) requires for the resource providers and resource factories that you configure. For example, most database vendors require additional custom properties for data sources that access the database.

Configuration
General Properties

* Scope: cells:mtcell
- Required: ☐
- Name: securityMechanism
- Value: 11
- Description: Specifies the DRDA security mechanism. Possible values are: 3 (CLEAR_TEXT_PASSWORD_SECURITY), 4 (USER_ONLY_SECURITY), 7 (ENCRYPTED_PASSWORD_SECURITY), 9 (ENCRYPTED_USER_AND_PASSWORD_SECURITY), or 11 (KERBEROS_SECURITY). Please refer to DB2 manual for the constant value. If this property is specified, the specified security mechanism is the only mechanism that is used. If no value is specified for this property, 3 is used.
- Type: java.lang.Integer

Figure 15-13 securityMechanism property panel

11. Save and synchronize configuration changes to the nodes.

Now, the data source is configured to connect to the database using Kerberos credentials for any application to use.

Verifying the connection to DB2

Use test connection to verify that the new data source is working.

Note: this verification simply ensures that you have configured the data source correctly. It uses basic authentication, not Kerberos authentication.

To verify that the connection is working, follow these steps:

1. Navigate to **Resources** → **JDBC** → **Data sources** in the left menu.
2. Select the **DB2forzOS** data source, and click **Test connection**.

If the connection is successful, the following message displays:

```
The test connection operation for data source DB2forzOS on server
dmgr at node mtdmnode was successful.
```

15.4.2 Configuring the Kerberos configuration file

You need to create a new Kerberos configuration file for WebSphere Application Server. This file must contain the information for the z/OS KDC as well as the Microsoft Active Directory.

For this scenario, we copy the Kerberos configuration file from the z/OS KDC (because these are on the same system). Then, we modify the file for cross-realm trust and generate a keytab file. Follow these steps to configure the Kerberos file:

1. Copy the `krb5.conf` file from the z/OS KDC, shown in Example 2-1 on page 17, to the `/etc/krb5` directory for WebSphere Application Server.
2. Modify the `krb5.conf` file to add the cross-realm trust between the z/OS KDC and Microsoft Kerberos KDC. Add the Microsoft Kerberos KDC information and `saw921-sys8.itso.ral.ibm.com` system information to the `realm` section of the Kerberos configuration file, `krb5.conf`, as shown in Example 15-3.

Example 15-3 Microsoft Kerberos KDC information in the krb5.conf file

```
[realms]
ITSO.IBM.COM = {
    kdc = wtsc60.itso.ibm.com:88
    kpasswd_server = wtsc60.itso.ibm.com:464
    default_domain = itso.ibm.com
}
KDC2008.ITSO.COM = {
    kdc = saw921-sys8.itso.ral.ibm.com:88
    kpasswd_server = saw921-sys8.itso.ral.ibm.com:464
    default_domain = kdc2008.itso.com
}
```

3. Generate the keytab file for both the SPNEGO WebSphere Application Server SPN and Kerberos WebSphere Application Server service principal name (SPN) using the **ktab** utility provided by Java as follows:

   a. Create the SPNEGO WebSphere Application Server SPN with the following command:

   ```
   ktab -a HTTP/wtsc60.itso.ibm.com@ITSO.IBM.COM
   ```

   Note: Modify the Microsoft user account that is mapped to the SPNEGO SPN. Set `Delegation` to `Trust` this user for delegation to any server (Kerberos only).

   b. Create the Kerberos WebSphere Application Server SPN with the following command:

   ```
   ktab -a WAS101/wtsc60.itso.ibm.com@ITSO.IBM.COM
   ```

   Note: The initial component for the Kerberos WebSphere Application Server SPN, `WAS101`, can be anything, unlike SPNEGO, which requires the initial component to be `HTTP`. After a component name is chosen, WebSphere Application Server Kerberos configuration needs to know this information. Refer to 15.4.5, "Configuring Kerberos authentication" on page 418.

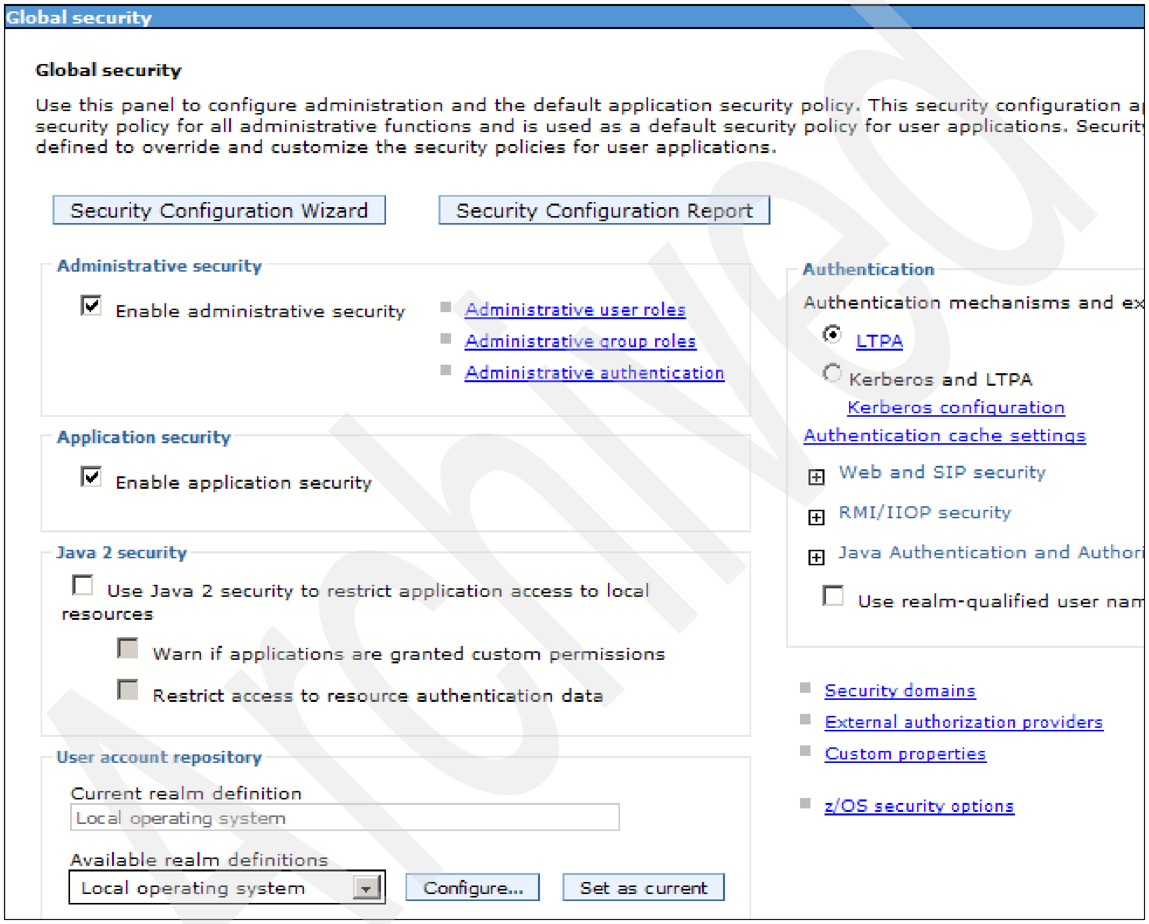
Now, the Kerberos configuration and keytab files are ready for WebSphere Application Server to interact with the cross-realm trust.

15.4.3 Verifying global security settings

For this scenario, we assume that when the profile was created the local OS (RACF) plus SAF authorization was selected. However, these settings do not provide all the settings that this scenario needs. The following steps might have already been done in your environment. To ensure that global security settings are configured correctly, follow these steps:

1. Log on to the WebSphere Application Server administrative console and navigate to **Security** → **Global security**.
2. In the Web and SIP security section, click the link for **Single sign-on (SSO)**. Ensure that **Enabled** is selected and that the Domain name is set. In this scenario, the Domain name field is set to `itso.ibm.com` value. Return to the Global security panel.

3. Click **External authorization providers**. Select **System Authorization Facility (SAF) authorization**, and click **OK**.
4. Verify that you have selected **Enabled** for both administrative and application security, but disable Use Java 2 security to restrict application access to local resources (as shown in Figure 15-14 on page 414).
5. Click **Apply** to apply all the settings in the global security panel.


Global security

Global security

Use this panel to configure administration and the default application security policy. This security configuration a
security policy for all administrative functions and is used as a default security policy for user applications. Securit
defined to override and customize the security policies for user applications.

Administrative security

[✓] Enable administrative security

Administrative user roles
Administrative group roles
Administrative authentication

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: Local operating system

Available realm definitions: Local operating system

Authentication

Authentication mechanisms and ex

● LTPA

○ Kerberos and LTPA

Kerberos configuration
Authentication cache settings
Web and SIP security
RMI/IIOP security
Java Authentication and Authori

[ ] Use realm-qualified user nam

Figure 15-14 Global security panel before Kerberos

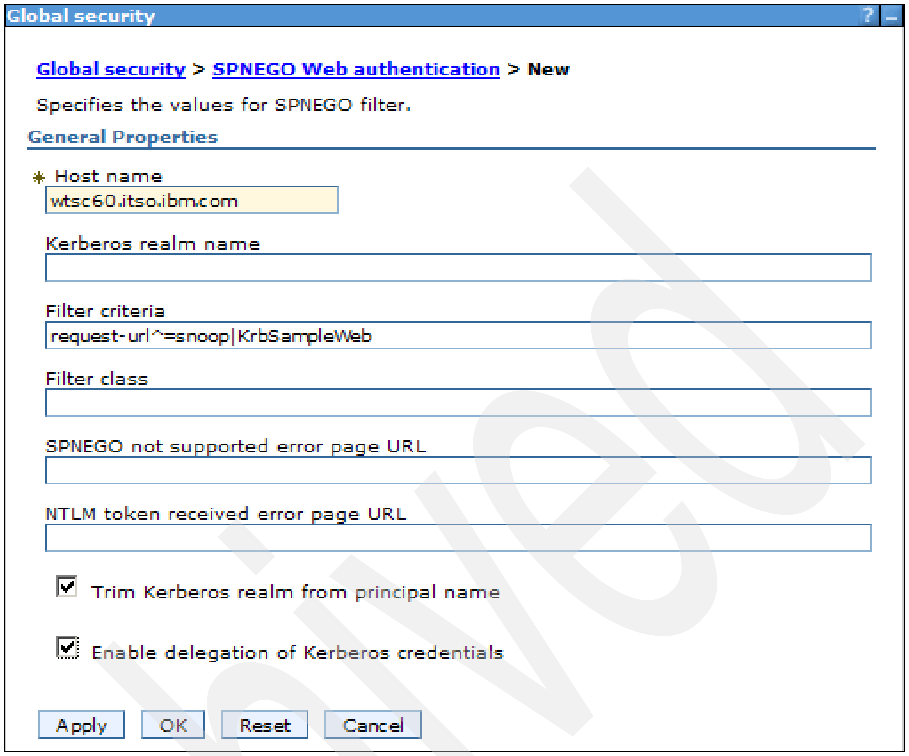
6. Save the configuration and restart WebSphere Application Server V7. Because this example changed the user registry in the global security configuration, the next time you log on to WebSphere, you need to use one of the IDs defined in step 5.

15.4.4 Configuring SPNEGO Web authentication

This section explains how to configure the SPNEGO Web authentication in WebSphere Application Server to allow clients to pass SPNEGO tokens that contain Kerberos credentials to authenticate with WebSphere Application Servers. Because this scenario configures a cross-realm trust between the Microsoft Kerberos realm and z/OS Kerberos realm, Microsoft client systems for the Microsoft Kerberos KDC realm can provide SPNEGO tokens to SPNEGO services that are configured and hosted in WebSphere Application Server.

These steps configure the KrbSampleWeb application (refer to 15.6, “Validating the scenario” on page 420) and the snoop application (the sample application that is provide by WebSphere Application Server). These sample applications are used to test the SPNEGO Web authentication. Follow these steps:

1. Log on to the WebSphere Application Server administrative console, and navigate to **Security** → **Global security**.
2. Expand **Web and SIP security**, and then click **SPNEGO Web authentication.**
3. Each application that will participate in SSO must have a filter defined. On the SPNEGO Filters list click **New**. Then, follow these steps, as shown in Figure 15-15 on page 416:
    a. Enter the fully qualified host name of WebSphere server in the Host name field, and enter the realm name in the Kerberos realm name field.
    b. Enter `request-url^=snoop|KrbSampleWeb` in the Filter criteria field.
    c. Select **Trim Kerberos realm from principal name** to remove the suffix of the principal user name.
    d. Select **Enable delegation of Kerberos credentials** to allow client credentials to flow to other processes.
    e. Click **Apply**. The realm name and filter criteria are validated. Return to the SPNEGO Web authentication page.


Global security

Global security > SPNEGO Web authentication > New

Specifies the values for SPNEGO filter.

General Properties

* Host name
wtsc60.itso.ibm.com

Kerberos realm name

Filter criteria
request-url^=snoop|KrbSampleWeb

Filter class

SPNEGO not supported error page URL

NTLM token received error page URL

[✓] Trim Kerberos realm from principal name

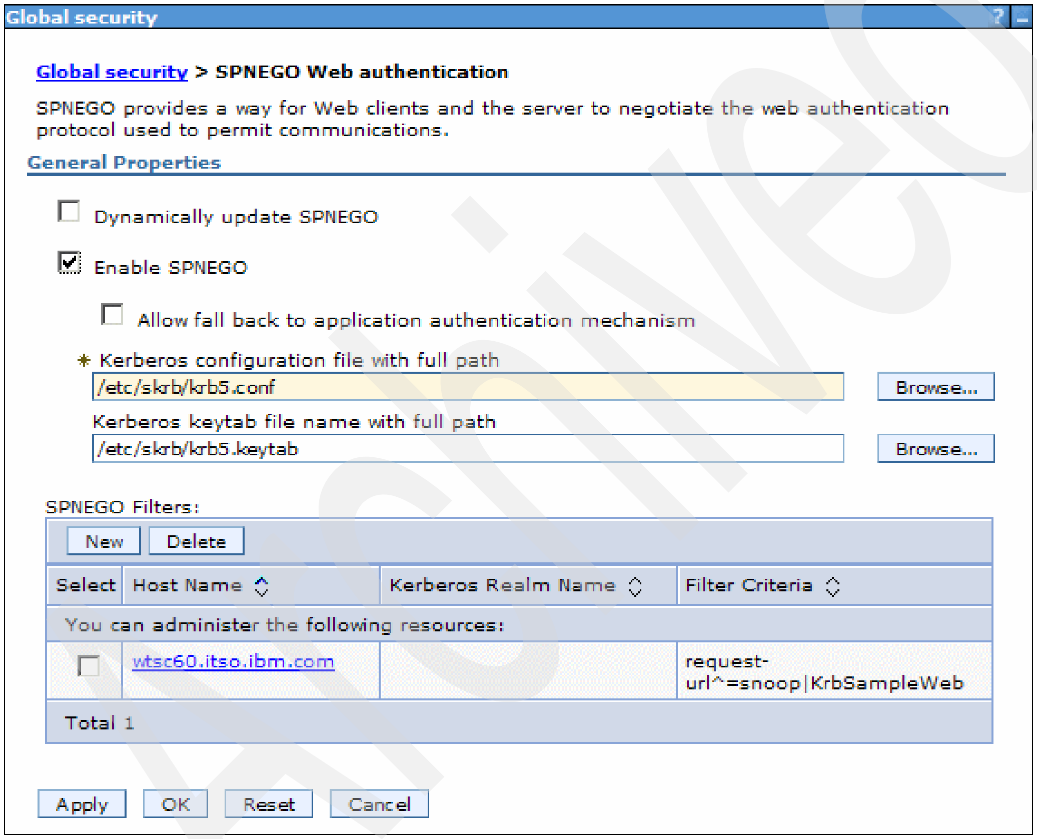
[✓] Enable delegation of Kerberos credentials

Figure 15-15 SPNEGO Web authentication filter

4. Select **Enable SPNEGO**, and enter the Kerberos configuration and keytab file paths, as shown in Figure 15-16. Then, click **Apply**.

> **Note:** The “Allow fall back to application authentication mechanism” option specifies that SPNEGO is used to log in to WebSphere Application Server first. If SPNEGO authentication fails, the authentication mechanism that is defined during application assembly time is used.

Click **OK**.



SPNEGO provides a way for Web clients and the server to negotiate the web authentication protocol used to permit communications.

General Properties

☐ Dynamically update SPNEGO

☑ Enable SPNEGO

☐ Allow fall back to application authentication mechanism

* Kerberos configuration file with full path
/etc/skrb/krb5.conf

Kerberos keytab file name with full path
/etc/skrb/krb5.keytab

SPNEGO Filters:

Select	Host Name	Kerberos Realm Name	Filter Criteria
You can administer the following resources:			
☐	wtsc60.itso.ibm.com		request-url^=snoop|KrbSampleWeb
Total 1			

Figure 15-16 Enable SPNEGO Web authentication window

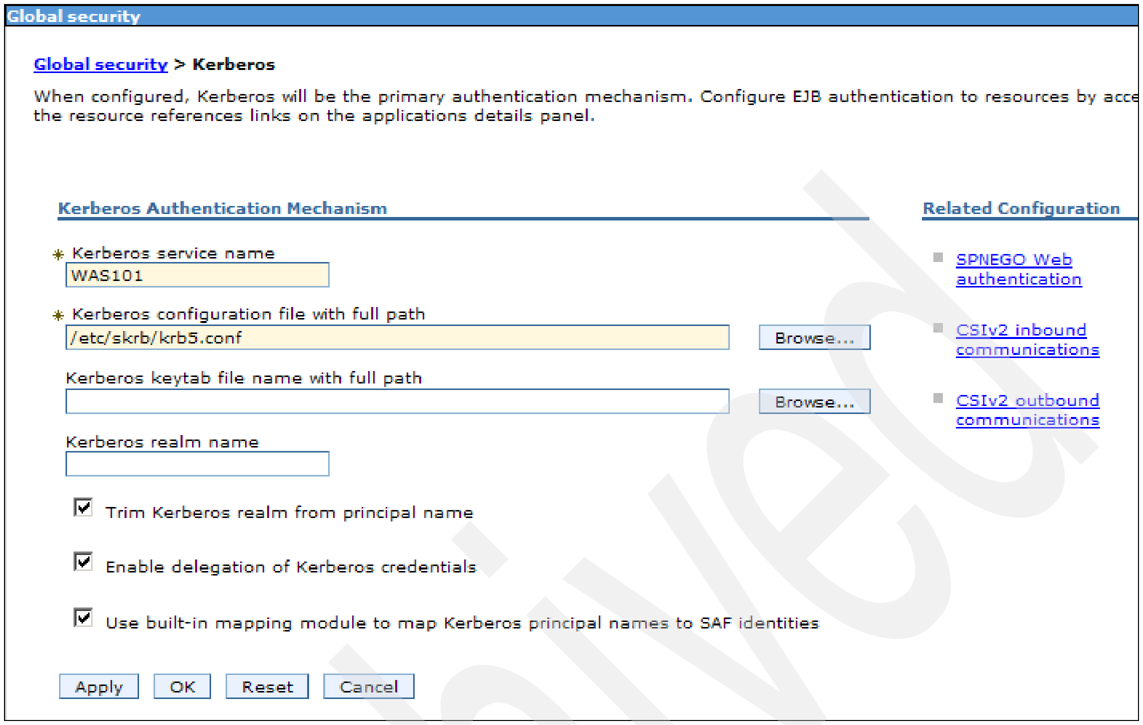
5. Save and synchronize configuration changes to the nodes.
6. The cell or affected server must be recycled to load the SPNEGO modifications to global security. However, because more configuration will be done in this case, the recycle can wait until everything is configured.

Note: If the "Dynamically update SPNEGO" option is enabled (see Figure 15-16), then after the initial recycle of the server to enable SPNEGO, future changes to the SPNEGO configuration load automatically without a server recycle.

15.4.5 Configuring Kerberos authentication

This step configures WebSphere Application Server for Kerberos authentication and enables WebSphere to pass Kerberos tokens to DB2. Follow these steps:

1. Log on to the WebSphere Application Server administrative console and select **Security** → **Global security**.
2. Select the **Kerberos configuration** link in the Authentication section. Then, follow these steps, as shown in Figure 15-17 on page 419:
   a. Enter `WAS101` in the Kerberos service name field. This name is the Kerberos WebSphere Application Server SPN initial component name (see 15.4.2, "Configuring the Kerberos configuration file" on page 412).
   b. Enter the location of the Kerberos configuration file, `/etc/skrb/krb5.conf`, in the Kerberos configuration file with full path field.
   c. Select **Trim Kerberos realm from principal name** to remove the suffix of the principal user name.
   d. Select **Enable delegation of Kerberos credentials** to allow the client credentials to flow to other processes.
   e. Select **Use built-in mapping module to map Kerberos principal names to SAF identities** to map a Kerberos principal name to a SAF identity on z/OS.
   f. Click **OK** to return to the Global security page and to automatically change the Authentication radio button from LTPA to Kerberos and LTPA.


Global security

Global security > Kerberos

When configured, Kerberos will be the primary authentication mechanism. Configure EJB authentication to resources by acce
the resource references links on the applications details panel.

Kerberos Authentication Mechanism

* Kerberos service name
WAS101

* Kerberos configuration file with full path
/etc/skrb/krb5.conf

Kerberos keytab file name with full path

Kerberos realm name

[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

[✓] Use built-in mapping module to map Kerberos principal names to SAF identities

Browse...
Browse...
Apply
OK
Reset
Cancel
Related Configuration
SPNEGO Web authentication
CSIv2 inbound communications
CSIv2 outbound communications

Figure 15-17 Kerberos configuration panel

3. Click **Apply**.
4. Save and synchronize the configuration changes to the nodes.
5. Recycle the cell to load the Kerberos authentication settings to global security.

15.5 Configuring SPNGEO on the client browser

The Web browser is a common client tool used in SSO scenarios, and we use it in this case to verify the SPNEGO configuration.

First, ensure that the client system is a member of the Microsoft Kerberos KDC realm. Then, configure the Web browser on the client workstation for SPNEGO as described in 6.3, “Configuring the Microsoft Windows client” on page 141. The scenario can use either the Microsoft Internet Explorer or Mozilla Firefox to test SPNEGO authentication with WebSphere Application Server.

When the browser accesses the Web page, it passes the client's Kerberos credentials automatically, and the returned page uses the Kerberos credentials to access the DB2 database.

15.6 Validating the scenario

This scenario uses a sample Kerberos application to test and verify the configuration. The sample application provides a simple protected servlet that contains a link to pull information from the database using an EJB with a resource reference.

The application verifies the client user credentials that are passed to the database to query the information using Kerberos token.

15.6.1 Installing and configuring the sample application

Installing the sample application requires executing steps in the WebSphere Application Server administrative console and TSO commands. The enterprise application installation is done in the administrative console, and then security mappings are executed using TSO commands because WebSphere application Server has SAF Authorization configured. Thus, all authorization is done using the z/OS SAF configuration, not the JEE security role mapping in WebSphere Application Server.

Modifying the resource adapter

Removing the alias from the data source forces the database connection to use only the client Kerberos credentials. To modify the resource adapter, follow these steps:

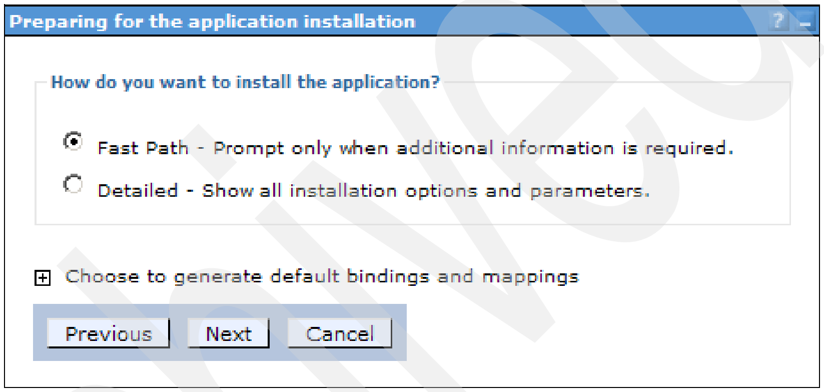
1. Navigate to **Resources** → **JDBC** → **Data sources** in the left menu.
2. Select the **DB2forzOS** data source from the list.
3. Change the Container-managed authentication alias to **(None)**.
4. Click **Apply**.
5. Save and synchronize configuration changes to the nodes.

Now, any successful access to the database proves that the user's Kerberos credentials are passed to the database for authentication.

Installing the application

To install the sample application:

1. Log on to the WebSphere Application Server administrative console, and navigate to **Applications** → **WebSphere enterprise applications**.
2. Click **Install** to install the sample enterprise application. Then, follow these steps:
   a. In the Path to the new application panel, click **Browse** to enter the location of the `KrbSample.ear` file, and click **Next**.
   b. Select the “Fast Path - Prompt only when additional information is required” option (shown in Figure 15-18), and click **Next**.



Preparing for the application installation

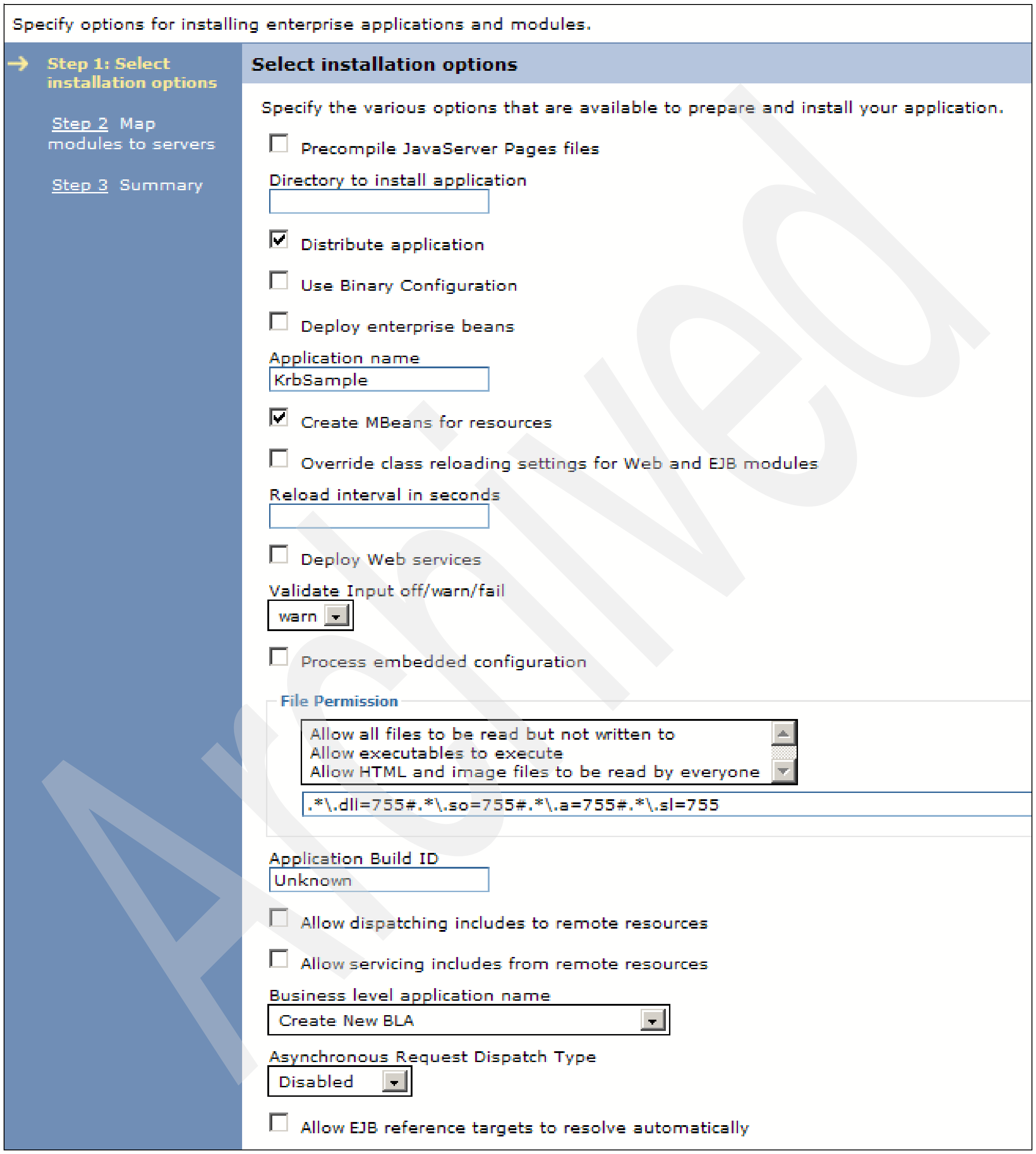
How do you want to install the application?

[✓ Fast Path - Prompt only when additional information is required.]

Detailed - Show all installation options and parameters.

Figure 15-18 Installation options

c. Click **Next** again to accept all the defaults installation options panel, as shown in Figure 15-19.



Specify options for installing enterprise applications and modules.

→ Step 1: Select installation options

Step 2 Map modules to servers

Step 3 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: KrbSample

- [✓] Create MBeans for resources
- [ ] Override class reloading settings for Web and EJB modules

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

.*\.dll=755#.*\.so=755#.*\.a=755#.*\.sl=755

Application Build ID: Unknown

- [ ] Allow dispatching includes to remote resources
- [ ] Allow servicing includes from remote resources

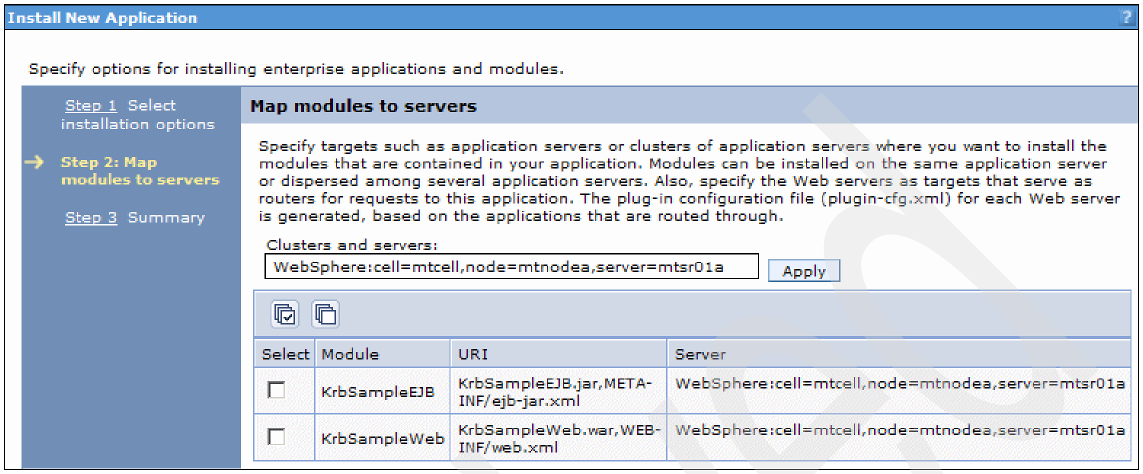
Business level application name: Create New BLA

Asynchronous Request Dispatch Type: Disabled

- [ ] Allow EJB reference targets to resolve automatically

Figure 15-19 Select installation options panel

d. Click **Next** again to accept all the defaults in the Map modules to servers panel, as shown in Figure 15-20.


Install New Application

Specify options for installing enterprise applications and modules.

Step 1 Select installation options

Step 2: Map modules to servers

Step 3 Summary

Map modules to servers

Specify targets such as application servers or clusters of application servers where you want to install the modules 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 to this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated, based on the applications that are routed through.

Clusters and servers:
WebSphere:cell=mtcell,node=mtnodea,server=mtsr01a [Apply]

Select	Module	URI	Server
☐	KrbSampleEJB	KrbSampleEJB.jar,META-INF/ejb-jar.xml	WebSphere:cell=mtcell,node=mtnodea,server=mtsr01a
☐	KrbSampleWeb	KrbSampleWeb.war,WEB-INF/web.xml	WebSphere:cell=mtcell,node=mtnodea,server=mtsr01a

Figure 15-20 Map modules to servers panel

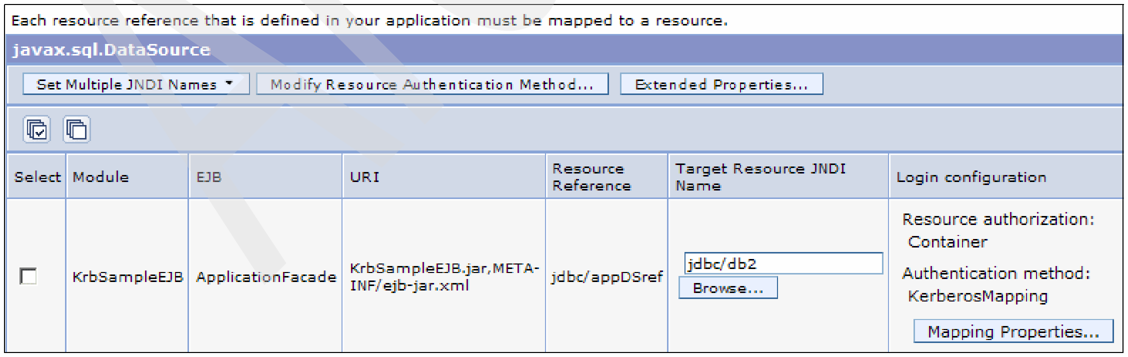
e. Click **Finish**.

3. Next, you need to verify that the JNDI name that the application uses to access the database is set to the name that you provided when the data source was created as follows:

   a. Navigate to **Applications** → **Application Types** → **WebSphere enterprise application**.

   b. Click **Resource References**.

   c. Enter `jdbc/db2` in the Target Resource JNDI Name text box, as shown in Figure 15-21, and click **OK**.


Each resource reference that is defined in your application must be mapped to a resource.

javax.sql.DataSource

Set Multiple JNDI Names ▾ | Modify Resource Authentication Method... | Extended Properties...

Select	Module	EJB	URI	Resource Reference	Target Resource JNDI Name	Login configuration
☐	KrbSampleEJB	ApplicationFacade	KrbSampleEJB.jar,META-INF/ejb-jar.xml	jdbc/appDSref	jdbc/db2 [Browse...]	Resource authorization: Container Authentication method: KerberosMapping [Mapping Properties...]

Figure 15-21 Map resource references to resources panel

4. Save and synchronize configuration changes to the nodes.

Note: The sample application is not designed for Java 2 security to be enabled because the policy files are not included. Make sure Java 2 security is disabled, or add the policy settings that are required to make this application work in a Java 2 secure environment.

Security role mapping steps

The scenario enables SAF authorization, which means that you need to create and map the Java EE application roles that are defined in the application in RACF.

Note: The test system that we used for this scenario had the SAF profile prefix, *MTCELL*, configured. The SAF profile prefix is added to all Java EE roles that WebSphere Application Server tries to query.

You need to check with the z/OS administrator to see whether you have a SAF profile prefix to determine the correct command. For more information, refer to the information center SAF profile prefix section:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/usec_safpropszos.html`

To create the Java EE application role in RACF and map RACF users to this role, follow these steps:

1. Start a TSO session, and log in to the ISPF program.
2. Select **ISPF Command Shell**.
3. Enter the RDEFINE EJBROLE MTCELL.user UACC(NONE) command to create the user (Java EE role) in RACF.
4. Enter the PE MTCELL.user CL(EJBROLE) ID(TAMT) ACCESS(READ) command to add TAMT to the role.

Note: A warning message might appear when you create the role and add users (refer to Step 3 on page 424 and Step 4 on page 424), such as:

```
ICH10006I RACLISTED PROFILES FOR EJBROLE WILL NOT REFLECT THE
ADDITION(S) UNTIL A SETROPTS REFRESH IS ISSUED.
```

You can ignore this warning because the REFRESH command refreshes the setup (refer to Step 5 on page 425).

5. Enter the SETROPTS RACLIST(EJBROLE) REFRESH command to refresh RACF.
6. Enter the PERMIT MTCELL ACCESS(READ) CLASS(APPL) ID(TAMT) command to permit TAMT to APPL access.

Note: You can disable the APPL class checking so that Step 6 on page 425 is not required to set up the user and role mapping. Refer to the information center at:

```
http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.zseries.doc/info/zseries/ae/usec_safpropszos.html
```

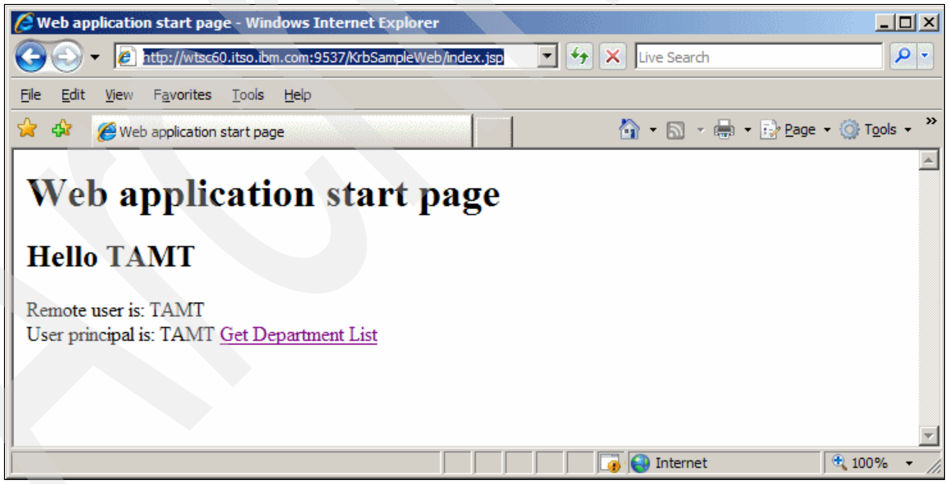
15.6.2 Testing the Kerberos sample application

You can use a SPNEGO-enabled Web browser on a client workstation that is part of the Microsoft domain to test the scenario as follows:

1. Open the browser to access the welcome URL for the application:

   `http://host:port/KrbSampleWeb/index.jsp`

   Figure 15-22 shows the resulting page.


Web application start page - Windows Internet Explorer
http://wtsc60.itso.ibm.com:9537/KrbSampleWeb/index.jsp
Live Search
File Edit View Favorites Tools Help
Web application start page
Page Tools
Internet
100%

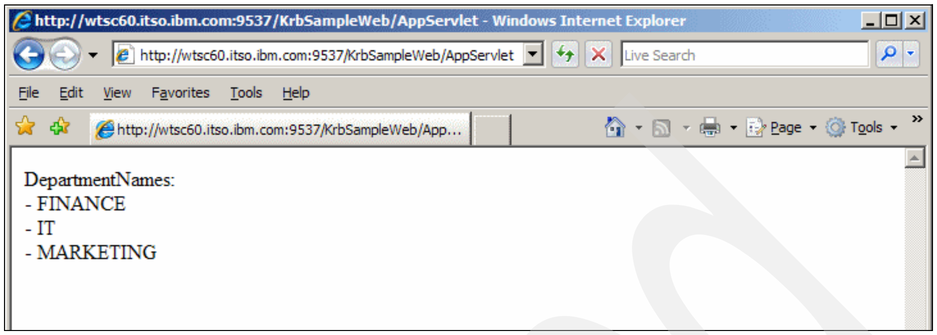
Web application start page

Hello TAMT

Remote user is: TAMT
User principal is: TAMT Get Department List

Figure 15-22 Sample Kerberos application Welcome page

2. Click the **GetDepartmentList** link to access the database, shown in Figure 15-23.


http://wtsc60.itso.ibm.com:9537/KrbSampleWeb/AppServlet - Windows Internet Explorer
http://wtsc60.itso.ibm.com:9537/KrbSampleWeb/AppServlet
Live Search
File
Edit
View
Favorites
Tools
Help
http://wtsc60.itso.ibm.com:9537/KrbSampleWeb/App...
Page
Tools

DepartmentNames:
- FINANCE
- IT
- MARKETING

Figure 15-23 Database response page

You now have access to the Welcome servlet without being prompted for a user ID and password. The SPNEGO token authenticated the client and the servlet calls an EJB, which in turn queries the database using the client's delegated Kerberos credentials.

16

Command-line administration with Kerberos authentication

The scenario in this chapter describes how to use the Kerberos authentication mechanism in the WebSphere Application Server command-line administrative tools. It shows how to configure SOAP and RMI connectors to perform authentication using Kerberos principal name or credential cache.

This chapter includes the following topics:

16.1 Scenario overview

The **wsadmin** tool runs scripts that allow you to configure the WebSphere Application Server environment, as well as to manage server runtime operations. The **wsadmin** tool is run by the administrator from an operating system command line.

You can use the **wsadmin** tool to perform the same tasks that can be performed from the administrative console. Similarly, authorization for performing tasks from **wsadmin** is based on the same administrative security role assignments that secures tasks that can be performed from the administrative console. To perform actions on a server that has security enabled requires that the **wsadmin** user be authenticated and then authorized to perform that task.

This scenario illustrates how a wsadmin client can authenticate using Kerberos.

Note: This scenario features the following components:

- Kerberos authentication for a **wsadmin** client
- An AIX key distribution center (KDC), using the IBM Tivoli Directory Server as the principal registry
- A **wsadmin** configuration for Kerberos authentication
- Authentication using a login prompt or cached credentials
- WebSphere Application Server configuration:
  - User registry: IBM Tivoli Directory Server
  - Authentication mechanism: Kerberos and LTPA
  - Kerberos authentication and configuration file

16.1.1 Step-by-step flow for the scenario

Figure 16-1 illustrates the basic flow for this scenario.

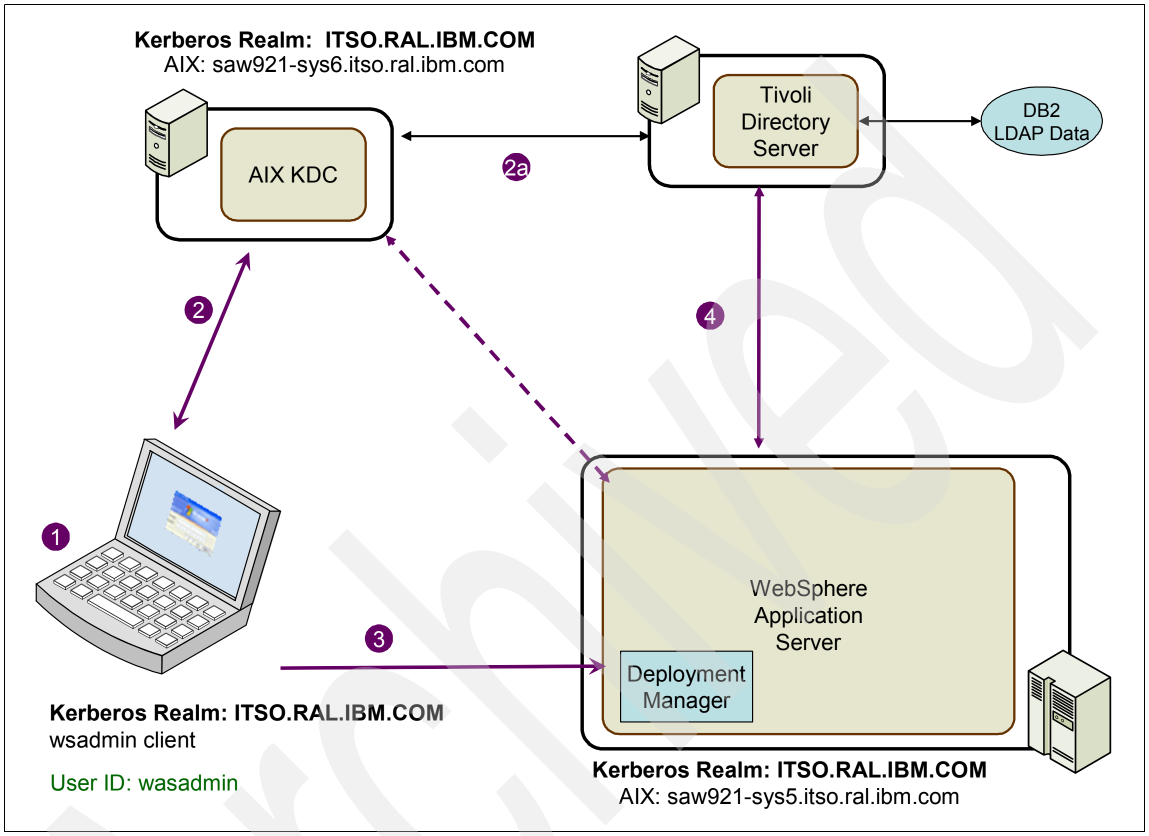

Kerberos Realm: ITSO.RAL.IBM.COM
AIX: saw921-sys6.itso.ral.ibm.com
AIX KDC
Tivoli Directory Server
DB2 LDAP Data
2a
2
4
1
3
WebSphere Application Server
Deployment Manager
Kerberos Realm: ITSO.RAL.IBM.COM
wsadmin client
User ID: wasadmin
Kerberos Realm: ITSO.RAL.IBM.COM
AIX: saw921-sys5.itso.ral.ibm.com

Figure 16-1 Step-by-step flow

This scenario includes the following steps:

1. The administrator starts the **wsadmin** client.
2. The **wsadmin** client obtains a Kerberos token (ticket-granting ticket and service ticket). Authentication takes place with the KDC using a user ID and password information (see “Kerberos authentication protocol” on page 5) or by using the local credential cache. The KDC validates the user in the principal registry, the IBM Tivoli Directory Server (ITDS) in this scenario.

3. The **wsadmin** client sends the Kerberos token to the WebSphere Application Server.
4. WebSphere Application Server decrypts the token and validates the user in the current WebSphere user registry for authorization. If the user is authenticated successfully and authorized, the **wsadmin** user interface starts.

At the end of this process, the administrator is authenticated successfully and is authorized to invoke administrative commands.

16.1.2 Scenario components

This scenario shows how to configure the **wsadmin** command-line tool to use the Kerberos authentication mechanism to WebSphere Application Server. To implement this scenario requires the following steps:

1. Configure the AIX KDC by creating the required Kerberos principals and generating the Kerberos keytab file.
2. Configure WebSphere Application Server. This task is multistep process that includes setting up Kerberos authentication mechanism.
3. Set up command-line administrative tools for Kerberos authentication mechanism.

The sections that follow explain the basic requirements for this scenario and include a step-by-step explanation of the flow.

16.1.3 Basic requirements for this scenario

This scenario assumes that following components are properly installed and configured:

- Kerberos KDC

  This scenario requires an AIX KDC that is configured with IBM Tivoli Directory Server as its registry.
- WebSphere Application Server V7.0.0.5

16.2 Configuring the AIX KDC

This section describes how to configure the AIX KDC. For more details about using an AIX KDC, see Chapter 3, "Configuring IBM Network Authentication Service KDC on AIX" on page 29.

16.2.1 Creating principals

Principals are used to authenticate users and services during Kerberos authentication. You need to create the following principals for this scenario:

- `wasadmin`

  Represents a user (the WebSphere Application Server administrator)

- `WAS/saw921-sys5.itso.ral.ibm.com`

  Represents the WebSphere Application Server service principal

  The service principal name for the WebSphere Application Server must be `<`*`KerberosName`*`>/<`*`fully qualified host name`*`>@REALM`. By convention *`KerberosName`* is `WAS`.

> **Important:** If you have nodes on multiple hosts in the cell, you need an SPN for each host.

To create principals, follow these steps:

1. Log in to the KDC system.
2. Start the **`kadmin`** tool using the following command:

   ```
   /usr/krb5/sbin/kadmin -p admin/admin
   ```

   This command prompts you for the password of the Kerberos principal `admin/admin`. Enter the password to start the command-line interface.
3. Create the principal for the administrator using the following command:

   ```
   addprinc -pw <password> wasadmin
   ```

Example 16-1 shows the output of this command.

Example 16-1 Creating a Kerberos principal

```
kadmin:  addprinc -pw <password> wasadmin
WARNING: no policy specified for wasadmin@ITSO.RAL.IBM.COM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "wasadmin@ITSO.RAL.IBM.COM" created.
```

4. Repeat step 3 to create the `WAS/saw921-sys5.itso.ral.ibm.com` principal.

16.2.2 Generating Kerberos keytab files

The Kerberos keytab file stores the Kerberos service principal names and keys that are used to validate incoming Kerberos token requests. The keytab file is created using tools that are provided by the KDC.

This scenario requires one keytab file for the WebSphere Application Server service principal.

Note: You must have a keytab file for WebSphere Application Server. It is highly recommended that you place all secret keys into one keytab file for easier management. In addition, each WebSphere Application Server node can read only a single file.

To create the keytab file for WebSphere Application Server service principal:

1. Log in to the KDC system.
2. Start the **kadmin** tool using the following command:

   ```
   /usr/krb5/sbin/kadmin -p admin/admin
   ```

   The Kerberos administrator user ID is admin/admin. This command prompts you for the password. Enter the password to start the command-line interface.
3. Create the keytab file using the following command:

   ```
   kadmin:  ktadd -k /etc/krb5/was.keytab -e arcfour-hmac:normal
   WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM
   ```

Example 16-2 shows the output of this command.

Example 16-2 Generating a keytab file for WebSphere Application Server

```
kadmin:  ktadd -k /etc/krb5/was.keytab -e arcfour-hmac:normal
WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM
Entry for principal
WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM with kvno 2,
encryption type ArcFour with HMAC/md5 added to keytab
WRFILE:/etc/krb5/was.keytab.
```

This command generates a new keytab file and stores it in `/etc/krb5/was.keytab`. The keytab file contains the Kerberos service principal name and its `arcfour-hmac` key.

Tip: Because this keytab file is transferred to the WebSphere Application Server system, which is usually separate from the KDC, and because it is used only to store Kerberos service principal names and keys that are used by the application server, it is good practice to create the keytab file in a separate file instead of the default file.

Note: If you have nodes on multiple hosts, you need to add keys for all the Kerberos server principal names to one `keytab` file by specifying the same `keytab` file in many **`ktadd`** commands.

WebSphere Application Server v7.0.0.5 supports the following encryption types:

– des-cbc-md5
– des-cbc-crc
– des3-cbc-sha1
– rc4-hmac
– arcfour-hmac
– arcfour-hmac-md5
– aes128-cts-hmac-sha1-96
– aes256-cts-hmac-sha1-96

For the latest information about encryption support, refer to the WebSphere Application Server V7 Information Center article “Creating a Kerberos configuration file” at:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/tsec_kerb_create_conf.html

In this scenario, the `keytab` file is encrypted using `arcfour-hmac`.

Caution: Ensure that you have a common encryption type for the Kerberos configuration file, the Kerberos `keytab` file, the Kerberos service principal name, and the Kerberos client.

4. Copy the new `keytab` file to the WebSphere Application Server system.

16.3 Configuring WebSphere Application Server

Next, you need to configure the WebSphere Application Server. Before you begin, ensure that you are using WebSphere Application Server V7.0.0.5 or later, because earlier versions do not provide full Kerberos authentication support.

16.3.1 Configuring Kerberos authentication

Before you configure Kerberos as the authentication mechanism, you need the following files:

- The Kerberos keytab file that contains Kerberos service principal. (You generated this file in the 16.2.2, "Generating Kerberos keytab files" on page 432.)
- The Kerberos configuration file (`krb5.ini` or `krb5.conf`). You create this in the next step.

Creating the Kerberos configuration file

You can create the configuration file using several methods (manual, tools, and so forth); however, the preferred method is to use the **`wsadmin`** command task.

To create the configuration file:

1. Go to the deployment manager profile /bin directory (*`profile_root`*`/bin`).
2. Start the **`wsadmin`** tool by issuing the following command:

```
[saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
n]# ./wsadmin.sh
```

3. Issue the following command:

```
wsadmin> $AdminTask createKrbConfigFile {-krbPath /tmp/krb5.conf
-realm ITSO.RAL.IBM.COM -kdcHost saw921-sys6.itso.ral.ibm.com
-dns itso.ral.ibm.com -keytabPath /tmp/was.keytab -encryption
arcfour-hmac }
```

This command using the following parameters:

– *krbPath* is the directory and file name of the Kerberos configuration file.
– *realm* is the Kerberos realm name.
– *kdcHost* is the KDC host name.
– *dns* is the DNS domain name.
– *keytabPath* is the directory and file name of the Kerberos keytab file.
– *encryption* is the encryption type used in the `keytab` file.

Example 16-3 shows the output of this command.

Example 16-3 Kerberos configuration file

```
[libdefaults]
        default_realm = ITSO.RAL.IBM.COM
        default_keytab_name = FILE:/tmp/was.keytab
        default_tkt_enctypes = arcfour-hmac
        default_tgs_enctypes = arcfour-hmac
        forwardable  = true
        renewable  = true
        noaddresses = true
        clockskew  = 300
[realms]
        ITSO.RAL.IBM.COM = {
                kdc = saw921-sys6.itso.ral.ibm.com:88
                default_domain = itso.ral.ibm.com
        }
[domain_realm]
        .itso.ral.ibm.com = ITSO.RAL.IBM.COM
```

4. Copy the Kerberos configuration file (`krb5.conf`) and the keytab (`was.keytab`) to the following directory *dmgr_profile_root*`/config/cells/cellname`.

   In this scenario, the files are copied to the following directory:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/config/cells/saw
   921-sys5Cell01
   ```

Best practice: Because the configuration and keytab files must be shared across all servers in the cell, it is best practice to put them in the deployment manager's `config/cell/`*`cellName`* directory. This directory is replicated across all nodes.

Note: The Kerberos executable tools (such as **`kinit`**, **`ktab`**, and **`klist`**) will not work if you place files in another directory. These tools look only in the default locations.

Configuring the authentication mechanism

To configure Kerberos authentication in the application server, follow these steps:

1. Open the WebSphere administrative console.
2. In the navigation tree select **Security** → **Global Security**.
3. In the Authentication section, click **Kerberos configuration**.
4. Specify the following values, as shown in Figure 16-2:
   - Kerberos service name:

     ```
     WAS
     ```

     The service name specified must match the service part from the Kerberos service principal name for WebSphere Application Server, in this case, the principal name is:

     ```
     WAS/saw921-sys5.itso.ral.ibm.com@ITSO.RAL.IBM.COM
     ```

     So, the service is `WAS`.
   - Kerberos configuration file:

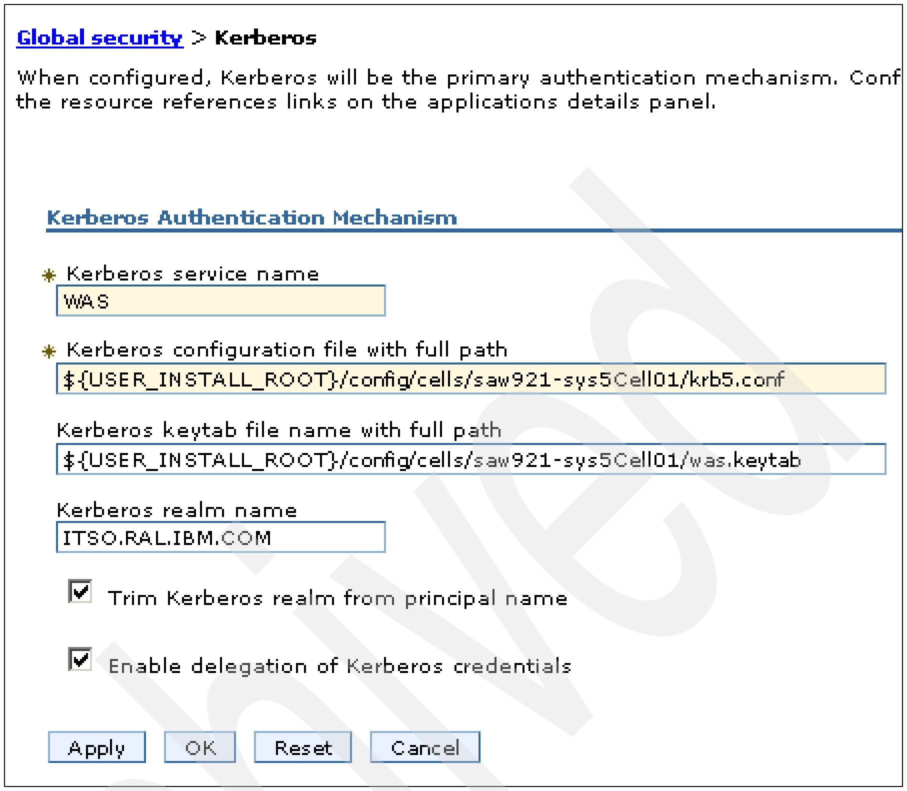
     ```
     ${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/krb5.conf
     ```
   - Kerberos keytab file:

     ```
     ${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/was.keytab
     ```
   - Kerberos realm name:

     ```
     ITSO.RAL.IBM.COM
     ```
   - Select the “Trim Kerberos realm from principal name” option.

     This option specifies whether Kerberos removes the suffix of the principal user name, starting from the at sign (@) that precedes the Kerberos realm name. If this attribute is set to `true`, the suffix of the principal user name is removed. If this attribute is set to `false`, the suffix of the principal name is retained.
   - Select the “Enable delegation of Kerberos credentials” option.

     This option specifies whether the Kerberos delegated credentials are to be stored in the subject by the Kerberos authentication. It also enables an application to retrieve the stored credentials and to propagate them to other applications downstream for additional Kerberos authentication with the credential from the Kerberos client.


Global security > Kerberos

When configured, Kerberos will be the primary authentication mechanism. Conf
the resource references links on the applications details panel.

Kerberos Authentication Mechanism

* Kerberos service name
WAS

* Kerberos configuration file with full path
${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/krb5.conf

Kerberos keytab file name with full path
${USER_INSTALL_ROOT}/config/cells/saw921-sys5Cell01/was.keytab

Kerberos realm name
ITSO.RAL.IBM.COM

[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

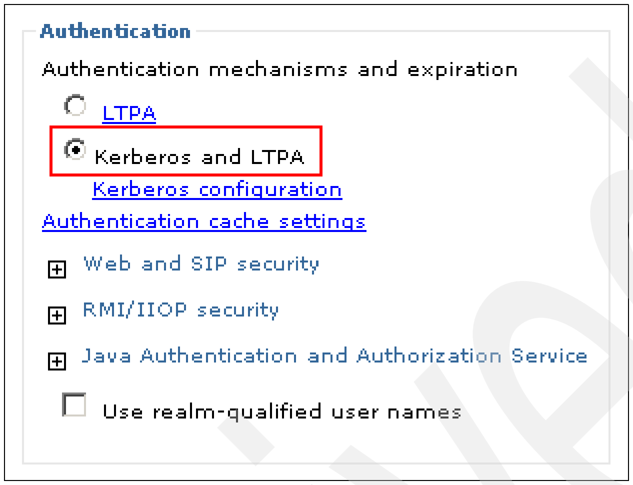
Figure 16-2 Defining authentication properties

Tip: You can use the WebSphere variable `${USER_INSTALL_ROOT}` in the path names to ensure that all nodes have access to the Kerberos configuration file and keytab file.

It is also recommended that you specify the path to the keytab file, instead of relying on the default path that is stored in the configuration file.

5. Click **OK**.

6. Navigate back to the Global Security page. Ensure that the “Kerberos and LTPA” option in the Authentication section is selected, as shown in Figure 16-3.


Authentication

Authentication mechanisms and expiration

( ) LTPA

(●) Kerberos and LTPA

Kerberos configuration

Authentication cache settings

⊞ Web and SIP security

⊞ RMI/IIOP security

⊞ Java Authentication and Authorization Service

☐ Use realm-qualified user names

Figure 16-3 Selecting authentication mechanism

7. Click **Apply**, and then click **Save**.

If you experience problems during authentication, see “Enabling the JGSS and KRB traces” on page 469 for instructions about how to enable Kerberos traces.

You can find more information about how to configure Kerberos authentication in the information center:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/tsec_kerb_auth_mech.html`

16.3.2 Configuring the user registry

WebSphere Application Server uses Kerberos mechanisms to authenticate the user; however, for authorization it still queries a user registry. For a query to be successful, there must be a mapping between the Kerberos principal name and some searchable user attribute in the registry. For simplicity of the example, we use the default file registry and the user ID matches the trimmed Kerberos principal name.

16.4 Setting up the command-line administration tools

In this section, we explain how to configure the command-line administration tool, **wsadmin**, to use Kerberos authentication to WebSphere Application Server. We show how to use both the SOAP and the RMI connectors and how to implement authentication using the Kerberos principal name and password or the Kerberos credential cache (`krb5Ccache`).

16.4.1 Configuring the SOAP connector

First, we explain how to configure WebSphere Application Server for Kerberos authentication for clients using the SOAP connector.

Authenticating with Kerberos principal name and password

To configure for authentication using a principal name and password, complete the following steps:

1. Go to the deployment managers *profile_home*/`properties` folder:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/properties
   ```

2. Open the `soap.client.props` file and edit the following properties:
   - `com.ibm.SOAP.authenticationTarget=KRB5`
   - `com.ibm.SOAP.loginSource=stdin`
   - `com.ibm.CORBA.krb5ConfigFile=/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf`

3. Go to the deployment managers *profile_home*/`bin` folder:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin
   ```

   Execute the following command (all in one line):

   ```
   [saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
   n]# ./wsadmin.sh -conntype SOAP -host
   saw921-sys5.itso.ral.ibm.com -port 8879 -javaoption
   "-Dcom.ibm.security.jgss.debug=all
   -Dcom.ibm.security.krb5.Krb5Debug=all"
   ```

   This command uses the following parameters:
   - `-conntype` is the type of the connector
   - `-host` is the deployment manager host name
   - `-port` is the deployment manager SOAP port

Tip: Add the following parameter to the **wsadmin** invocation to get debugging information for the Kerberos authentication.

```
   -javaoption "-Dcom.ibm.security.jgss.debug=all
   -Dcom.ibm.security.krb5.Krb5Debug=all"
```

4. At the login prompt, provide the following information as shown in Example 16-4:
   - User name: `wasadmin` (Kerberos principal)
   - Kerberos password (not the user registry password)

Example 16-4 Running wsadmin over RMI connector

```
[saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin]#
./wsadmin.sh -conntype SOAP -host saw921-sys5.itso.ral.ibm.com -port
8879 -javaoption "-Dcom.ibm.security.jgss.debug=all
-Dcom.ibm.security.krb5.Krb5Debug=all"
....
Realm/Cell Name: <default>
Username: wasadmin
Password:
...
[JGSS_DBG_CTX] initSecContext: returning buffer, len=1201
[KRB_DBG_KDC] Credentials:main:Client Name:wasadmin@ITSO.RAL.IBM.COM
[JGSS_DBG_CTX] succesfully removed service creds from subject
WASX7209I: Connected to process "dmgr" on node
saw921-sys5CellManager01 using SOAP connector; The type of process
is: DeploymentManager
WASX7029I: For help, enter: "$Help help"
wsadmin>
```

The output shows that you are authenticated successfully using a Kerberos user principal name and password and are connected to the deployment manager.

Authenticating using cached Kerberos credentials

To configure for authentication using cached credentials, complete the following steps:

1. Go to the deployment managers *`profile_home`*`/properties` folder:

   `/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/properties`

2. Open the `soap.client.props` file and edit the following properties:

   – `com.ibm.SOAP.authenticationTarget=KRB5`

   – `com.ibm.SOAP.loginSource=krb5Ccache`

   – `com.ibm.SOAP.krb5ConfigFile=/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf`

   If the credential cache is not in the default location, it has to be specified using the `com.ibm.CORBA.krb5CcacheFile` property.

3. Open `wsjaas_client.conf` and edit the entry with the bolded values, as shown in Example 16-5.

   Example 16-5 Edit the wsjass_client.conf file

```
WSKRB5Login{
    com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapperClient
required credsType=INITIATOR useFirstPass=false forwardable=false
renewable=false noAddress=false;
};
```

4. Create a Kerberos credential cache by executing the following command (all on one line):

```
[saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
n]#../java/jre/bin/java
-Djava.security.krb5.conf=/usr/IBM/WebSphere/AppServer/profiles/s
cn3Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf
com.ibm.security.krb5.internal.tools.Kinit wasadmin
```

   Example 16-6 shows the output from this command.

   Example 16-6 Creating a credential cache

```
[root@saw921-sys5 bin]# ../java/jre/bin/java
-Djava.security.krb5.conf=/usr/IBM/WebSphere/AppServer/profiles/scn3
Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf
com.ibm.security.krb5.internal.tools.Kinit wasadmin
Password for wasadmin@ITSO.RAL.IBM.COM:

Done!
New ticket is stored in cache file /root/krb5cc_root
```

5. Go to the deployment managers `profile_home/bin` folder:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin
   ```

   Then, execute the following command (all on one line):

   ```
   [saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
   n]# ./wsadmin.sh -conntype SOAP -host
   saw921-sys5.itso.ral.ibm.com -port 8879 -javaoption
   "-Dcom.ibm.security.jgss.debug=all
   -Dcom.ibm.security.krb5.Krb5Debug=all"
   ```

   The **wsadmin** user interface should start *without* providing a user name and password as shown in Example 16-7.

Example 16-7 Authenticating with cached credentials

```
[saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin]#
./wsadmin.sh -conntype SOAP -host saw921-sys5.itso.ral.ibm.com -port
8879 -javaoption "-Dcom.ibm.security.jgss.debug=all
-Dcom.ibm.security.krb5.Krb5Debug=all"
...
[JGSS_DBG_CTX] initSecContext: returning buffer, len=1186
[KRB_DBG_KDC] Credentials:main:Client Name:wasadmin@ITSO.RAL.IBM.COM
[JGSS_DBG_CTX] succesfully removed service creds from subject
WASX7209I: Connected to process "dmgr" on node
saw921-sys5CellManager01 using SOAP connector;  The type of process
is: DeploymentManager
WASX7029I: For help, enter: "$Help help"
wsadmin>
```

16.4.2 Configuring the RMI connector

This section shows how to configure WebSphere Application Server for Kerberos authentication for clients using the RMI connector.

Authenticating using Kerberos principal name and password

To configure for authentication using a principal name and password, complete the following steps:

1. Go to the deployment managers *`profile_home`*`/properties` folder:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/properties
   ```

2. Open the `sas.client.props` file, and edit the following properties:

   – `com.ibm.CORBA.authenticationTarget=KRB5`

   – `com.ibm.CORBA.loginSource=stdin`

– `com.ibm.CORBA.krb5ConfigFile=/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf`

3. Go to the deployment managers `profile_home/bin` folder:

   ```
   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin
   ```

   Execute the following command (all in one line):

   ```
   [saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin]#
   ./wsadmin.sh -conntype RMI -host saw921-sys5.itso.ral.ibm.com
   -port
   9809 -javaoption "-Dcom.ibm.security.jgss.debug=all
   -Dcom.ibm.security.krb5.Krb5Debug=all"
   ```

   This command uses the following parameters:

   – `-conntype` is the type of the connector
   – `-host` is the deployment manager host name
   – `-port` is the deployment manager port

4. At the login prompt provide the following information, as shown in Example 16-8:

   – User name: `wasadmin` (Kerberos principal)
   – Kerberos password (not the user registry password)

Example 16-8 Running wsadmin over RMI connector

```
[saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3AppSvr01/bin
]# ./wsadmin.sh -conntype RMI -host saw921-sys5.itso.ral.ibm.com
-port 9809 -javaoption "-Dcom.ibm.security.jgss.debug=all
-Dcom.ibm.security.krb5.Krb5Debug=all"
.....
Realm/Cell Name: <default>
Username: wasadmin
Password:
....
WASX7209I: Connected to process "dmgr" on node
saw921-sys5CellManager01 using RMI connector; The type of process
is: DeploymentManager
....
[JGSS_DBG_CRED] Got client creds from Subject
[KRB_DBG_KDC] Credentials:P=839735:O=0:CT:Client
Name:wasadmin@ITSO.RAL.IBM.COM
[JGSS_DBG_CRED] Got creds for wasadmin@ITSO.RAL.IBM.COM
[JGSS_DBG_CRED] Kerberos ticket end time: (1250343261 secs) Sat Aug
15 09:34:21 EDT 2009
[JGSS_DBG_CRED] Remaining lifetime of kerberos ticket: 86377 secs
[JGSS_DBG_CRED] Requested initLifetime: 2147483647 secs
```

```
[JGSS_DBG_CRED] Starttime: (1250256884 secs) Fri Aug 14 09:34:44 EDT
2009
[JGSS_DBG_CRED] InitLifetime: 86377 secs
[JGSS_DBG_CRED] Adding mech cred
[JGSS_DBG_CRED] getName found name: wasadmin@ITSO.RAL.IBM.COM,
mech=1.2.840.113554.1.2.2
[JGSS_DBG_CRED] Krb5 name type = 0
WASX7029I: For help, enter: "$Help help"
wsadmin>
```

Authenticating using the Kerberos credential cache

To configure for authentication using the Kerberos credential cache, complete the following steps:

1. Go to the deployment managers *profile_home*/properties folder:

   /usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/properties

2. Open the sas.client.props file, and edit the following properties:

   – com.ibm.CORBA.authenticationTarget=KRB5

   – com.ibm.CORBA.loginSource=krb5Ccache

   – com.ibm.CORBA.krb5ConfigFile=/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf

   If the credential cache is not in the default location, it has to be specified using the com.ibm.CORBA.krb5CcacheFile property.

3. Open the wsjaas_client.conf, and edit the bolded values, as shown in Example 16-9.

   Example 16-9 Edit the wsjass_client.conf file

```
WSKRB5Login{
    com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapperClient
required credsType=INITIATOR useFirstPass=false forwardable=false
renewable=false noAddress=false;
};
```

4. Create a credential cache by executing the following command (all on one line):

```
[saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
n]#../java/jre/bin/java
-Djava.security.krb5.conf=/usr/IBM/WebSphere/AppServer/profiles/s
cn3Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf
com.ibm.security.krb5.internal.tools.Kinit wasadmin
```

Example 16-10 shows the output from this command.

Example 16-10 Creating a credential cache

```
[root@saw921-sys5 bin]# ../java/jre/bin/java
-Djava.security.krb5.conf=/usr/IBM/WebSphere/AppServer/profiles/scn3
Dmgr01/config/cells/saw921-sys5Cell01/krb5.conf
com.ibm.security.krb5.internal.tools.Kinit wasadmin
Password for wasadmin@ITSO.RAL.IBM.COM:

Done!
New ticket is stored in cache file /root/krb5cc_root
```

5. Go to the deployment manager `profile_home/bin` folder:

   `/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bin`

   Execute the following command (all on one line):

   ```
   [saw921-sys5][/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/bi
   n]# ./wsadmin.sh -conntype RMI -host saw921-sys5.itso.ral.ibm.com
   -port 9809 -javaoption "-Dcom.ibm.security.jgss.debug=all
   -Dcom.ibm.security.krb5.Krb5Debug=all"
   ```

6. The **`wsadmin`** user interface should start *without* having to provide a user name and password as shown in Example 16-11.

Example 16-11 Authenticating with cached credentials

```
...
[JGSS_DBG_CRED] Got client creds from Subject
[KRB_DBG_KDC] Credentials:P=960626:O=0:CT:Client
Name:wasadmin@ITSO.RAL.IBM.COM
[JGSS_DBG_CRED] Got creds for wasadmin@ITSO.RAL.IBM.COM
[JGSS_DBG_CRED] Kerberos ticket end time: (1248387862 secs) Thu Jul
23 18:24:22 EDT 2009
[JGSS_DBG_CRED] Remaining lifetime of kerberos ticket: 85866 secs
[JGSS_DBG_CRED] Requested initLifetime: 2147483647 secs
[JGSS_DBG_CRED] Starttime: (1248301996 secs) Wed Jul 22 18:33:16 EDT
2009
[JGSS_DBG_CRED] InitLifetime: 85866 secs
[JGSS_DBG_CRED] Adding mech cred
[JGSS_DBG_CRED] getName found name: wasadmin@ITSO.RAL.IBM.COM,
mech=1.2.840.113554.1.2.2
[JGSS_DBG_CRED] Krb5 name type = 0
WASX7029I: For help, enter: "$Help help"
wsadmin>
```

Implementing Kerberos in a flexible management environment

This chapter describes the steps to configure Kerberos in a flexible management environment. The scenario in this chapter uses the following new components implemented in WebSphere Application Server V7:

- The job manager
- The administrative agent

This chapter describe the use of Kerberos tokens in this environment and provides examples of using Kerberos in a job manager architecture. It includes the following topics:

- Scenario overview
- Configuring the job manager
- Configuring the administrative agent and application server systems
- Configuring wsadmin to use Kerberos
- Validating the solution

17.1 Scenario overview

WebSphere Application Server V7.0 introduced the concept of flexible management to improve administrative tasks in a multi-cell environment. Two new management server processes provide these services: t

- The job manager
- The administrative agent

The administrative agent provides a single point of administration for multiple stand-alone application servers (also referred to as *unfederated application servers*) on a single system.

With the job manager, you can submit administrative jobs for application servers that are registered to administrative agents and for deployment managers. This ability provides a single point of administration for multiple distributed and stand-alone server environments. You can also submit jobs to a large number of servers over a geographically dispersed area. The jobs can be used to manage applications, to modify the WebSphere configuration, or to do general purpose tasks such as run a script.

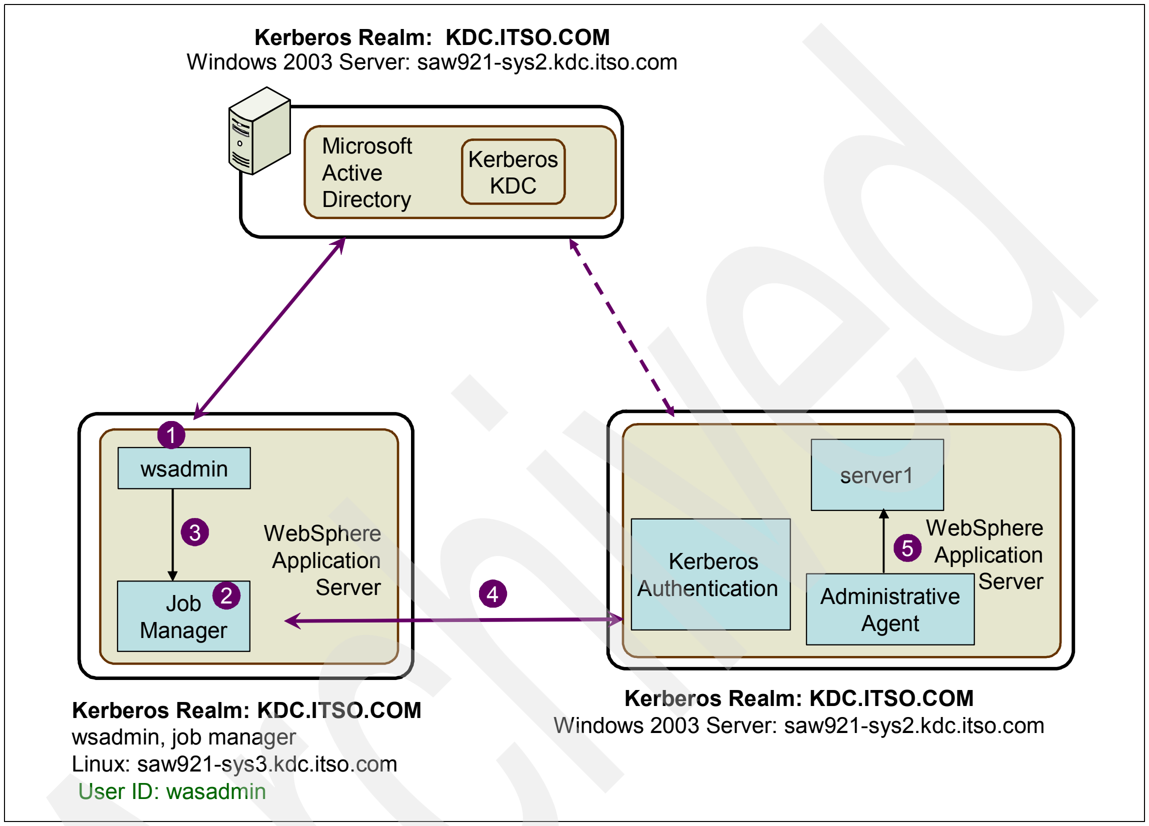
In this scenario, we illustrate how to implement the Kerberos authentication protocol for an architecture that uses these flexible management features. An administrator uses **`wsadmin`** to create a job in the job manager. This job is executed in an application server that is registered into the job manager server.

Note: This scenario features the following components:

- Kerberos authentication for a wsadmin client
- Microsoft Active Directory
- Authentication using a login prompt or cached credentials
- WebSphere Application Server job manager configuration:
  - User registry: Microsoft Active Directory
  - Authentication mechanism: Kerberos and LTPA
  - Kerberos authentication and configuration file

17.1.1 Step-by-step flow of this scenario

Figure 17-1 shows the environment for the scenario used to illustrate the concepts.



Kerberos Realm: KDC.ITSO.COM
Windows 2003 Server: saw921-sys2.kdc.itso.com
Microsoft Active Directory
Kerberos KDC
1
wsadmin
3
WebSphere Application Server
Job Manager
2
4
server1
5
WebSphere Application Server
Kerberos Authentication
Administrative Agent
Kerberos Realm: KDC.ITSO.COM
wsadmin, job manager
Linux: saw921-sys3.kdc.itso.com
User ID: wasadmin
Kerberos Realm: KDC.ITSO.COM
Windows 2003 Server: saw921-sys2.kdc.itso.com

Figure 17-1 Scenario overview

The scenario environment consists of a job manager on one system (`saw921-sys3`) that sends jobs to the administrative agent on a second system (`saw921-sys2`). The key distribution center (KDC) server used by both WebSphere installations is also on `saw921-sys2`.

This scenario includes the following steps:

1. The administrator uses **`wsadmin`** interactively to generate a Kerberos token or uses the Kerberos cached credential to provide credentials to the job manager.
2. The job manager validates the Kerberos token to log in the user and to accept the job.

3. The administrator submits a job in **`wsadmin`** to be executed. The submitted job and the user security token, the Kerberos token, is persisted to the job database.
4. The administrative agent checks the job manager for jobs it needs to review:
   a. The administrative agent connects to the job manager using HTTPS, and submits a client certificate in the HTTP header.
   b. The job manager verifies the client certificate, and in the HTTPS reply sends, back the user's security token and the job parameters. The job manager fetches the information from the job database.
5. The administrative agent verifies the user security token that is provided.
6. The job is executed.

17.1.2 Scenario components

This scenario requires the following components:

- A system running the job manager
  - Platform: Linux Red Hat Enterprise Linux AS release 4 (Nahant Update 7)
  - Machine name: `saw921-sys3.kdc.itso.com`
- A second system running the administrative agent process and a stand-alone application server
  - Platform: AIX 5.3 0006910F4C00
  - Machine name: `saw921-sys2.kdc.itso.com`
- A Kerberos KDC
  - Platform: AIX 5.3 0006910F4C00
  - Machine name: `saw921-sys2.kdc.itso.com`

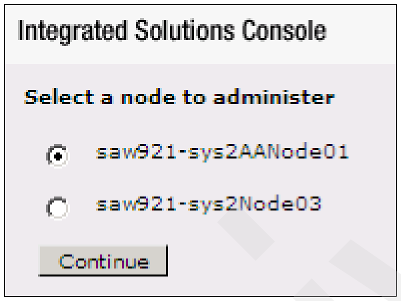
17.1.3 Basic requirements for this scenario

This scenario assumes that the following configuration is in place:

- One Linux system with WebSphere Application Server V7.0.0.5 or later installed and a job manager profile created and running.
- One Windows system with WebSphere Application Server V7.0.0.5 or later installed with the following profiles created:
  - Administrative agent profile
  - Application server profile
- A Microsoft Kerberos KDC running on Windows 2003 server. In this scenario, we call the KDC realm *KDC.ITSO.com*.

- The **registerNode** command used to register the stand-alone application server with the administrative agent. After the registration completes, the administrative agent can manage the stand-alone application server.

  After the registration is complete, the console for the server is available only through administrative agent administrative console port. Accessing the administrative agent console takes you to a node selection window (shown in Figure 17-2) where you have the option of accessing the console for the administrative agent (`saw921-sys2AANode01`) or the console for the registered application server (`saw921-sys2Node03`).


Integrated Solutions Console

Select a node to administer

[✓ saw921-sys2AANode01]

saw921-sys2Node03

Figure 17-2 Selecting node at administrative agent console

- The administrative agent is registered with the job manager, which means that the job manager can submit and manage jobs for the stand-alone application servers that are registered to the administrative agent. The registration is done with the **wsadmin $AdminTask registerWithJobManager** command.

17.2 Configuring the job manager

This section provides information about configuring the WebSphere Application Server job manager system for the scenario.

17.2.1 Configuring LDAP as the user registry

For this scenario, we used Microsoft Active Directory as the LDAP user registry that is used by the WebSphere Application Server components that are involved in this solution.

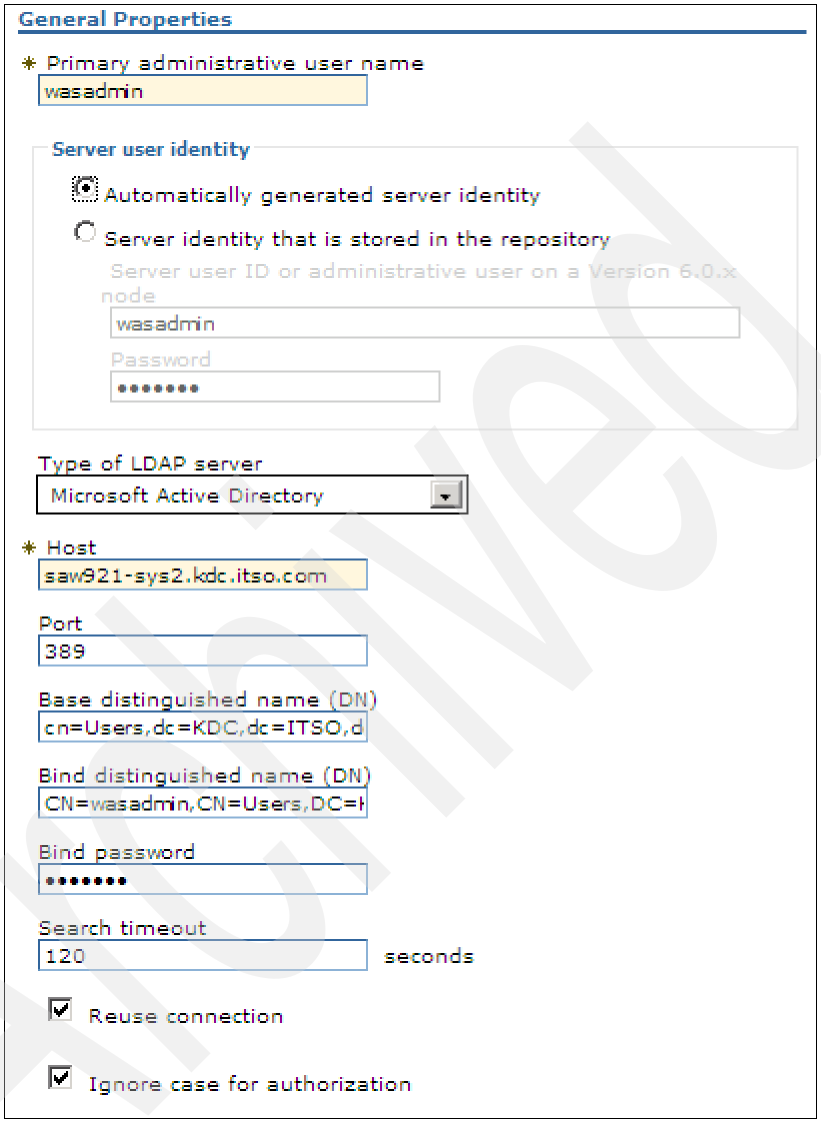
The following steps show how to configure the LDAP server as the user registry for the job manager. You can repeat these same procedures for the administrative agent and the stand-alone WebSphere Application Server (server1).

1. Open the console, and select **Security** → **Global Security**.
2. Select **Standalone LDAP registry** from the Available realm definitions drop-down menu, and click **Configure**.
3. In the stand-alone LDAP registry configuration panel, enter the information about the LDAP server. For the scenario, we used the values listed in Table 17-1.

Table 17-1 Values for LDAP configuration

Property	**Value**
Primary administrative user name	wasadmin
Type of LDAP server	Microsoft Active Directory
Host	saw921-sys2.kdc.itso.com
Port	389
Base distinguished name (DN)	cn=Users,dc=KDC,dc=ITSO,dc=COM
Bind distinguished name (DN)	cn=wasadmin,cn=Users,dc=KDC,dc=ITSO, dc=COM
Bind password	< password >

Figure 17-3 shows the LDAP configuration panel complete.


General Properties

* Primary administrative user name: wasadmin

Server user identity

- [✓ Automatically generated server identity]
- Server identity that is stored in the repository

Server user ID or administrative user on a Version 6.0.x node: wasadmin

Password: •••••••

Type of LDAP server: Microsoft Active Directory

* Host: saw921-sys2.kdc.itso.com

Port: 389

Base distinguished name (DN): cn=Users,dc=KDC,dc=ITSO,d

Bind distinguished name (DN): CN=wasadmin,CN=Users,DC=I

Bind password: •••••••

Search timeout: 120 seconds

[✓] Reuse connection

[✓] Ignore case for authorization

Figure 17-3 Configuring LDAP standalone server

4. Click **Apply**, and save the configuration.
5. Click **Test Connection** to verify that the configuration is correct.

6. Select **Security** → **Global Security** again. Then, select **Standalone LDAP registry** from the Available realm definitions drop-down menu, and click **Set as current**.
7. Click **OK** to save the configuration and to restart the server.

17.2.2 Enabling Kerberos

This section explains how to enable Kerberos in WebSphere Application Server for this scenario.

Note: Job manager jobs can be associated with caller credentials—either with Lightweight Third-Party Authentication (LTPA) or Kerberos or with specified credentials using a user ID and password. Both types of credentials are stored with the job. The password is obfuscated using the standard utilities and can be encrypted when the password encryption plug point is enabled. LTPA and Kerberos tokens are refreshed in the job that was created in the job manager as long as the authentication mechanism allows them to be refreshed.

Creating the keytab file for the job manager system

First, you need to create a keytab file that the job manager will use. To accomplish this task, execute the **`ktpass`** command in the same system where the Microsoft Active Directory is running, as shown in Example 17-1. See 4.6.2, “Creating the Kerberos keytab file” on page 86for information about using the **`ktpass`** tool.

This process assumes a Kerberos service principal name (SPN) of `WAS/saw921-sys3.kdc.itso.com` was created in the KDC realm.

Example 17-1 The ktpass tool example

```
C:\>ktpass -princ WAS/saw921-sys3.kdc.itso.com@KDC.ITSO.COM -mapuser
saw921-sys3 -pass <password> -out c:\kdc_itso_com.keytab
Targeting domain controller: saw921-sys2.kdc.itso.com
Using legacy password setting method
Successfully mapped WAS/saw921-sys3.kdc.itso.com to saw921-sys3.
WARNING: pType and account type do not match. This might cause
problems.
Key created.
Output keytab to c:\kdc_itso_com.keytab:
Keytab version: 0x502
keysize 77 WAS/saw921-sys3.kdc.itso.com@KDC.ITSO.COM ptype 0
(KRB5_NT_UNKNOWN)
```

```
vno 3 etype 0x17 (RC4-HMAC) keylength 16
(0x31a1f13010baa0c4f9fc8ead6de09db5)
```

Securely transfer this keytab file and place it in the `/etc` directory for the job manager.

Creating the Kerberos configuration file

Next, create the Kerberos configuration file (`krb5.conf`) on the job manager system. Execute the **`wsadmin createKrbConfigFile`** command from the <*Profile_Home*>/bin directory. The command creates the Kerberos file as shown in Example 17-2.

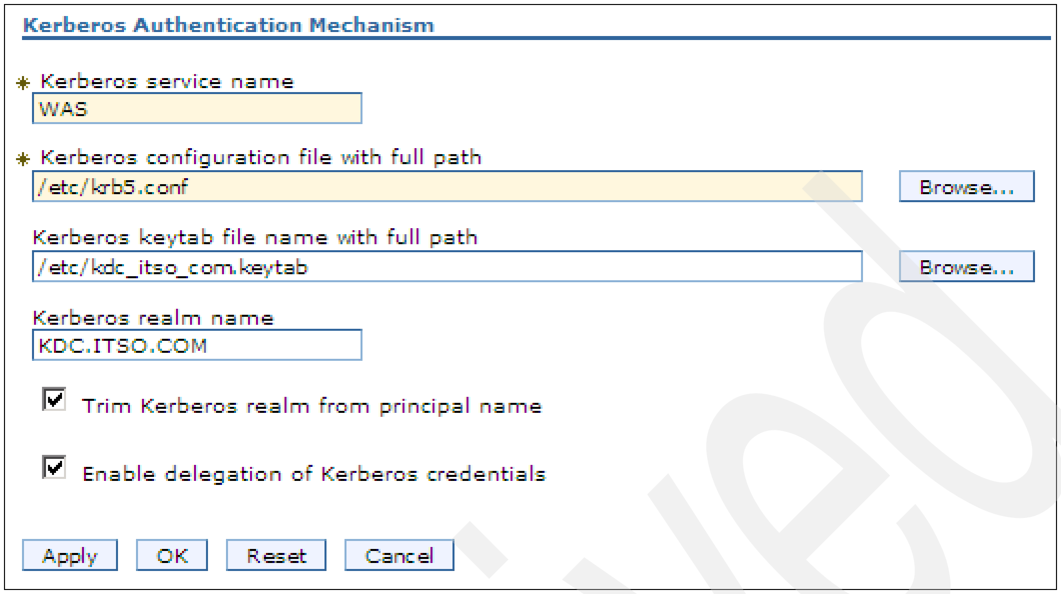
Example 17-2 The wsadmin command execution output

```
wsadmin>$AdminTask createKrbConfigFile {-krbPath /etc/krb5.conf -realm
KDC.ITSO.COM -kdcHost saw921-sys2.kdc.itso.com -dns kdc.itso.com
-keytabPath /etc/saw921-sys3.keytab}
/etc/krb5.conf has been created.
```

Enabling Kerberos

Now, enable Kerberos in WebSphere Application Server as follows:

1. In the WebSphere administrative console and select **Security** → **Global Security**. Click **Kerberos Configuration**.
2. Enter the values for the Kerberos service name. (The Kerberos service name needs to match with the Kerberos SPN.) Enter the path for the Kerberos configuration file, the path for the keytab file, and the realm name, as shown in Figure 17-4.


Kerberos Authentication Mechanism

* Kerberos service name: WAS

* Kerberos configuration file with full path: /etc/krb5.conf

Kerberos keytab file name with full path: /etc/kdc_itso_com.keytab

Kerberos realm name: KDC.ITSO.COM

[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

Figure 17-4 Configuring Kerberos

3. Click **OK**, and save the configuration.
4. Restart the job manager.

17.3 Configuring the administrative agent and application server systems

Repeat all the steps in "Configuring the job manager" on page 451 for the administrative agent and application server configuration. Remember to change the name of the Kerberos SPN because these processes are running in a different system. In our scenario, they are running on `saw921-sys2.kdc.itso.com`.

17.4 Configuring wsadmin to use Kerberos

The last step is to enable **wsadmin** to use Kerberos as its authentication protocol. This step makes it possible to authenticate from the **wsadmin** tool to WebSphere Application Server using a Kerberos principal name and password or using the Kerberos credential cache (`krb5Ccache`).

This configuration is done under the properties directory for the job manager profile. In this scenario, we used the SOAP protocol to connect to the job manager instance.

Note: If you want to use the RMI connector instead SOAP, see 16.4.2, "Configuring the RMI connector" on page 442 for the steps that are required to configure the properties files to use Kerberos as the authentication mechanism for RMI.

To configure **wsadmin** to use Kerberos, follow these steps:

1. In the <*Job_Manager_Profile_Home*>`/properties` directory, open and edit the `soap.client.props` file. Table 17-2 shows the parameters and values that need to be updated.

   Table 17-2 Update these soap.client.props properties

Property	Value
com.ibm.SOAP.securityEnabled	true
com.ibm.SOAP.authenticationTarget	KRB5
com.ibm.SOAP.loginSource	krb5Ccache
com.ibm.SOAP.krb5ConfigFile	/etc/krb5.conf

   Note: You can also enter `krb5Ccache:prompt` for the `com.ibm.SOAP.loginSource` property. This property indicates that authentication should be tried using `krb5Ccache` first. If that fails, then it reverts back to prompt.

2. Also in the <*Job_Manager_Profile_Home*>`/properties` directory, open and edit the `wsjaas_client.conf` file. Set the value for the following options in the `WSKRB5Login` entry to `false`, as shown in Example 17-3:
   - `useDefaultKeytab`
   - `useDefaultCcache`

– tryFirstPass
– useFirstPass
– forwardable
– renewable
– noaddress

Example 17-3 wsjaas_client.conf

```
WSKRB5Login{
    com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapperClient
required credsType=INITIATOR useFirstPass=false forwardable=false
renewable=false noAddress=false;
};
```

Save and close the file.

3. Now, test to see if Kerberos authentication is working. First, authenticate with the KDC using the **Kinit** class, as shown in Example 17-4.

Example 17-4 KDC authentication

```
[root@saw921-sys3 bin]# /opt/WebSphere/AppServer/java/jre/bin/java
-Djava.security.krb5.conf=/etc/krb5.conf
com.ibm.security.krb5.internal.tools.Kinit wasadmin
Password for wasadmin@KDC.ITSO.COM:

Done!
New ticket is stored in cache file /root/krb5cc_root
```

4. To verify if the Kerberos credential was stored in the cache file, execute the **Klist** class, as shown in Example 17-5.

Example 17-5 The klist output

```
[root@saw921-sys3 bin]# /opt/WebSphere/AppServer/java/jre/bin/java
-Djava.security.krb5.conf=/etc/krb5.conf
com.ibm.security.krb5.internal.tools.Klist

Credentials cache: /root/krb5cc_root
Default principal: wasadmin@KDC.ITSO.COM
Number of entries: 1

[1] Service principal: krbtgt/KDC.ITSO.COM@KDC.ITSO.COM
        Valid starting: Tuesday, July 21, 2009 at 9:47:41 AM
        Expires: Tuesday, July 21, 2009 at 7:47:41 PM
```

From the output of this test, you can see some information about the authentication that was made into KDC. For example, you can verify that the

Kerberos ticket was stored in the /root/krb5cc_root file. This file is used to authenticate to the job manager because the credentials are stored there.

5. Next execute wsadmin.sh from the bin directory for the job manager profile, as shown in Example 17-6.

Example 17-6 Executing wsadmin.sh

```
[root@saw921-sys3 bin]#
/opt/WebSphere/AppServer/profiles/JobMgr01/bin/wsadmin.sh
WASX7209I: Connected to process "jobmgr" on node saw921-sys3JobMgr01
using SOAP connector;  The type of process is: JobManager
WASX7029I: For help, enter: "$Help help"
wsadmin>
```

If the Kerberos authentication is configured correctly, you receive the **wsadmin** prompt, which means that the credentials stored in the Kerberos cache file were used to authenticate to the job manager process.

Debugging problems

If you encounter problems during this step you can trace the problem. In the wsadmin.properties file, uncomment the following line:

```
com.ibm.ws.scripting.traceString=com.ibm.*=all=enabled
```

Then, execute wsadmin.sh again. The output is stored in the following location:

```
<Job_Manager_Profile_Home>/logs/wsadmin.traceout
```

The output shown in Example 17-7 was extracted from wsadmin.traceout file.

Example 17-7 The wsadmin.traceout output

```
Private Credential:
--- GSSCredential ---
        Number of mehanism credentials: 1

[1] Kerberos credential, mechanism: 1.2.840.113554.1.2.2
        Owner:          wasadmin@KDC.ITSO.COM
        Usage:          initiate only
        StartTime:      7/21/09 9:52 AM
        InitLifeTime:   35,690 seconds
        AcceptLifeTime: unknown
        Krb5Client:     wasadmin@KDC.ITSO.COM
        Krb5Server:     krbtgt/KDC.ITSO.COM@KDC.ITSO.COM
--- End of GSSCredential ---

        Private Credential:
```

```
token name:         com.ibm.wsspi.wssecurity.token.krbAuthnToken
version:            1
hashCode:           1181329944
uniqueId:           null
kerberos principal: wasadmin
realm:              KDC.ITSO.COM
expiration:         Tue Jul 21 19:47:41 EDT 2009
renew until:        null
isReadOnly:         false
isAddressless:      false
isForwardable:      true
isRenewable:        false
KerberosTicket:     Ticket (hex) =
```

17.5 Validating the solution

To validate solution, you can run two tests. First, you can create a job to start the application server, server1, using the administrative console. Then, you create a job to stop this same application server using **wsadmin**. Kerberos authentication is used to accomplish this task.

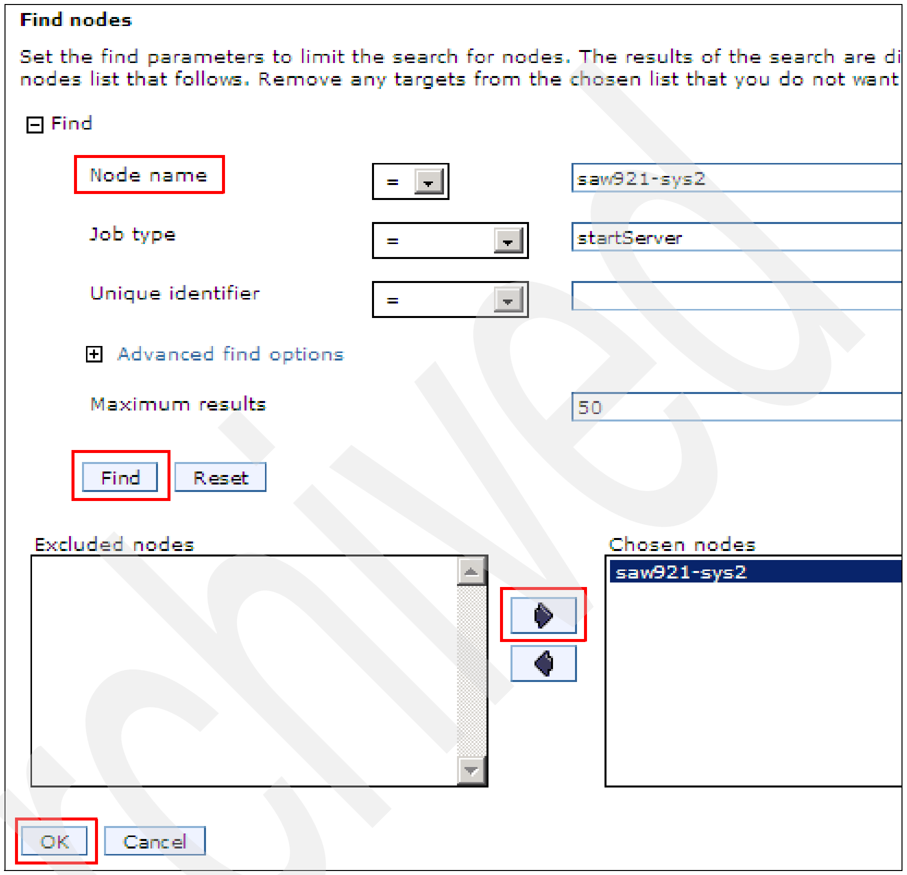
17.5.1 Create a job using the job manager console

In this scenario, server1 and the administrative agent are on sys2. The job manager is on sys3. This first test uses the job manager console. WebSphere Application Server uses the LDAP server to validate the data (user ID and password) for authentication. For this test, you do not use Kerberos.

To create a job using the job manager console:

1. Open the job manager console, and select **Jobs** → **Submit**. In Step 1 Choose a job type, select **Start server** for the Job type parameter, and click **Next**.
2. In Step 2 Choose job targets, select the Node names, and then click **Find** to locate the target node `saw921-sys2`.

3. In the Find nodes panel, click **Find** to display the node name in the Excluded nodes field. Select the node name, and then click to move the node name to the Chosen nodes field, as shown in Figure 17-5. Click **OK**.



Find nodes

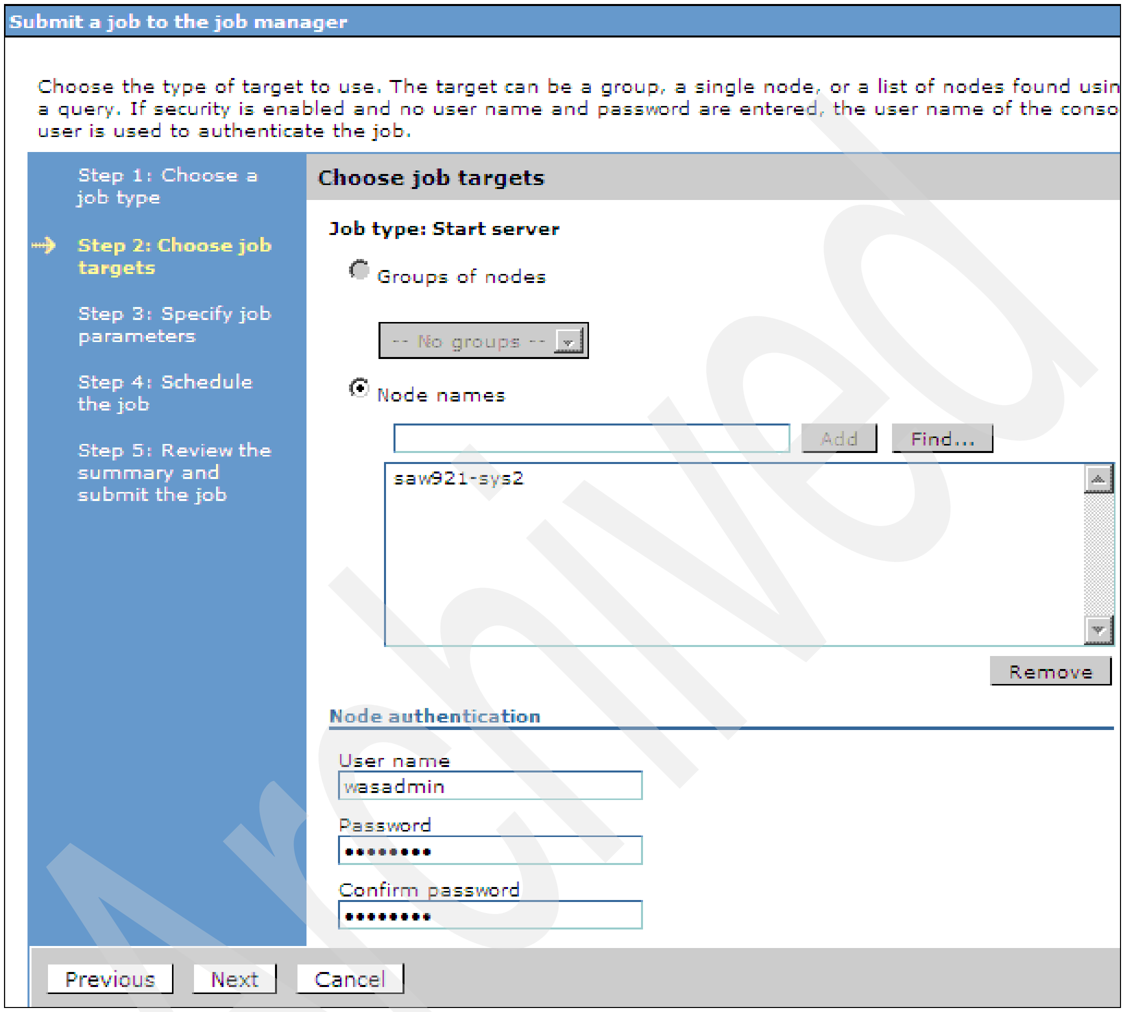
Set the find parameters to limit the search for nodes. The results of the search are di
nodes list that follows. Remove any targets from the chosen list that you do not want

Find

Node name	=	saw921-sys2
Job type	=	startServer
Unique identifier	=	
Advanced find options		
Maximum results		50

Figure 17-5 Find the target node

4. Back in the Step 2 window, shown in Figure 17-6, provide the user name and password for the node authentication parameters, and click **Next**.


Submit a job to the job manager

Choose the type of target to use. The target can be a group, a single node, or a list of nodes found usin a query. If security is enabled and no user name and password are entered, the user name of the conso user is used to authenticate the job.

Step 1: Choose a job type
Step 2: Choose job targets
Step 3: Specify job parameters
Step 4: Schedule the job
Step 5: Review the summary and submit the job

Choose job targets

Job type: Start server

Groups of nodes [-- No groups --]

[✓ Node names]

saw921-sys2

Node authentication

User name: wasadmin

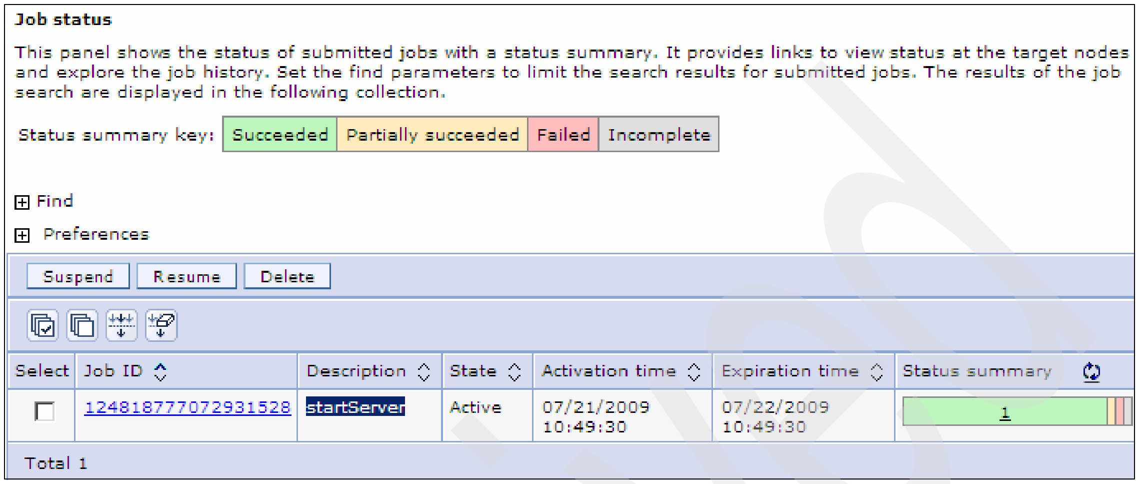
Password: ••••••••

Confirm password: ••••••••

Figure 17-6 Submitting a job-to-job manager

5. For Step 3: Specify job parameters, provide the server name for the stand-alone server, in this case `server1`, and then click **Next.**
6. In Step 4: Schedule the job, leave the default parameters, making the job available now. Click **Next**.
7. A summary is displayed, Click **Finish** to submit the job.

8. Now verify the status of the job by selecting **Jobs** → **Status**. If the job completed successfully, a window indicates the job status is *Succeeded* (green).



Job status

This panel shows the status of submitted jobs with a status summary. It provides links to view status at the target nodes and explore the job history. Set the find parameters to limit the search results for submitted jobs. The results of the job search are displayed in the following collection.

Status summary key: Succeeded | Partially succeeded | Failed | Incomplete

Select	Job ID	Description	State	Activation time	Expiration time	Status summary
☐	124818777072931528	startServer	Active	07/21/2009 10:49:30	07/22/2009 10:49:30	1

Total 1

Figure 17-7 Job status panel

9. Also verify that application server running. Go to the application server system (`saw921-sys2`) and check the `<`*`profile_home`*`>/logs/<`*`server_name`*`>/SystemOut.log` file, shown in Example 17-8.

Example 17-8 server1 SystemOut.log

```
[7/21/09 10:49:22:968 EDT] 0000000e authz         I   CWWIM2000I
Initialization of the authorization component completed successfully.
[7/21/09 10:49:22:984 EDT] 0000000e UserManagemen I   CWWIM6003I
Initialization of the dynamic reload manager completed successfully.
[7/21/09 10:49:23:000 EDT] 00000000 WsServerImpl  A   WSVR0001I: Server
server1 open for e-business
```

17.5.2 Create a job using wsadmin

You can create a job using **wsadmin** as a second test to validate this scenario. A new job to stop this same application server is submitted, but this test uses **wsadmin** and Kerberos authentication to accomplish this task. To create a job using **wsadmin**, follow these steps:

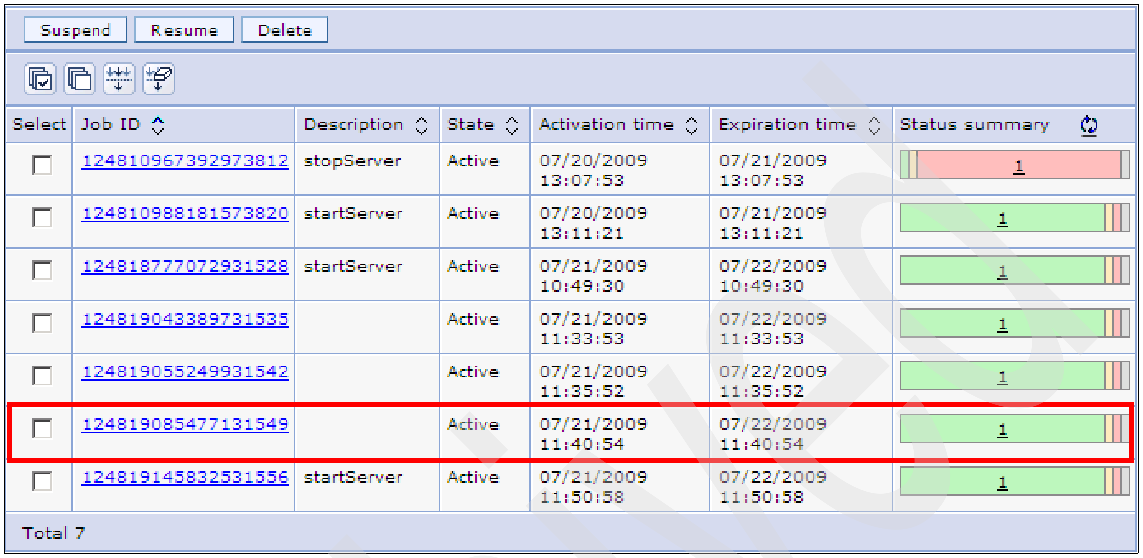
1. Execute **wsadmin** from the <*Job_Manager_Profile_Home*>/bin directory.
2. From the **wsadmin** prompt use the **submitJob** command interactively. The command asks for the parameters to execute the job, as shown in Example 17-9.

Example 17-9 Executing submitJob command

```
wsadmin>$AdminTask submitJob {-interactive}
Submit New Job

Submits a new job for execution.

*Job Type (jobType): stopServer
Target managed node name list (targetList): saw921-sys2
Group Name (group):
Job Parameters (jobParams): {serverName server1}
usernameTitle (username):
passwordTitle (password):
Description (description):
Activation Date Time (activationDateTime):
Expiration Date time (expirationDateTime):
Recurring Interval (executionWindow):
Recurring Interval Unit (executionWindowUnit):
Email Address (email):

Submit New Job

F (Finish)
C (Cancel)

Select [F, C]: [F] F
WASX7278I: Generated command line: $AdminTask submitJob {-jobType
stopServer -targetList {saw921-sys2 } -jobParams {{serverName
server1}}}
124819085477131549
wsadmin>
```

The command provides the job number, in this case 124819085477131549.

3. You can use the job manager console to verify the status for this job. Go into console and navigate to **Jobs** → **Status** and look for the job number, as shown in Figure 17-8.



Suspend | Resume | Delete

Select	Job ID	Description	State	Activation time	Expiration time	Status summary
☐	124810967392973812	stopServer	Active	07/20/2009 13:07:53	07/21/2009 13:07:53	1
☐	124810988181573820	startServer	Active	07/20/2009 13:11:21	07/21/2009 13:11:21	1
☐	124818777072931528	startServer	Active	07/21/2009 10:49:30	07/22/2009 10:49:30	1
☐	124819043389731535		Active	07/21/2009 11:33:53	07/22/2009 11:33:53	1
☐	124819055249931542		Active	07/21/2009 11:35:52	07/22/2009 11:35:52	1
☐	124819085477131549		Active	07/21/2009 11:40:54	07/22/2009 11:40:54	1
☐	124819145832531556	startServer	Active	07/21/2009 11:50:58	07/22/2009 11:50:58	1

Total 7

Figure 17-8 Jobs status panel

Also, view the `SystemOut.log` for the application server and make sure the process is stopped, as shown in Example 17-10.

Example 17-10 SystemOut.log for server1

```
[7/21/09 11:40:41:000 EDT] 0000001b TCPChannel    I   TCPC0002I: TCP
Channel TCPInboundChannel_ipcc.Default_IPC_Connector_Name has stopped
listening on host 127.0.0.1  (IPv4: 127.0.0.1) port 9635.
[7/21/09 11:40:41:062 EDT] 0000001b FailureScopeC A   WTRN0105I: The
transaction service has shutdown successfully with no transactions
requiring recovery.
[7/21/09 11:40:41:078 EDT] 0000001b ServerCollabo A   WSVR0024I: Server
server1 stopped
```

18

Problem determination

This chapter provides tips for troubleshooting problems you might encounter in the Kerberos environment. It includes the following topics:

- General tips for troubleshooting Kerberos issues
- Common problems when setting up Kerberos
- Common problems when configuring KDCs
- Common problems when configuring Kerberos trust realms
- Common problems when using Kerberos authentication mechanisms
- Common problems when using SPNEGO

18.1 General tips for troubleshooting Kerberos issues

This section provides general tips for troubleshooting problems in a Kerberos environment.

18.1.1 Helpful checks to verify the Kerberos setup

You can complete the following checks to ensure that the Kerberos environment is set up correctly:

1. Verify that the Kerberos keytab file is valid with the correct encryption. Run the following command:

   ```
   <JAVA_HOME>/jre/bin/java
   com.ibm.security.krb5.internal.tools.Klist -e –k <keytab file>
   ```

   Example 18-1 shows an example of the correct output.

   Example 18-1 Sample Klist output

   ```
   ./java com.ibm.security.krb5.internal.tools.Klist -e -k
   /tmp/krb5.keytab

   Key table: /tmp/krb5.keytab
   Number of entries: 1

   [1] principal: WAS/washost.test.com@KDC.TEST.COM
           KVNO: 3

           Encryption type: DES CBC mode with MD5
   ```

2. Verify that the Kerberos configuration file (`krb5.ini` for Windows and `krb5.conf` for other platforms) points to the keytab. Run the following command:

   ```
   <JAVA_HOME>/jre/bin/java -Djava.security.krb5.conf=<krb5 file>
   com.ibm.security.krb5.internal.tools.Ktab
   ```

   Example 18-2 shows an example of the correct output.

   Example 18-2 Sample Ktab output

   ```
   ./java -Djava.security.krb5.conf=/tmp/krb5.conf
   com.ibm.security.krb5.internal.tools.Ktab
   1 entries in keytab, name: /tmp/krb5.keytab
           KVNO    Principal
           ----    ---------
   3 WAS/washost.test.com@KDC.TEST.COM
   ```

3. Verify that the Kerberos configuration file points correctly to the KDC. Run the following command:

   ```
   <JAVA_HOME>/jre/bin/java -Djava.security.krb5.conf=<krb5 file>
   com.ibm.security.krb5.internal.tools.Kinit <User or SPN>
   ```

   When you run the command, it prompts you for the password. Example 18-3 shows an example of the correct output.

   Example 18-3 Getting a TGT using Kinit using password

   ```
   ./java -Djava.security.krb5.conf=/tmp/krb5.conf
   com.ibm.security.krb5.internal.tools.Kinit
   WAS/washost.test.com@KDC.TEST.COM
   Password for WAS/washost.test.com@KDC.TEST.COM

   Done!
   New ticket is stored in cache file /root/krb5cc_root
   ```

4. If only one key is inside the keytab, verify that the Kerberos configuration file and Kerberos keytab are valid using the following command:

   ```
   <JAVA_HOME>/jre/bin/java -Djava.security.krb5.conf=<krb5 file>
   com.ibm.security.krb5.internal.tools.Kinit -k
   ```

   When you run the command, it prompts you for the password. Example 18-4 shows an example of the correct output.

   Example 18-4 Getting a TGT using kinit using keytab file

   ```
   ./java -Djava.security.krb5.conf=/tmp/krb5.conf
   com.ibm.security.krb5.internal.tools.Kinit -k
   Done!
   New ticket is stored in cache file /root/krb5cc_root
   ```

18.1.2 Enabling traces in WebSphere Application Server

If you encounter problems enabling SPNEGO or Kerberos in WebSphere Application Server, you can diagnose the failure by enabling traces as described in this section.

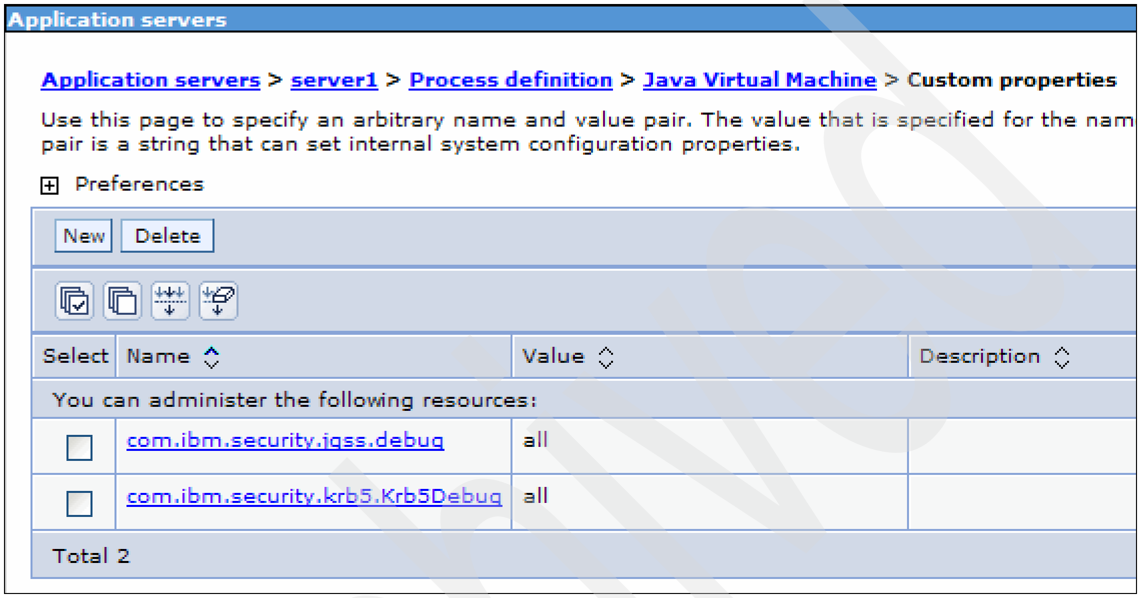
Enabling the JGSS and KRB traces

To enable the Java Generic Security Services (JGSS) and Kerberos (KRB) traces, follow these steps:

1. In the WebSphere administration console navigation tree, click **System administration** → **Deployment manager**.

2. In the deployment manager page, select **Java and Process Management** → **Process definition** → **Java Virtual Machine** → **Custom Properties**.
3. Set the following properties, as shown in the Figure 18-1:

```
com.ibm.security.jgss.debug to all
com.ibm.security.krb5.Krb5Debug to all
```



Application servers

Application servers > server1 > Process definition > Java Virtual Machine > Custom properties

Use this page to specify an arbitrary name and value pair. The value that is specified for the nam
pair is a string that can set internal system configuration properties.

Select	Name	Value	Description
You can administer the following resources:			
☐	com.ibm.security.jgss.debug	all	
☐	com.ibm.security.krb5.Krb5Debug	all	
Total 2			

Figure 18-1 Enable JGSS and KRB traces

4. Click **OK**, and click **Save** changes.
5. Ensure that the nodes are synchronized and restart the environment (application servers, node agents, and deployment manager).

Important: Remember to disable Kerberos debugging after validating the authentication because it logs sensitive information and can have a negative impact on application server performance.

You can also use the following command line options:

```
-Dcom.ibm.security.jgss.debug=all
-Dcom.ibm.security.krb5.Krb5Debug=all
```

These options are the parameters passed to `java.exe` when WebSphere Application Server is invoked. You can use them when starting the server or starting the Java Thin Client application.

WSS trace option

The following command line option for `java.exe` allows you to trace WSS:

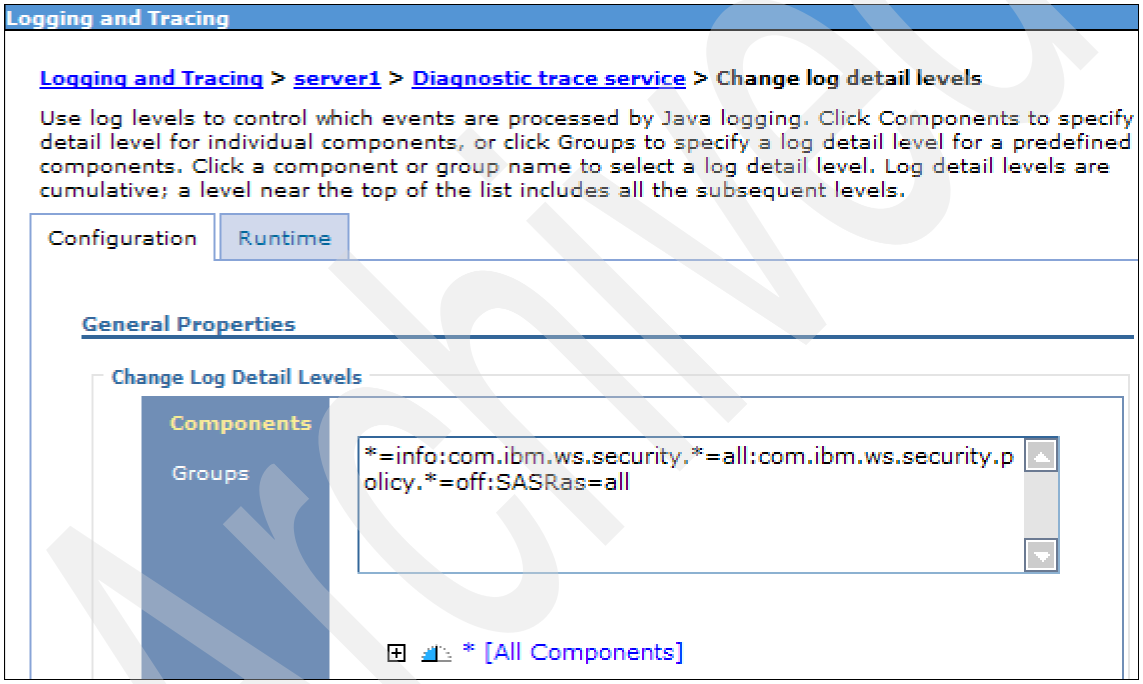
```
-Dcom.ibm.security.ktp.debug=all
```

This option does not exist by default. You can add it as a property and set to `all` manually, if WSS needs to be traced.

WebSphere security trace option

WebSphere security traces can also be enabled, using the following trace string as shown in Figure 18-2:

```
com.ibm.ws.security.*=all:com.ibm.ws.security.policy.*=off:SASRas=all
```


Logging and Tracing

Logging and Tracing > server1 > Diagnostic trace service > Change log detail levels

Use log levels to control which events are processed by Java logging. Click Components to specify detail level for individual components, or click Groups to specify a log detail level for a predefined components. Click a component or group name to select a log detail level. Log detail levels are cumulative; a level near the top of the list includes all the subsequent levels.

```
*=info:com.ibm.ws.security.*=all:com.ibm.ws.security.p
olicy.*=off:SASRas=all
```

Figure 18-2 Enable WebSphere Security trace

18.1.3 Time management

Having synchronized time among the systems is critical for Kerberos to work. Kerberos tickets are valid only for a *time window*. The default time window is 5 minutes. If time is not synchronized, you will get a `Clock Skew too great` message as shown in Example 18-5.

The messages are issued in the `SystemOut.log` when JGSS and KRB traces are enabled. The error can happen during WebSphere Application Server startup or when the request is handled during runtime. Depending upon the Kerberos environment, the messages can vary.

Example 18-5 Clock Skew error output

```
[KRB_DBG_KDC] KRBError:main: >>>KRBError:
[KRB_DBG_KDC] KRBError:main:          sTime is Fri Mar 13 13:51:10
EDT 2009 1236966670000
[KRB_DBG_KDC] KRBError:main:          suSec is 796826
[KRB_DBG_KDC] KRBError:main:          error code is 37
[KRB_DBG_KDC] KRBError:main:          realm is KDC.IBM.COM
[KRB_DBG_KDC] KRBError:main:          sname is krbtgt/KDC.IBM.COM
com.ibm.security.krb5.KrbException, status code: 37
        message: Clock skew too great
```

You can use Network Time Protocol (NTP) to keep the systems using common time. Microsoft Active Directory domain controllers provide NTP daemons to the domain. If WebSphere Application Server is not running on a Windows domain system, you must put measures in place to keep WebSphere Application Server synchronized with the KDC and the client systems.

18.1.4 Useful links

SPNEGO Troubleshooting tips are available using the following link:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rsec_SPNEGO_troubles.html`

Web services troubleshooting tips are available using the following link:

`http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=/com.ibm.websphere.base.doc/info/aes/ae/rwbs_troubleshoot.html`

The following link has additional problem determination information involving DB2 and Kerberos:

http://www.ibm.com/developerworks/data/library/techarticle/dm-0603see/index.html#db2kerb

18.2 Common problems when setting up Kerberos

This section lists common problems that you might encounter when setting up Kerberos.

18.2.1 Message: “Server principal is not found in security registry”

This error message is issued in `SystemOut.log` as shown in Example 18-6.

Example 18-6 Sample SystemOut.log output

```
[KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
>>>KRBError:
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
cTime is Fri Feb 06 05:51:01 GMT 2009 1233899461000
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
sTime is Fri Feb 06 05:51:01 GMT 2009 1233899461000
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
suSec is 957935
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
error code is 7
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
crealm is KDC.IBM.COM
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
cname is user8
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
realm is KDC.IBM.COM
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
sname is WAS/TEST.COM
 [KRB_DBG_KDC] KRBError:WebSphere:ORB.thread.pool t=007c1e88:
etext is Server principal is not found in security registry
 com.ibm.security.krb5.KrbException, status code: 7
   message: Server principal is not found in security registry
   at com.ibm.security.krb5.KrbTgsRep.<init>(KrbTgsRep.java:43)
   at com.ibm.security.krb5.KrbTgsReq.getReply(KrbTgsReq.java:47)
```

Resolution

The problem is usually a configuration problem. Use the information from the messages to compare to the Kerberos configuration.

- Check that the SPN is as expected:
  - Use fully qualified host name of the system and DNS
  - Beware of case sensitivity (`TEST.COM` is not `test.com`)
  - Review WebSphere Application Server global security configuration settings
- Perform the Kerberos setup verification steps listed in “Helpful checks to verify the Kerberos setup” on page 468.

18.2.2 Message: “KDC has no support for encryption type”

This error message is issued in `SystemOut.log` as shown in Example 18-7. The encryption type is listed in the message.

Example 18-7 Example SystemOut.log output

```
[KRB_DBG_AS] KrbAsReq:main: request etypes:
[1] aes128-cts-hmac-sha1-96
…
com.ibm.security.krb5.KrbException, status code: 14
        message: KDC has no support for encryption type
        at com.ibm.security.krb5.KrbAsRep.<init>(KrbAsRep.java:30)
        at com.ibm.security.krb5.KrbAsReq.getReply(KrbAsReq.java:77)
        at com.ibm.security.krb5.KrbAsReq.getReply(KrbAsReq.java:268)
        at com.ibm.security.krb5.internal.tools.Kinit.b(Kinit.java:52)
        at
com.ibm.security.krb5.internal.tools.Kinit.<init>(Kinit.java:56)
        at
com.ibm.security.krb5.internal.tools.Kinit.main(Kinit.java:74)
com.ibm.security.krb5.KrbException, status code: 14
        message: KDC has no support for encryption type
```

Resolution

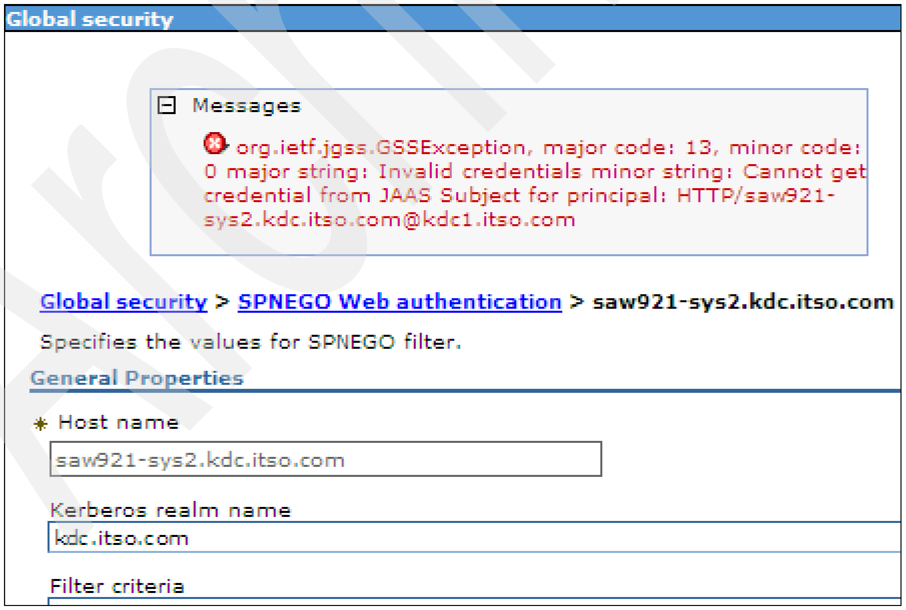
Review the following configuration items to ensure that the encryption type is supported:

- Review the encryption types configured for support on the KDC system
  - Encryption types enabled on the KDC
  - Client user configured encryptions
  - SPN user configured encryptions
  - The krbtgt principal configured encryptions
- Review the encryption types configured for support on the client or server system. You can find the encryption types in the Kerberos configuration file (`krb5.ini` or `krb5.conf`).

18.2.3 Message "Cannot get credential from JAAS Subject for principal"

The following error message is issued when configuring WebSphere Application Server as shown in the Figure 18-3.

```
org.ietf.jgss.GSSException, major code: 13, minor code: 0 major
string: Invalid credentials minor string: Cannot get credential from
JAAS Subject for principal <principal name>
```



Global security

Messages

org.ietf.jgss.GSSException, major code: 13, minor code: 0 major string: Invalid credentials minor string: Cannot get credential from JAAS Subject for principal: HTTP/saw921-sys2.kdc.itso.com@kdc1.itso.com

Global security > SPNEGO Web authentication > saw921-sys2.kdc.itso.com

Specifies the values for SPNEGO filter.

General Properties

* Host name: saw921-sys2.kdc.itso.com

Kerberos realm name: kdc.itso.com

Filter criteria

Figure 18-3 Cannot get credential from JAAS Subject for principal

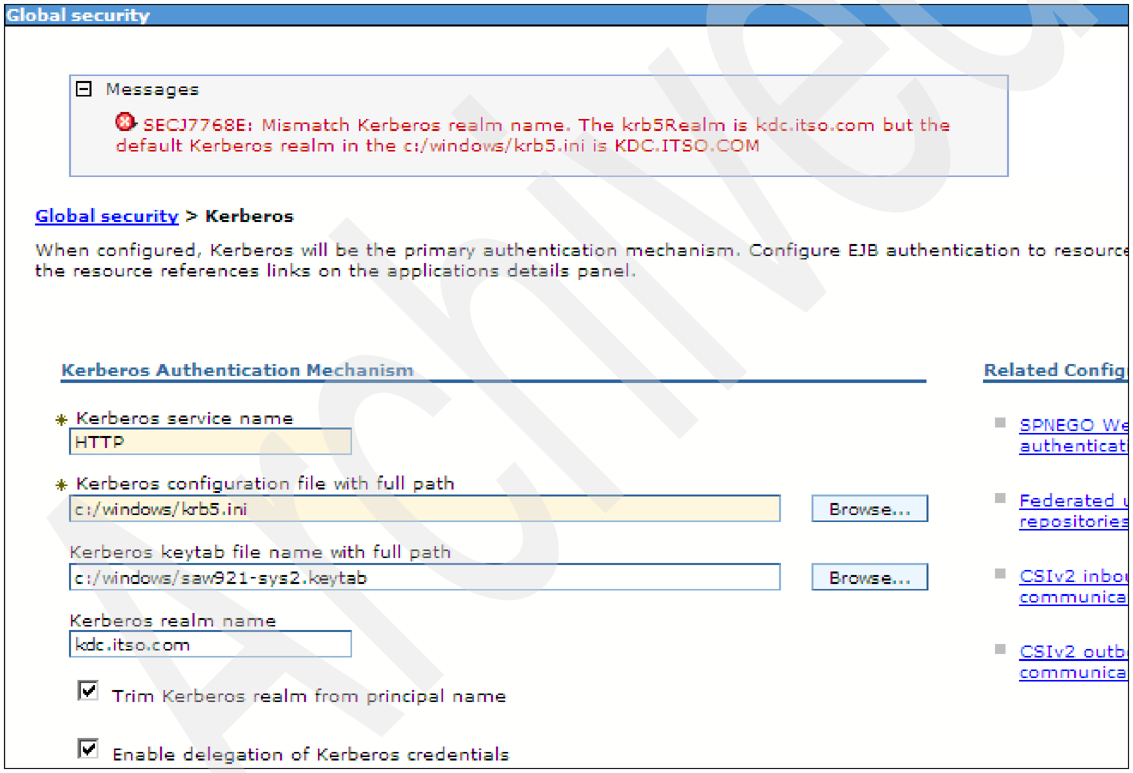
Resolution

Make sure that the Kerberos realm name and the keytab file name and path are defined correctly. In this case, the Kerberos realm name should be `KDC.ITSO.COM` instead of `kdc.itso.com` (case sensitive).

18.2.4 Message: "SECJ7768E: Mismatch Kerberos realm name"

The following error message is issued when you configure WebSphere Application Server for Kerberos support as shown in Figure 18-4:

```
"SECJ7768E: Mismatch Kerberos realm name. The krb5Realm is
kdc.itso.com but the default Kerberos realm in the /etc/krb5.conf is
KDC.ITSO.COM"
```



SECJ7768E: Mismatch Kerberos realm name. The krb5Realm is kdc.itso.com but the default Kerberos realm in the c:/windows/krb5.ini is KDC.ITSO.COM

Global security > Kerberos

When configured, Kerberos will be the primary authentication mechanism. Configure EJB authentication to resourc
the resource references links on the applications details panel.

Kerberos Authentication Mechanism

* Kerberos service name: HTTP

* Kerberos configuration file with full path: c:/windows/krb5.ini

Kerberos keytab file name with full path: c:/windows/saw921-sys2.keytab

Kerberos realm name: kdc.itso.com

[✓] Trim Kerberos realm from principal name

[✓] Enable delegation of Kerberos credentials

Figure 18-4 SECJ7768E: Mismatch Kerberos realm name

Resolution

You can resolve the problem by making sure that the Kerberos realm name matches the realm name that is defined in the Kerberos configuration file (`krb5.conf` or `krb5.conf`).

18.2.5 Message: “Cannot get credential for principal service”

The following error message is issued when you configure Kerberos using WebSphere Application Server administrative console (**Global Security** → **Kerberos**):

```
"org.ietf.jgss.GSSException, major code: 11, minor code: 0 major
string: General failure, unspecified at GSSAPI level minor string:
Cannot get credential for principal service
HTTP/9.42.170.217@42.170.217"
```

Resolution

The principal service must be in the following format:

```
<service name>/<fully qualified hostname>@KerberosRealm
```

The exception happens when the fully qualified host name is not specified and it was not found from the `/etc/hosts` file or the DNS server.

On UNIX or Linux systems, if the `hosts` line in the `/etc/nsswitch.conf` file is configured to first look in the `hosts` file before it looks in DNS, the Kerberos configuration might fail if the hosts file contains an entry for the system that is not the fully qualified host name.

You can resolve the problem by updating the `/etc/hosts` file to contain a fully qualified host name for the system.

18.3 Common problems when configuring KDCs

This section lists common problems that are typically encountered when configuring KDCs.

18.3.1 Unable to start AIX KDC daemon

The AIX KDC daemon does not start. It displays the message `Unable to load the plug-in object or library`.

Resolution

The error message implies that the AIX KDC daemon cannot find the LDAP plug-in library. You can resolve this by setting the LIBPATH environment variable, so that the library path is set up correctly.

Run the command shown in Example 18-8, and restart the Kerberos daemons.

Example 18-8 Resolving the library path issue for AIX KDC

```
[saw921-sys6][/] #export
LIBPATH=$LIBPATH:/opt/IBM/ldap/V6.1/lib:/opt/IBM/ldap/V6.1/lib64
```

Tip: It is recommended that you add the command shown in the Example 18-8 to the shell profile file, so that the LIBPATH is correctly set even after the system is rebooted.

18.3.2 The config.krb5 command is unable to configure a KDC on AIX

The **`config.krb5`** command fails with the error message shown in Example 18-9 when configuring KDC on AIX.

Example 18-9 Error when another Kerberos configuration already exist

```
[saw921-sys6][/]# /usr/krb5/sbin/config.krb5 -S -d itso.ral.ibm.com -r
ITSO.RAL.IBM.COM
Initializing configuration...
/etc/krb5/krb5_cfg_type already exists.
Configuration cannot be completed.
A previous configuration exists.
To remove the existing configuration, run the unconfig.krb5 command.
[saw921-sys6][/]#
```

Resolution

If there is an existing Kerberos configuration on AIX, the **`config.krb5`** command will fail. It can be resolved by running the **`/usr/krb5/sbin/unconfig.krb5`** command to remove the existing configuration.

The **`unconfig.krb5`** command removes the Kerberos principal database (existing) and all the Kerberos configuration files from the system.

18.3.3 Message: “GSS-API error” when running kinit or kadmin

This error message is issued when running the **kinit** or **kadmin** command.

Resolution

The error message is more likely to happen when time is not synchronized between the KDC system and the Kerberos client system.

Make sure that both the systems are within the time difference of plus or minus 5 minutes. Note that Kerberos is independent of time zones or DST settings, so when you synchronize, use the same time zone across all the systems.

If the error message is still getting issued, you might need to enable trace on the KDC to debug the failing scenario.

18.3.4 Message: “Decrypt integrity check failed” when running kinit or kadmin

This message is issued when running the command **kinit** or **kadmin**.

Resolution

The message displays when the entered password does not match the password expected by the KDC. Make sure that correct password is entered.

18.3.5 Message: “DSA is unwilling to perform” is issued when loading the Kerberos schema

This message is issued when loading the Kerberos schema to Tivoli Directory Server as shown in Example 18-10.

Example 18-10 LDAP error when trying to load IBM NAS Kerberos schema

```
[saw921-sys6][/usr/krb5/ldif]# ldapmodify -h sys4.itso.ral.ibm.com -D
cn=root -w adminpwd01 -f ./IBM.KRB.schema.ldif -v
ldap_init(sys4.itso.ral.ibm.com, 389)
add attributetypes:
        BINARY (85 bytes) ( 1.2.840.113556.1.4.49 NAME
'badPasswordTime' SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )
add ibmattributetypes:
       BINARY (67 bytes) ( 1.2.840.113556.1.4.49 DBNAME( 'badPwdTime'
'badPasswordTime' ) )
Operation 0 modifying entry cn=schema
ldap_modify: DSA is unwilling to perform
```

Resolution

The message is issued when the KDC is not able to communicate with the Tivoli Directory Server (LDAP server). Make sure that Tivoli Directory Server is not running in configuration only mode. If so, stop the server and try to start in normal mode. You can use the **idsldapsearch** command as shown in Example 18-11 to check whether Tivoli Directory Server is running in configuration mode in windows.

Example 18-11 Check the configuration mode for the Tivoli Directory Server

```
C:\Documents and Settings\Administrator>idsldapsearch -s base -b " "
objectclass=* ibm-slapdisconfigurationmode
ibm-slapdisconfigurationmode=TRUE
```

If the Tivoli Directory Server server is running in configuration mode, then stop it and start Tivoli Directory Server from the command line as shown in Example 18-12. Check whether any error messages get issued to the display.

Example 18-12 Starting IBM Tivoli Directory Server from the command line

```
C:\Documents and Settings\Administrator>idsslapd -I idsinst -n
GLPSRV041I Server starting.
GLPCOM024I The extended Operation plugin is successfully loaded from
libevent.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libtranext.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libldaprepl.dll.
GLPSRV155I The DIGEST-MD5 SASL Bind mechanism is enabled in the
configuration file.
GLPCOM021I The preoperation plugin is successfully loaded from
libDigest.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libevent.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libtranext.dll.
GLPCOM023I The postoperation plugin is successfully loaded from
libpsearch.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libpsearch.dll.
GLPCOM025I The audit plugin is successfully loaded from
C:/LDAP/V6.1/lib/libldapaudit.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libevent.dll.
GLPCOM023I The postoperation plugin is successfully loaded from
libpsearch.dll.
```

```
GLPCOM024I The extended Operation plugin is successfully loaded from
libpsearch.dll.
GLPCOM022I The database plugin is successfully loaded from
C:/LDAP/V6.1/lib/libback-config.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libevent.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libtranext.dll.
GLPCOM023I The postoperation plugin is successfully loaded from
libpsearch.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libpsearch.dll.
GLPCOM022I The database plugin is successfully loaded from
C:/LDAP/V6.1/lib/libback-rdbm.dll.
GLPCOM010I Replication plugin is successfully loaded from
C:/LDAP/V6.1/lib/libldaprepl.dll.
GLPCOM021I The preoperation plugin is successfully loaded from
libpta.dll.
GLPCOM016E Logon failure: the specified account password has expired.
GLPCOM014E Failed to log on user: idsinst.
GLPRDB089E The directory server is unable to log on as the database
administrator.
GLPSRV064E Failed to initialize be_config.
GLPSRV015I Server configured to use 636 as the secure port.
GLPCOM024I The extended Operation plugin is successfully loaded from
libloga.dll.
GLPCOM024I The extended Operation plugin is successfully loaded from
libidsfget.dll.
GLPSRV180I Pass-through authentication is disabled.
```

The messages indicate that the password of the Tivoli Directory Server instance owner user account `'idsinst'` is expired. The problem is resolved by resetting the password and starting Tivoli Directory Server again.

18.3.6 Message: “Client not found in Network Authentication Service database or client locked out” when running kinit

As shown in Example 18-13, this message is issued when running the **kinit** command to get a Kerberos TGT and the Tivoli Directory Server is used as the LDAP database.

Example 18-13 Error while running kinit

```
[saw921-sys6][/]# kinit admin/admin
Unable to obtain initial credentials.
        Status 0x96c73a06 - Client not found in Network Authentication
Service database or client locked out.
```

Resolution

The message is issued when either the Kerberos principal information cannot be read from the Kerberos database or the Kerberos principal is locked out due to continuous incorrect logon attempts. (This feature is turned off by default.)

The command has failed because the Tivoli Directory Server instance was stopped. Thus, the Kerberos principal information could not be found. The problem is fixed by starting the Tivoli Directory Server instance.

18.4 Common problems when configuring Kerberos trust realms

This section lists common problems that are typically encountered when configuring Kerberos trust realms.

18.4.1 Message: “Key version is not available” getting service ticket from cross-realm KDC

This message is issued when getting credential from cross-realm KDC, as shown in Example 18-14. This error typically occurs while getting a service ticket for a z/OS service principal from an AIX KDC.

Example 18-14 Error while getting a cross-realm service ticket

```
[saw921-sys6][/]# kvno krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
Unable to get credentials. krbtgt/ITSO.IBM.COM@ITSO.IBM.COM
Status 0x96c73a2c - Key version is not available.
```

Resolution

The error message in this example is issued when z/OS KDC cannot find the key requested by the AIX KDC for the principal `krbtgt/ITSO.IBM.COM@ITSO.IBM.COM`.

You can resolve this problem by making sure that the key version number and encryption types of all the cross-realm TGT principals are the same in both realms. Also, make sure that both the systems are within the time difference of plus or minus 5 minutes. Note that the Kerberos is independent of time zones or DST settings, so when synchronizing, use the same time zone across all the systems.

If you still get the same message, then you might need to enable tracing on the KDC to find the exact reason.

18.5 Common problems when using Kerberos authentication mechanisms

This section lists common problems that are typically encountered when using Kerberos authentication mechanisms.

18.5.1 Message: "CWWIM4512E The password match failed" when logging on to the WebSphere administrative console

This message is issued when you are unable to logon to the WebSphere Application Server administrative console. Example 18-15 shows the exception seen in `SystemOut.log`.

Example 18-15 Example SystemOut.log output

```
[7/9/09 0:25:20:471 EDT] 00000013 SystemOut     O [KRB_DBG_AS] KrbAsReq:WebContainer
: 2: request etypes:
[1] des3-cbc-sha1[2] rc4-hmac[3] des-cbc-crc[4] des-cbc-md5[5] des-cbc-crc
[7/9/09 0:25:20:495 EDT] 00000013 FfdcProvider  I com.ibm.ws.ffdc.impl.FfdcProvider
logIncident FFDC1003I: FFDC Incident emitted on
/usr/IBM/WebSphere/AppServer/profiles/scn3Dmgr01/logs/ffdc/dmgr_7cbc7cbc_09.07.09_00.
25.20.4762288066913221754620.txt
com.ibm.ws.security.auth.kerberos.Krb5LoginModuleWrapper.login 552
[7/9/09 0:25:21:054 EDT] 00000013 exception     W
com.ibm.ws.wim.adapter.file.was.FileAdapter login CWWIM4512E The password match
failed.
[7/9/09 0:25:21:097 EDT] 00000013 exception     W
com.ibm.ws.wim.adapter.file.was.FileAdapter login
com.ibm.websphere.wim.exception.PasswordCheckFailedException: CWWIM4512E The password
match failed.
        at com.ibm.ws.wim.adapter.file.was.FileAdapter.login(FileAdapter.java:2009)
        at com.ibm.ws.wim.ProfileManager.loginImpl(ProfileManager.java:3351)
        at
com.ibm.ws.wim.ProfileManager.genericProfileManagerMethod(ProfileManager.java:277)
        at com.ibm.ws.wim.ProfileManager.login(ProfileManager.java:381)
        at com.ibm.websphere.wim.ServiceProvider.login(ServiceProvider.java:485)
        at
com.ibm.ws.wim.registry.util.LoginBridge.checkPassword(LoginBridge.java:169)
        at com.ibm.ws.wim.registry.WIMUserRegistry$1.run(WIMUserRegistry.java:181)
        at
com.ibm.ws.security.auth.ContextManagerImpl.runAs(ContextManagerImpl.java:4565)
        at
com.ibm.ws.security.auth.ContextManagerImpl.runAsSystem(ContextManagerImpl.java:4653)
```

Resolution

You can resolve this problem by updating the Kerberos configuration file (`krb5.ini` or `krb5.conf`) to use only one encoding type as shown in Example 18-16.

Example 18-16 Example Kerberos configuration file

```
...
...
default_tkt_enctypes = arcfour-hmac
default_tgs_enctypes = arcfour-hmac
...
...
```

If multiple encoding is required, disable Kerberos pre-authentication for the corresponding Kerberos principal.

18.5.2 db2start fails when processing the IBMkrb5 plugin

When you start DB2 using the **`db2start`** command, it fails with the error message shown in Example 18-17.

Example 18-17 Example command output

```
bash-2.05b$ db2start
SQL1365N  db2start or db2stop failed in processing the plugin
"IBMkrb5". Reason code = "10".
```

```
07/16/2009 07:32:43     0   0   SQL1365N  db2start or db2stop failed in
processing the plugin "". Reason code = "".
SQL1032N  No start database manager command was issued.  SQLSTATE=57019
```

Resolution

You can resolve this problem by ensuring that the DB2 instance owner (`db2inst1`) has read access to the keytab file `krb5.keytab`. Go to the `/etc/krb5` directory and issuing the **`chmod a+r krb5.keytab`** command.

18.5.3 Message: "com.ibm.security.krb5.KrbException, status code: 7 message: Server not found in Kerberos database" during wsadmin authentication

This message is issued in `wsadmin.traceout` during **`wsadmin`** authentication as shown in Example 18-18.

Example 18-18 Example SystemOut.log output

```
[7/20/09 15:18:36:906 EDT] 00000000 Krb5WSSecurit E   SECJ9314E: An
unexpected exception occurred when trying to run initSecContext()
method : GSSException: org.ietf.jgss.GS
SException, major code: 11, minor code: 0
 major string: General failure, unspecified at GSSAPI level
 minor string: Error: java.lang.Exception: Error:
com.ibm.security.krb5.KrbException, status code: 7
 message: Server not found in Kerberos database
```

Resolution

This exception happens when the principal is not retrievable from the keytab file. Perform the verification steps listed in "Helpful checks to verify the Kerberos setup" on page 468.

18.5.4 Message: “com.ibm.websphere.security.auth.WSLoginFailedException: Login Failure: all modules ignored” when starting wsadmin

This exception is issued when starting **`wsadmin`** as shown in Example 18-19. WebSphere Application Server cannot authenticate the user for starting `wsadmin.sh` using Kerberos credentials.

Example 18-19 Example command output

```
[root@saw921-sys3 bin]# ./wsadmin.sh
WASX7023E: Error creating "SOAP" connection to host "localhost";
exception information:
com.ibm.websphere.security.auth.WSLoginFailedException: Login Failure:
all modules ignored
WASX7213I: This scripting client is not connected to a server process;
please refer to the log file
/opt/WebSphere/AppServer/profiles/JobMgr01/logs/wsadmin.traceout for
additional information.
WASX8011W: AdminTask object is not available.
WASX7029I: For help, enter: "$Help help"
wsadmin>quit
[root@saw921-sys3 bin]#
```

Resolution

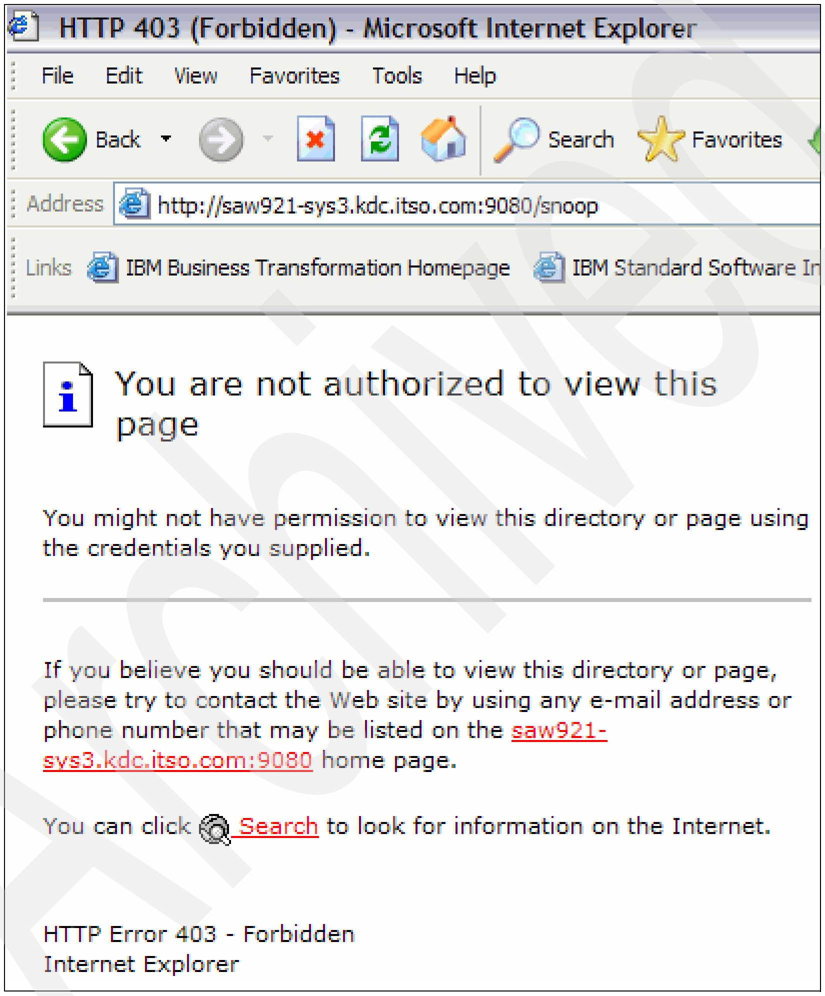
This exception happens when Kerberos credentials are not available. Perform the Kerberos verification steps listed in “Helpful checks to verify the Kerberos setup” on page 468.

18.6 Common problems when using SPNEGO

This section lists common problems that are typically encountered when using SPNEGO.

18.6.1 Message: "CWWIM4538E Multiple principals were found for the 'gcezar' principal name" is issued when using SPNEGO authentication

When SPNEGO is enabled, the browser returns an HTTP response code 403 when you attempt to access an application, as shown in Figure 18-5.



HTTP 403 (Forbidden) - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
Links
IBM Business Transformation Homepage
IBM Standard Software In

http://saw921-sys3.kdc.itso.com:9080/snoop

You are not authorized to view this page

You might not have permission to view this directory or page using the credentials you supplied.

If you believe you should be able to view this directory or page, please try to contact the Web site by using any e-mail address or phone number that may be listed on the saw921-sys3.kdc.itso.com:9080 home page.

You can click Search to look for information on the Internet.

HTTP Error 403 - Forbidden
Internet Explorer

Figure 18-5 Unable to access snoop application when SPNEGO is enabled

Also the following error message is found in `Systemout.log` as shown in Example 18-20.

Example 18-20 Exception error

```
[7/10/09 13:41:28:031 EDT] 00000019 exception     E
com.ibm.ws.wim.ProfileManager loginImpl CWWIM4538E  Multiple principals
were found for the 'gcezar' principal name.
```

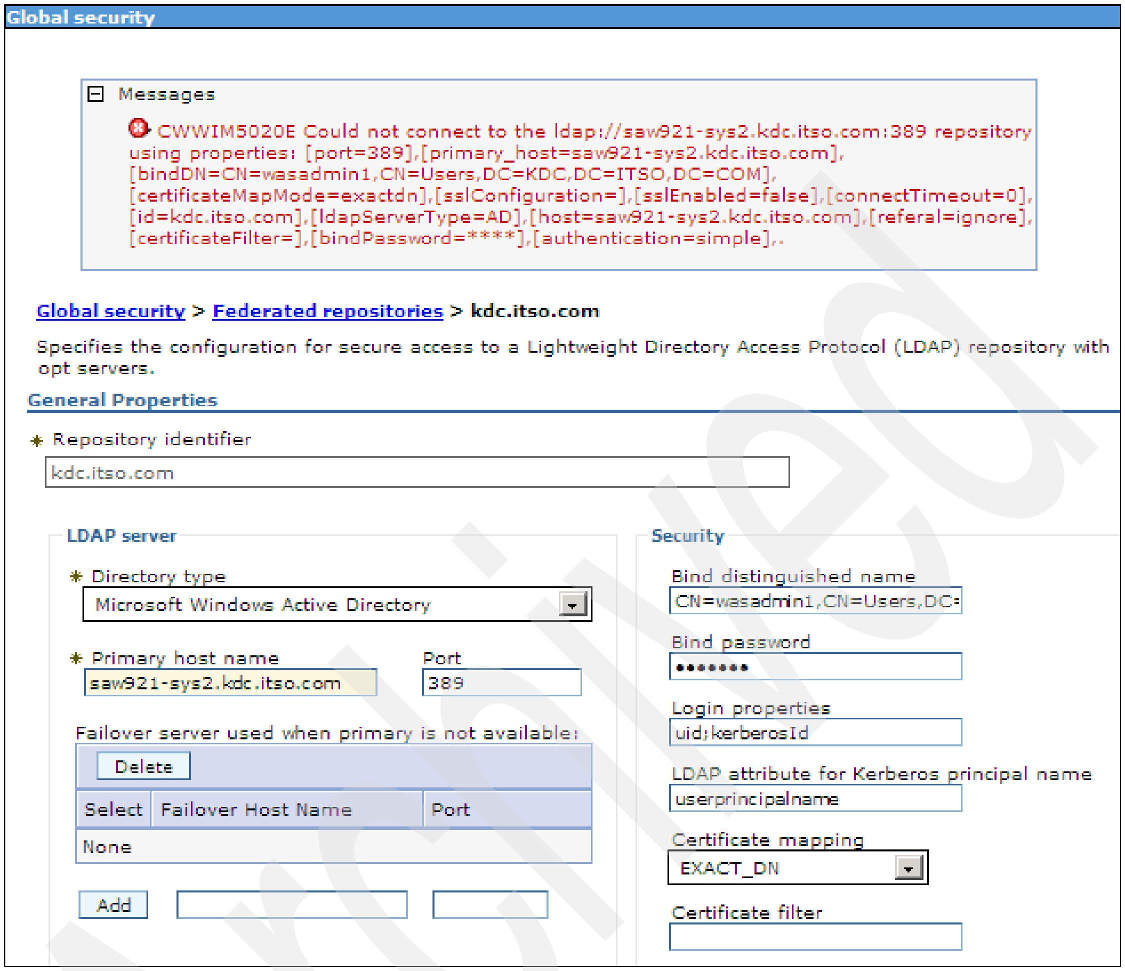
Resolution

You can resolve this problem by making sure that the client user ID is unique in the realms that you have defined in your federated repositories configuration.

18.6.2 Message: "CWWIM5020E Could not connect to the LDAP repository using properties" when configuring the LDAP server

The following error message is issued when configuring the LDAP server as shown in Figure 18-6 on page 489:

```
CWWIM5020E Could not connect to the
ldap://saw921-sys2.kdc.itso.com:389 repository using properties:
[port=389],[primary_host=saw921-sys2.kdc.itso.com],
[bindDN=CN=Users,DC=kdc,DC=itso,DC=com],[certificateMapMode=exactdn]
,[sslConfiguration=],[sslEnabled=false],[connectTimeout=0],[id=kdc.i
tso.com],[ldapServerType=AD],[host=saw921-sys2.kdc.itso.com],[refera
l=ignore],[certificateFilter=],[bindPassword=****],[authentication=s
imple],. "
```


Global security

Messages

CWWIM5020E Could not connect to the ldap://saw921-sys2.kdc.itso.com:389 repository using properties: [port=389],[primary_host=saw921-sys2.kdc.itso.com],[bindDN=CN=wasadmin1,CN=Users,DC=KDC,DC=ITSO,DC=COM],[certificateMapMode=exactdn],[sslConfiguration=],[sslEnabled=false],[connectTimeout=0],[id=kdc.itso.com],[ldapServerType=AD],[host=saw921-sys2.kdc.itso.com],[referal=ignore],[certificateFilter=],[bindPassword=****],[authentication=simple],.

Global security > Federated repositories > kdc.itso.com

Specifies the configuration for secure access to a Lightweight Directory Access Protocol (LDAP) repository with opt servers.

General Properties

* Repository identifier: kdc.itso.com

LDAP server

* Directory type: Microsoft Windows Active Directory

* Primary host name: saw921-sys2.kdc.itso.com

Port: 389

Failover server used when primary is not available:

Select	Failover Host Name	Port
None		

Security

Bind distinguished name: CN=wasadmin1,CN=Users,DC=

Bind password: •••••••

Login properties: uid;kerberosId

LDAP attribute for Kerberos principal name: userprincipalname

Certificate mapping: EXACT_DN

Certificate filter:

Figure 18-6 Could not connect to the LDAP

Resolution

You can resolve this problem by ensuring that the “Bind distinguished name” is configured properly and that the properties shown in the error message are valid for the LDAP server.

A

JAAS custom mapping login module source code

This appendix provides the Java source code for the JAAS custom mapping login module that we use in Chapter 14, “Single sign-on from a Java thin client with AIX and z/OS Kerberos trusted realms” on page 347. This login module maps a Kerberos principal name to a user in the WebSphere registry.

The Java code in Example A-1 provides an example of a JAAS custom mapping login module. This login module extracts the Kerberos principal name from the `KRBAuthnToken` and maps it to a user in the WebSphere user registry. In this example, the mapping logic assumes that the Kerberos principal has the format of `krb_username`. The `krb_` prefix is removed to then map to the user `username`. The mapped identity, `username`, is added to the hash table as using the property `AttributeNameConstants.WSCREDENTIAL_USERID`.

Example A-1 Sample JAAS custom mapping login module

```
// This sample program is provided AS IS and may be used, executed,
copied and modified without royalty payment by customer (a) for its own
// instruction and study, (b) in order to develop applications designed
to run with an IBM WebSphere product, either for customer's own
internal use
// or for redistribution by customer, as part of such an application,
in customer's own products.
```

```
//
// Product 5630-A36,  (C) COPYRIGHT International Business Machines
Corp.,2009
// All Rights Reserved * Licensed Materials - Property of IBM

package com.ibm.websphere.security;

import java.util.Map;
import java.lang.reflect.Array;
import javax.security.auth.Subject;
import javax.security.auth.callback.*;
import javax.security.auth.login.LoginException;
import javax.security.auth.spi.LoginModule;
import javax.security.auth.kerberos.*;
import com.ibm.websphere.security.auth.WSLoginFailedException;
import com.ibm.wsspi.wssecurity.platform.token.KRBAuthnToken;
import com.ibm.wsspi.security.token.AttributeNameConstants;

/**
 *
 * @author IBM Corporation
 * @version 1.0
 * @since 1.0
 *
 */

public class ITSOSpnegoMappingLoginModule implements LoginModule {
    /*
     *
     * Constant that represents the name of this mapping module.
Whenever this sample
     * code is used to create a class with a different name, this value
should be changed.
     *
     */
    private final static String MAPPING_MODULE_NAME =
"com.ibm.websphere.security.ITSOSpnegoMappingLoginModule";

    private String mapUid = null;
    /**
     * Construct an uninitialized WSLoginModuleImpl object.
     */
    public ITSOSpnegoMappingLoginModule() {
        debugOut("ITSOSpnegoMappingLoginModule() entry");
        debugOut("ITSOSpnegoMappingLoginModule() exit");
    }

    /**
     * Initialize this login module.
     *
     *
     * This is called by the  LoginContext after this login module is
     * instantiated. The relevant information is passed from the
LoginContext
     * to this login module. If the login module does not understands
any of the data
     * stored in the sharedState and options parameters,
     * they can be ignored.
     *
     *
     * @param subject The subject to be authenticated.
     * @param callbackHandler
     *               A  CallbackHandler for communicating with the end
user to gather login information
                     (e.g., username and password).
     * @param sharedState
     *                The state shared with other configured login
modules.
     * @param options The options specified in the login configuration
for this particular login module.
     */
    public void initialize(Subject subject, CallbackHandler
callbackHandler,
                           Map sharedState, Map options) {
        debugOut("initialize(subject = \"" + subject.toString() +
                 "\", callbackHandler = \"" +
callbackHandler.toString() +
                 "\", sharedState = \"" + sharedState.toString() +
                 "\", options = \"" + options.toString() + "\")");

        this.subject = subject;
        this.callbackHandler = callbackHandler;
        this.sharedState = sharedState;
        this.options = options;

        debug =
"true".equalsIgnoreCase((String)this.options.get("debug"));

        debugOut("initialize() exit");
    }
    /**
     *
     * Method to authenticate a Subject (phase 1).
     *
     *
     *
     * This method authenticates a Subject. It uses CallbackHandler to 
gather
     * the Subject information, like username and password for example, 
and verify these
     * information. The result of the authentication is saved in the 
private state within
     * this login module.
     *
     *
     * @return  true if the authentication succeeded, or false
     *         if this login module should be ignored.
     * @exception LoginException
     *                   If the authentication fails.
     */
    public boolean login() throws LoginException
    {
        debugOut("ITSOSpnegoMappingLoginModule.login() entry");

        boolean succeeded = false;
        KRBAuthnToken krbAuthnToken = null;
        String kerbPrefix = "krb_";

        java.util.Set krb5Principals= 
subject.getPrivateCredentials(KRBAuthnToken.class);

        java.util.Iterator krb5PrincIter = krb5Principals.iterator();

        while (krb5PrincIter.hasNext()) {
            krbAuthnToken = (KRBAuthnToken)krb5PrincIter.next();
            String kerbPrincipal = (String) 
krbAuthnToken.getTokenPrincipal() + "@" + 
krbAuthnToken.getTokenRealm();
            debugOut("Kerberos principal name: "+ 
kerbPrincipal.toString());
            String principal = null;

            int index = kerbPrincipal.indexOf("@");
            if (index > -1)
```

```
             {
                 principal =
kerbPrincipal.substring(0,kerbPrincipal.indexOf("@"));
             }
             else
             {
                principal = kerbPrincipal;
             }
            if (principal.startsWith(kerbPrefix))
             {

                 mapUid =
principal.substring(kerbPrefix.length(),principal.length());
                 debugOut("mapUid: "+mapUid);

                 java.util.Hashtable customProperties =
(java.util.Hashtable)

sharedState.get(AttributeNameConstants.WSCREDENTIAL_PROPERTIES_KEY);
                 if (customProperties == null) {
                     customProperties = new java.util.Hashtable();
                 }
                 succeeded = true;

customProperties.put(AttributeNameConstants.WSCREDENTIAL_USERID,
mapUid);


Map<String,java.util.Hashtable>mySharedState=(Map<String,java.
                 util.Hashtable>)sharedState;

mySharedState.put(AttributeNameConstants.WSCREDENTIAL_PROPERTIES_KEY,
                 customProperties);

                 debugOut("Add a mapping user ID to Hashtable, mapping
ID = "+mapUid);

                 debugOut("login() custom properties = " +
customProperties);
             }
             else
                 debugOut("Kerberos principal doesn't start with
correct prefix, will skip mapping.");
        }
        succeeded = true;
        debugOut("ITSOSpnegoMappingLoginModule.login() exit");

        return succeeded;
    }

    /**
     *
     * Method to commit the authentication result (phase 2).
     *
     *
     *
     * This method is called if the LoginContext's overall
authentication
     * succeeded (the revelant REQUIRED, REQUISITE, SUFFICIENT and
OPTIONAL login module
     * succeeded).
     *
     *
     * @return true if the commit succeeded, or false
     *         if this login module should be ignored.
     * @exception LoginException
     *                    If the commit fails.
     */
    public boolean commit() throws LoginException
    {
        debugOut("commit()");

        debugOut("commit()");

        return true;
    }

    /**
     * Method to abort the authentication process (phase 2).
     *
     *
     * This method is called if the LoginContext's overall
authentication
     * failed (the revelant REQUIRED, REQUISITE, SUFFICIENT and
OPTIONAL login module
     * did not succeed).
     *
     *
     *
     * If this login module's authentication attempt succeeded, then
this method cleans
     * up the previous state saved in phase 1.
     *
     *
     * @return true if the abort succeeded, or false
     *         if this login module should be ignored.
     * @exception LoginException
     *                   If the abort fails.
     */
    public boolean abort() throws LoginException {
        debugOut("abort() entry");
        debugOut("abort() exit");
        return true;
    }

    /**
     * Method which logs out a Subject.
     *
     * @return true if the logout succeeded, or false
     *         if this login module should be ignored.
     * @exception LoginException
     *                   If the logout fails.
     */
    public boolean logout() throws LoginException
    {
        debugOut("logout() entry");
        debugOut("logout() exit");

        return true;
    }

    private void cleanup()
    {
        debugOut("cleanup() entry");
        debugOut("cleanup() exit");
    }

    /*
     *
     * Private method to print trace information.  This implementation
uses System.out
     * to print trace information to standard output, but a custom
tracing system can
     * be implemented here as well.
     *
     */
    private void debugOut(Object o)
    {
        System.out.println("Debug: " + MAPPING_MODULE_NAME);
        if (o != null) {
            if (o.getClass().isArray()) {
                int length = Array.getLength(o);
                for (int i = 0; i < length; i++) {
                    System.out.println("\t" + Array.get(o, i));
                }
            } else {
                System.out.println("\t" + o);
            }
        }
    }
    private Subject subject;
    private CallbackHandler callbackHandler;
    private Map sharedState;
    private Map options;

    protected boolean debug = false;
}
```

B

Configuring Web browsers for SPNEGO

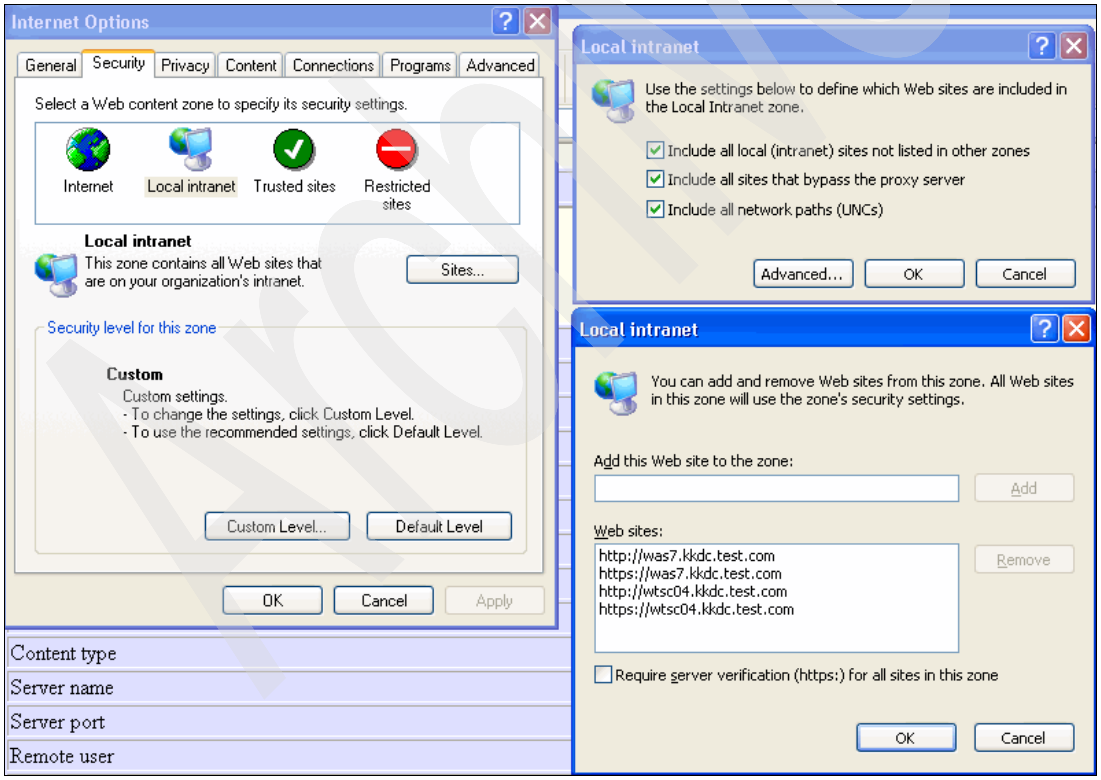
This appendix describes how to configure the Web browser on the user workstation. The examples in this appendix use Microsoft Internet Explorer and Mozilla Firefox.

Configuring Microsoft Internet Explorer

Complete the following steps to ensure that Microsoft Internet Explorer is enabled to perform SPNEGO authentication:

1. On the user workstation, log on to the Microsoft Active Directory domain.
2. Launch Microsoft Internet Explorer.
3. Within Internet Explorer, click **Tools** → **Internet Options**, and select the Security tab. Select the Local intranet icon, and click **Sites**.
4. In the Local intranet window, shown in Figure B-1, ensure that the “Include all local (intranet) sites not listed in other zones” option is selected, and then click **Advanced**. Then, complete the “Add this Web site to the zone” field with the host names of the Web sites for which SSO is enabled. Click **OK**.

   In this example, the z/OS test system at `http://wtsc04.kkdc.test.com` and the distributed WebSphere system at `http://was7.kkdc.test.com` are entered.



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.

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)

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

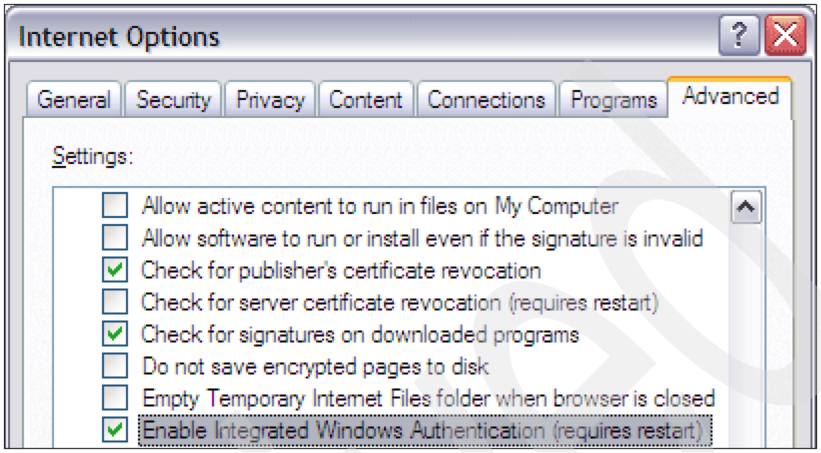
Add this Web site to the zone:

Web sites:
http://was7.kkdc.test.com
https://was7.kkdc.test.com
http://wtsc04.kkdc.test.com
https://wtsc04.kkdc.test.com

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

Figure B-1 Internet Explorer Local intranet window

5. On the Internet Options window, click the Advanced tab, and scroll to Settings. Ensure that the “Enable Integrated Windows Authentication (requires restart)” option is selected, as shown in Figure B-2. Click **OK**.



- [ ] 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)

Figure B-2 Internet Explorer Internet Options window

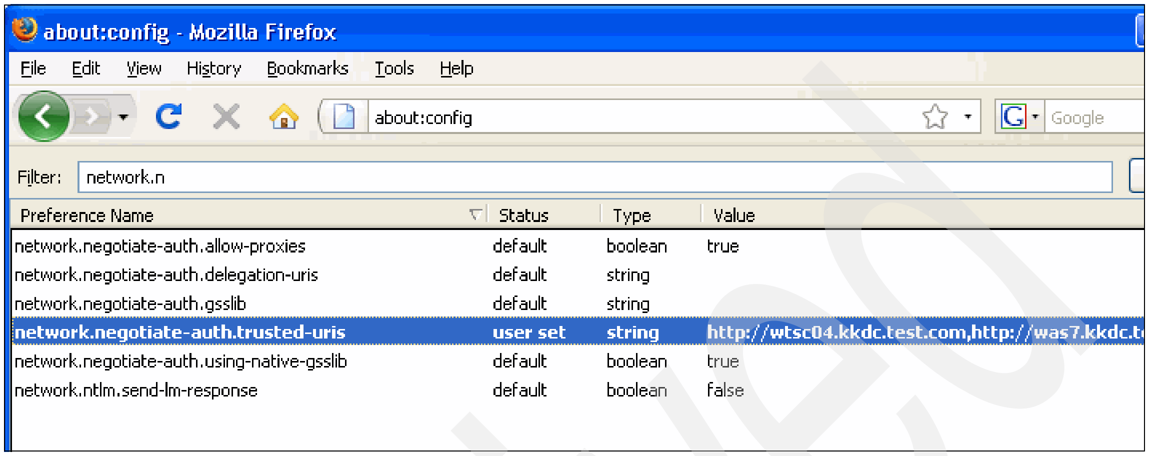
6. Restart Microsoft Internet Explorer to activate this configuration.

Configuring Mozilla Firefox

Complete the following steps to ensure that Mozilla Firefox is enabled to perform SPNEGO authentication:

1. On the user workstation, log on to the Microsoft Active Directory domain.
2. Launch Mozilla Firefox.
3. In the browser address field, type `about:config`.
4. In the filter field, type `network.n`.
5. Double-click **network.negotiate-auth.trusted-uris** to list the sites that are permitted to engage in SPNEGO authentication with the browser.

6. Enter a comma-delimited list of trusted domains or URLs., such as `http://wtsc04.kkdc.test.com,http://was7.kkdc.test.com` in this example, as shown in Figure B-3. The separator is comma.


about:config - Mozilla Firefox
File Edit View History Bookmarks Tools Help
about:config
Google
Filter: network.n

Preference Name	Status	Type	Value
network.negotiate-auth.allow-proxies	default	boolean	true
network.negotiate-auth.delegation-uris	default	string	
network.negotiate-auth.gsslib	default	string	
network.negotiate-auth.trusted-uris	**user set**	**string**	**http://wtsc04.kkdc.test.com,http://was7.kkdc.t**
network.negotiate-auth.using-native-gsslib	default	boolean	true
network.ntlm.send-lm-response	default	boolean	false

Figure B-3 Mozilla Firefox configuration window

7. If the deployed SPNEGO solution uses the advanced Kerberos feature of credential delegation, double-click **network.negotiate-auth.delegation-uris** to list the sites for which the browser can delegate user authorization to the server.
8. Click **OK** and the configuration displays as updated.
9. Restart the Firefox browser to activate this configuration.

C

Sample applications

This appendix includes information for implementing the applications that we used to test the Kerberos configuration in several chapters of this book. For information about downloading the Web material for this book, see Appendix E, “Additional material” on page 525.

Java code for the SPNEGO client sample

We used the SpnegoClientSample application in Chapter 14, “Single sign-on from a Java thin client with AIX and z/OS Kerberos trusted realms” on page 347. Example C-1 shows the code for the application.

Example C-1 Sample SpnegoClientSample used for this chapter

```
// This sample program is provided AS IS and may be used, executed,
copied and modified without royalty payment by customer (a) for its own
// instruction and study, (b) in order to develop applications designed
to run with an IBM WebSphere product, either for customer's own
internal use
// or for redistribution by customer, as part of such an application,
in customer's own products.
//
// Product 5630-A36,  (C) COPYRIGHT International Business Machines
Corp.,2009
// All Rights Reserved * Licensed Materials - Property of IBM

package com.ibm.ws.security.auth.kerberos;
import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.URL;

import org.ietf.jgss.*;
import java.security.*;
import javax.security.auth.Subject;
import javax.security.auth.login.*;
import javax.security.auth.callback.CallbackHandler;
import
com.ibm.websphere.security.auth.callback.NonPromptCallbackHandler;
import com.ibm.websphere.security.auth.callback.WSCallbackHandlerImpl;

import com.ibm.ws.util.Base64;

public class SpnegoClientSample {
    private static Oid krb5MechOid    = null;
    private static Oid spnegoMechOid  = null;
    private static String host = null;
    private static String port = "9080";  // default port
    private static String app = null;     // default app
    private static String realm = null;   // it can be blank if no
trusted realms
    private static boolean wCcache = false;
    private static String user = null;
    private static String pwd = null;

    public static void main(String[] args) throws Exception {

        getParameters(args);

        String targetSpn = "HTTP"+"/"+host+"@"+realm;
        System.out.println("targetSpn: " + targetSpn);

        String targetUrl = "http://"+host+":"+port+"/"+app;
        System.out.println("targetUrl: " + targetUrl);

        getMechOid();

        GSSManager manager = GSSManager.getInstance();

        GSSCredential clientGssCreds = null;

        GSSName gssUserName = manager.createName(user, 
GSSName.NT_USER_NAME, krb5MechOid);

        System.setProperty("javax.security.auth.useSubjectCredsOnly", 
"false");

        if (pwd != null) {
            // Authenticate to the KDC
            Subject subject = getKerberosTgt();

System.setProperty("javax.security.auth.useSubjectCredsOnly", "true");

            final Oid fnKrb5MechOid = krb5MechOid;
            final Oid fnSpnegoMechOid = spnegoMechOid;
            final GSSManager fnManager = manager;
            final GSSName fnGssUserName = gssUserName;
            clientGssCreds = (GSSCredential) Subject.doAs(subject, new 
PrivilegedExceptionAction()
            {
                public Object run() throws GSSException, Exception  {
                        try {
                                GSSCredential gssCred = 
fnManager.createCredential(fnGssUserName.canonicalize(fnKrb5MechOid), 
GSSCredential.INDEFINITE_LIFETIME, fnKrb5MechOid, 
GSSCredential.INITIATE_ONLY);
                                return gssCred;
                        } catch (GSSException gsse) {
                           gsse.printStackTrace();
                        } catch (Exception e) {
                           e.printStackTrace();
                        }
                return null; }
            });

        } else if (wCcache) {
            // Use Microsoft native Kerberos credential cache.
            System.out.println("Use Windows native Kerberos credential
cache");
            clientGssCreds = manager.createCredential(null,

GSSCredential.INDEFINITE_LIFETIME,
                                                      krb5MechOid,

GSSCredential.INITIATE_ONLY);
        } else {
            // Use the Kerberos credential cache; need to run the Java
kinit command first
            System.out.println("Use Kerberos credential cache from the
kinit command");
            clientGssCreds =
manager.createCredential(gssUserName.canonicalize(krb5MechOid),

GSSCredential.INDEFINITE_LIFETIME,

krb5MechOid,

GSSCredential.INITIATE_ONLY);
        }

        clientGssCreds.add (gssUserName,
                          GSSCredential.INDEFINITE_LIFETIME,
                          GSSCredential.INDEFINITE_LIFETIME,
                          spnegoMechOid,
                          GSSCredential.INITIATE_ONLY);


        System.out.println("Client GSS creds: " + clientGssCreds);

        GSSName gssServerName = manager.createName(targetSpn,
GSSName.NT_USER_NAME);
```

```
        System.out.println("Target server name: " + gssServerName);

        GSSContext clientContext =
manager.createContext(gssServerName.canonicalize(spnegoMechOid),
                                                    spnegoMechOid,
                                                    clientGssCreds,

GSSContext.DEFAULT_LIFETIME);
        clientContext.requestCredDeleg(true);

        byte[] krbToken = new byte[0];
        krbToken = clientContext.initSecContext(krbToken, 0,
krbToken.length);

        System.out.println("Kerberos service principal: " +
clientContext.getTargName());
        System.out.println("State of GSS delegation: " +
clientContext.getCredDelegState());

        URL url = new URL(targetUrl);
        System.out.println("Open connection for URL: "+url);
        HttpURLConnection con= (HttpURLConnection)
url.openConnection();

        try {
            con.setRequestProperty("Authorization", "Negotiate " +
Base64.encode(krbToken));
            con.getResponseCode();
           System.out.println("Response code:" + con.getResponseCode()
+ " for URL "+ url);
            if (con.getResponseCode() == 200) {
                System.out.println("Successfully connect to URL "+
url);
            } else {
                System.out.println("Failed to connect to URL "+ url +
"with response code: "+con.getResponseCode());
            }
        } catch (IOException e) {
            e.printStackTrace();
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    private static void usage()
    {
        System.out.println("Usage:\njava spnegoClientSample
[options]\n");
        System.out.println("where [options] include:\n");
        System.out.println("-host\t\tprovide the target server host
name");
        System.out.println("-port\t\tprovide the target server port
number;");
        System.out.println("-realm\t\tprovide the target Kerberos realm
name");
        System.out.println("-app\t\tprovide the application name");
        System.out.println("-wCcache\t\tuse the Microsoft Windows
native Kerberos credential cache");
        System.out.println("-user\t\tprovide the user in the KDC");
        System.out.println("-pwd\t\tprovide the password for the -user
parameter");
        System.out.println("-help\t\tget help how to use the
parameters");
        System.out.println("-?\t\tget help how to use the parameters");
        System.exit(0);
    }

    // Get the Kerberos TGT
    private static Subject getKerberosTgt()
    {
        CallbackHandler callbackHandler = new
WSCallbackHandlerImpl(user, null, pwd);

        LoginContext context = null;
        try {
            context = new LoginContext("WSKRB5Login", callbackHandler);
            context.login();
        } catch (LoginException e) {
            System.err.println("Login failed");
            e.printStackTrace();
            System.exit(1);
        }
        Subject subject = context.getSubject();
        return subject;
    }

    private static void getMechOid() {
        try {
        krb5MechOid    = new Oid("1.2.840.113554.1.2.2");
        spnegoMechOid  = new Oid("1.3.6.1.5.5.2");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static void getParameters(String args[]) {
        if(args.length != 0) {
            for (int i = 0; i < args.length; i++) {
                if (args[i].equalsIgnoreCase("-wCcache")) {
                    wCcache = true;
                }

                if (args[i].equalsIgnoreCase("-host")) {
                    host = args[i+1];
                }
                if (args[i].equalsIgnoreCase("-port")) {
                    port = args[i+1];
                }

                if (args[i].equalsIgnoreCase("-app")) {
                    app = args[i+1];
                }

                if (args[i].equalsIgnoreCase("-realm")) {
                    realm = args[i+1];
                }

                if (args[i].equalsIgnoreCase("-user")) {
                    user = args[i+1];
                }

                if (args[i].equalsIgnoreCase("-pwd")) {
                    pwd = args[i+1];
                }

                if(args[i].equalsIgnoreCase("-help") ||
args[i].equalsIgnoreCase("?"))
                {
                    usage();
                }
            }
        } else {
            usage();
        }
    }
}
```

WeatherJavaBean application

We used the WeatherJavaBean application included in this archive as a starting point for the scenario in Chapter 10, "WS-SecurityKerberos with a J2EE Web services client" on page 217. The Web material for the WeatherJavaBean application consists of the files that are required to build the database that is used by the application and the project interchange files that are intended for import into a Rational Application Developer V7.5 workspace.

The files are located in the following folders:

- The `WeatherJavaBean` folder

  This folder contains the `WeatherWebService.zip` project interchange file that includes the Web service provider (WeatherJavaBeanServer application) and a Web service client (WeatherJavaBeanWebClientEar).

- The `Weather Derby Database` folder

  This folder contains a compressed file, `weather_database.zip`, of the Weather database:

Importing project interchange files

Use the following procedure to import project interchange files (PIFs) into a Rational Application Developer workspace:

1. In the workbench of the Rational Application Developer, select **File** → **Import**. Then, expand **Other folder**, and click **Project Interchange**. Click **Next**.
2. Click **Browse** to locate the project interchange file (*name*`.zip` file) and then click **Select All**.
3. Click **Finish**.

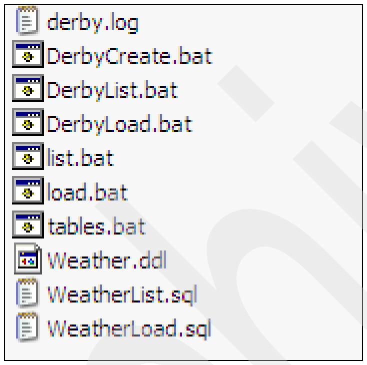
Using the WeatherJavaBean application

We use the WeatherJavaBean application as a starting point for the scenario described in Chapter 10, “WS-SecurityKerberos with a J2EE Web services client” on page 217.

Set up the WEATHER database (Derby)

The WEATHER database is implemented as a Derby database. To set up the WEATHER Derby database, follow these steps:

1. Extract the weather database files from `weather_db.zip` into the root (`C:\`) directory. Figure C-1 shows a listing of these files.



```
derby.log
DerbyCreate.bat
DerbyList.bat
DerbyLoad.bat
list.bat
load.bat
tables.bat
Weather.ddl
WeatherList.sql
WeatherLoad.sql
```

Figure C-1 WEATHER database files

2. Open the following files in a text editor:
   - `DerbyCreate.bat`
   - `DerbyLoad.bat`
   - `DerbyList.bat`
3. Modify the paths on the first three lines of each file to match the installation directory of WebSphere Application Server as follows:
   - `SET WAS_HOME=C:\`*`WAS_HOME`*
   - `SET JAVA_HOME=C:\`*`WAS_HOME`*`\java`
   - `SET DERBY_HOME=C:\`*`WAS_HOME`*`\derby`

4. Run **`DerbyCreate.bat`** to create the database schema and tables. A `WEATHER` directory is created that contains the database itself. See Figure C-2.

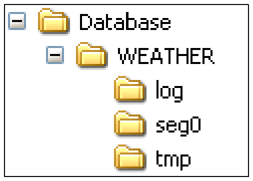


Figure C-2 The WEATHER directory

Note: Keep the full path of the database directory in a convenient location. It is used when configuring the data source for the Web service modules that are derived from the weather forecast application packages. Given the directory structure shown in Figure C-2, the path is:

```
C:/Database/WEATHER
```

5. Run **`DerbyLoad.bat`** to load the weather application data into the database.
6. Run **`DerbyList.bat`** to test the database and list all its contents, as shown in Example C-2.

Example C-2 Database listing

```
C:\weatherdb\Database>java -classpath
"C:\webspherev7\appserver\derby\lib\derby
jar;C:\webspherev7\appserver\derby\lib\derbytools.jar"
org.apache.derby.tools.i
 list.bat
ij version 10.3
ij> connect 'jdbc:derby:WEATHER';
ij> run 'WeatherList.sql';
ij> SELECT * from ITSO.SANJOSE;
WEATHERDA&|CONDITION           |WINDDIR
|TEMPERATURE|WINDSPEED
--------------------------------------------------------------------
2006-01-07|stormy              |W              |6          |11
2006-07-07|rainy               |NE             |33         |5
2006-09-17|sunny               |SW             |23         |6
2006-09-18|partly cloudy       |W              |20         |9
2006-09-19|cloudy              |W              |17         |11
2006-09-28|sunny               |WE             |30         |7

6 rows selected
ij> disconnect all;
ij> exit;
```

7. Create a JDBC provider for Derby and a data source for the weatherdb database. The JNDI name is `jdbc/weather`. The path to the database is `C:/Database/WEATHER`.

Importing the base Web services application

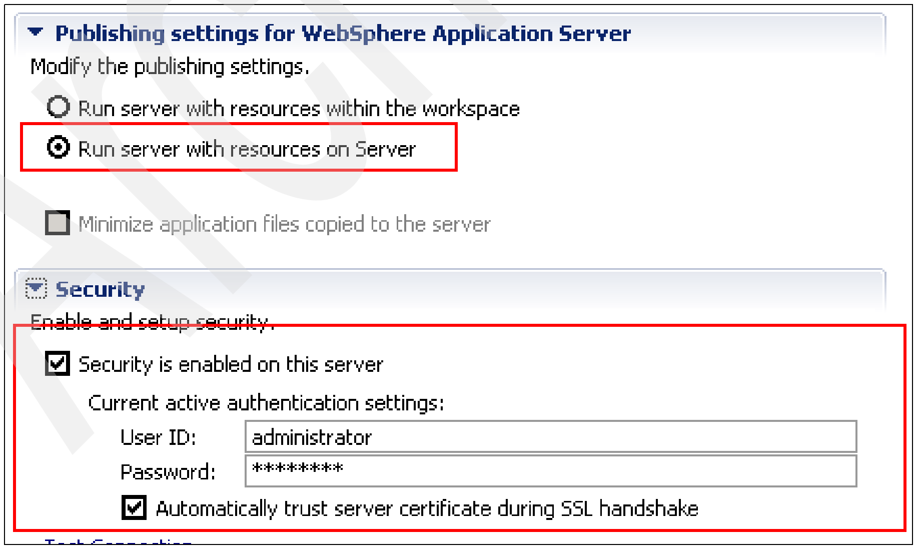
To import the base Weather Web services application, complete the following steps:

1. In the workbench of the Rational Application Developer, select **File** → **Import**. Then, expand **Other,** and select **Project Interchange**. Click **Next**.
2. Click **Browse** to locate `WeatherWebService.zip`, and then click **Select All.**
3. Click **Finish**.

Preparing the integrated test environment

If you are using Rational Application Developer and its integrated WebSphere Application Server test environment, complete the following steps to prepare the integrated test environment:

1. In the Servers view, double-click **WebSphere Application Server V7** to open the server configuration editor.
2. Under the Publishing settings for WebSphere Application Server section, select **Run server with resources on Server**, as shown in Figure C-3.



Publishing settings for WebSphere Application Server

Modify the publishing settings.

( ) Run server with resources within the workspace

(●) Run server with resources on Server

[ ] Minimize application files copied to the server

Security

Enable and setup security.

[✓] Security is enabled on this server

Current active authentication settings:

User ID: administrator

Password: *********

[✓] Automatically trust server certificate during SSL handshake

Figure C-3 Server editor settings

3. In the server configuration editor, expand the Security section, and select **Security is enabled on this server**. Complete the User ID and Password fields, also shown in Figure C-3, and specify the administrator user of the WebSphere administrative console.
4. Select **Automatically trust server certificate during SSL handshake**.
5. Save and close the server configuration editor.

Note: The “Run server with resources on server” option installs and copies the full application and its server-specific configuration from the workbench into the directories of the server. If you use this option, you cannot modify the settings inside the `.ear` file using the WebSphere administrative console.

In Chapter 10, “WS-SecurityKerberos with a J2EE Web services client” on page 217, we use the administrative console to apply the policy set and to bind to the Web service application. Thus, we use the “Run server with resources on Server” option.

Deploying the enterprise applications to the server

To deploy the application to the server, follow these steps:

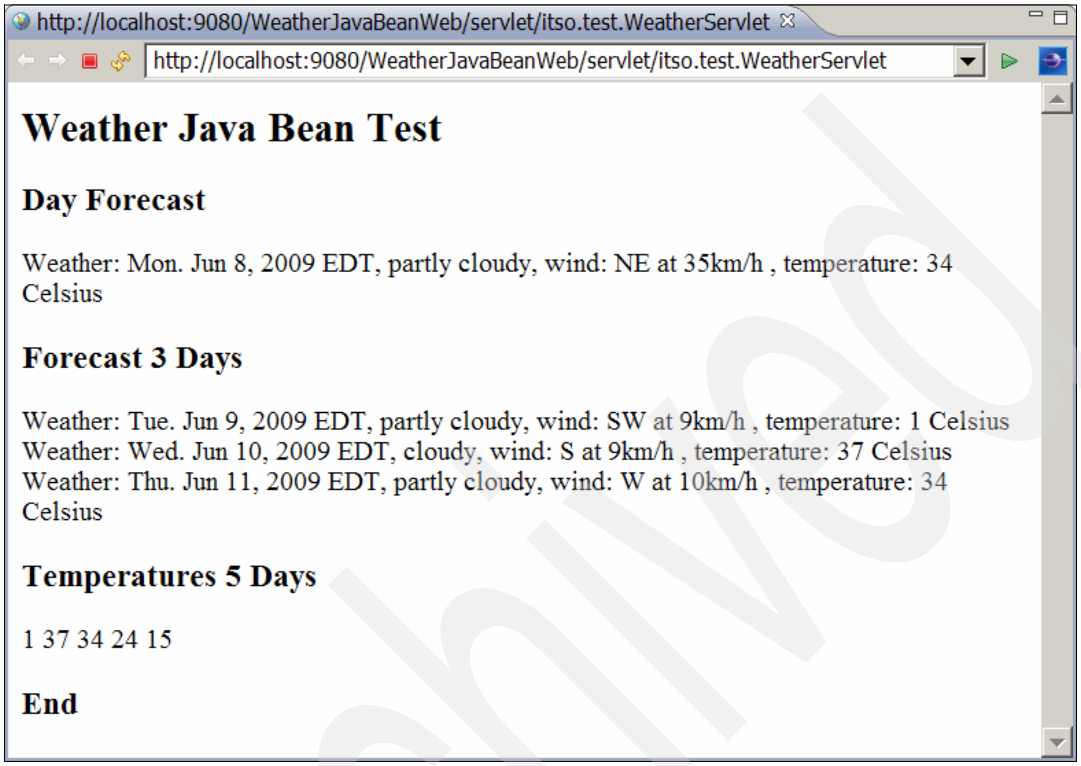
1. Start the server.
2. In the Servers view, right-click **WebSphere Application Server V7.0,** and select **Add and Remove Projects.** Select the following projects to add to the server:
   – WeatherJavaBeanServer
   – WeatherJavaBeanWebClient

Testing the enterprise applications

Our Weather Web services application is built on top of an existing Java Platform, Enterprise Edition (Java EE) application. So, we need to make the base Java EE application work before we expose it as a Web service. To test the basic functionality of the applications, a simple servlet is included in the base code to make sure the base application:

1. In the Enterprise Explorer, expand **WeatherJavaBeanWeb** → **Java Resources** → **src** → **itso.test**. Right-click **WeatherServlet,** and select **Run As** → **Run on Server**.
2. When prompted for Server Selection, ensure that the correct server is selected.
3. Click **Finish**.

The servlet should produce sample output in the Web browser, as shown in Figure C-4.


http://localhost:9080/WeatherJavaBeanWeb/servlet/itso.test.WeatherServlet

Weather Java Bean Test

Day Forecast

```
Weather: Mon. Jun 8, 2009 EDT, partly cloudy, wind: NE at 35km/h , temperature: 34
Celsius
```

Forecast 3 Days

```
Weather: Tue. Jun 9, 2009 EDT, partly cloudy, wind: SW at 9km/h , temperature: 1 Celsius
Weather: Wed. Jun 10, 2009 EDT, cloudy, wind: S at 9km/h , temperature: 37 Celsius
Weather: Thu. Jun 11, 2009 EDT, partly cloudy, wind: W at 10km/h , temperature: 34
Celsius
```

Temperatures 5 Days

```
1 37 34 24 15
```

End

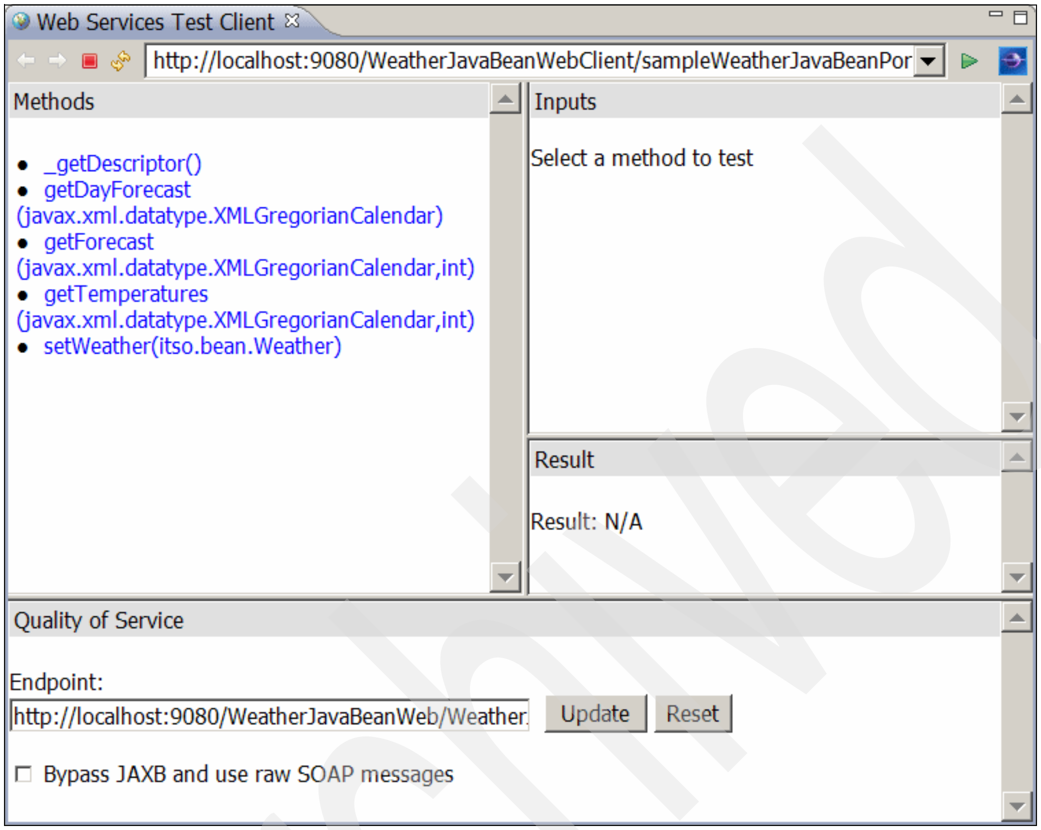
Figure C-4 Test the enterprise application

Testing the Weather Web service application

A sample JSP client is provided to test the Weather Web service application. To test it, follow these steps:

1. In the Enterprise Explorer, expand **WeatherJavaBeanWebClient** → **WebContent** → **sampleWeatherJavaBeanPortProxy**. Right-click **TestClient.jsp,** and select **Run As** → **Run on Server**:

The JSP page displays in the browser, as shown in Figure C-5.



- _getDescriptor()
- getDayForecast (javax.xml.datatype.XMLGregorianCalendar)
- getForecast (javax.xml.datatype.XMLGregorianCalendar,int)
- getTemperatures (javax.xml.datatype.XMLGregorianCalendar,int)
- setWeather(itso.bean.Weather)

Select a method to test

Result: N/A

Endpoint:
http://localhost:9080/WeatherJavaBeanWeb/Weather

☐ Bypass JAXB and use raw SOAP messages

Figure C-5 Sample Web service JSP client

2. Take a close look at the Endpoint section. Note that the endpoint is set to:

   ```
   http://localhost:9080/WeatherJavaBeanWebClient/sampleWeatherJavaB
   eanPortProxy/TestClient.jsp
   ```

   If your WebSphere Application Server is not running on port 9080, you need to update the endpoint. For example, if the server is running on port 9081, you need to set the endpoint to the following URL, and then click **Update**:

   ```
   http://localhost:9081/WeatherJavaBeanWebClient/sampleWeatherJavaB
   eanPortProxy/TestClient.jsp
   ```

3. Select the `getDayForecast` method, and enter a value for `arg0`. You can copy and paste the example value that is provided by the JSP client, for example `2009-04-10T16:22:19`. Click **Invoke**. Example C-3 shows the result.

Example C-3 getDayForecast result

```
returnp:
  condition: partly cloudy
  date: 2009-06-08T00:00:00-04:00
  dbflag: true
  temperatureCelsius: 34
  windDirection: NE
  windSpeed: 35
```

D

Installing the Application Client for WebSphere Application Server

We use the IBM Application Client for WebSphere Application Server and Java thin client in Chapter 14, “Single sign-on from a Java thin client with AIX and z/OS Kerberos trusted realms” on page 347. This appendix shows how to install the Application Client software, including the thin clients.

In this example, the software is installed in the following directory:

```
/usr/IBM/WebSphere/AppClient
```

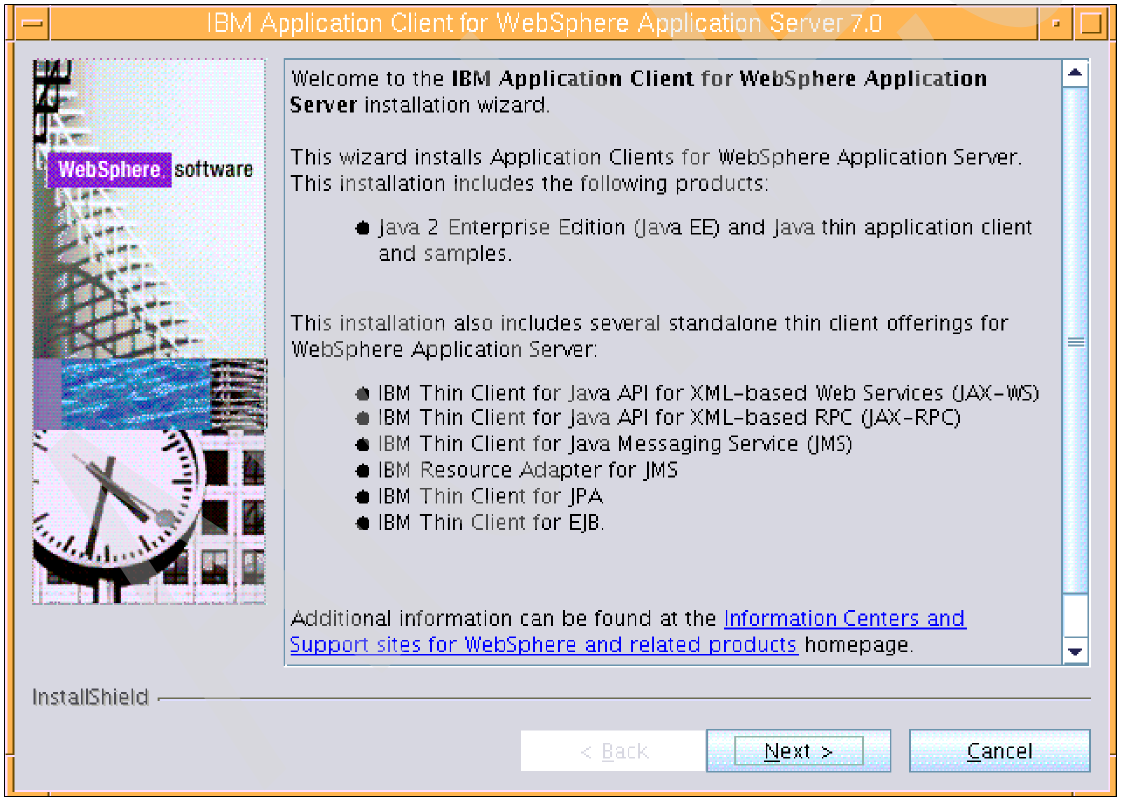
Note: When installing application clients, the current working directory must be the directory where the installer binary program is located. This placement is important because the resolution of the class files location is done in the current working directory.

To install the Application Client for WebSphere Application Server, follow these steps:

1. On the command line, change to the directory where the installer binary program is located:
   - To install from a product installation: **`cd /app_server_root/AppClient`**
   - To install from the product CD: **`cd CD_mount_point/AppClient`**
   - To install from a free download Application Client installation: **`cd /download_dir/AppClient`**
2. Run the **`install`** command. For example:

   ```
   . /install
   ```

3. Click **Next** to continue when the Welcome panel displays as shown in Figure D-1.



Welcome to the **IBM Application Client for WebSphere Application Server** installation wizard.

This wizard installs Application Clients for WebSphere Application Server. This installation includes the following products:

- Java 2 Enterprise Edition (Java EE) and Java thin application client and samples.

This installation also includes several standalone thin client offerings for WebSphere Application Server:

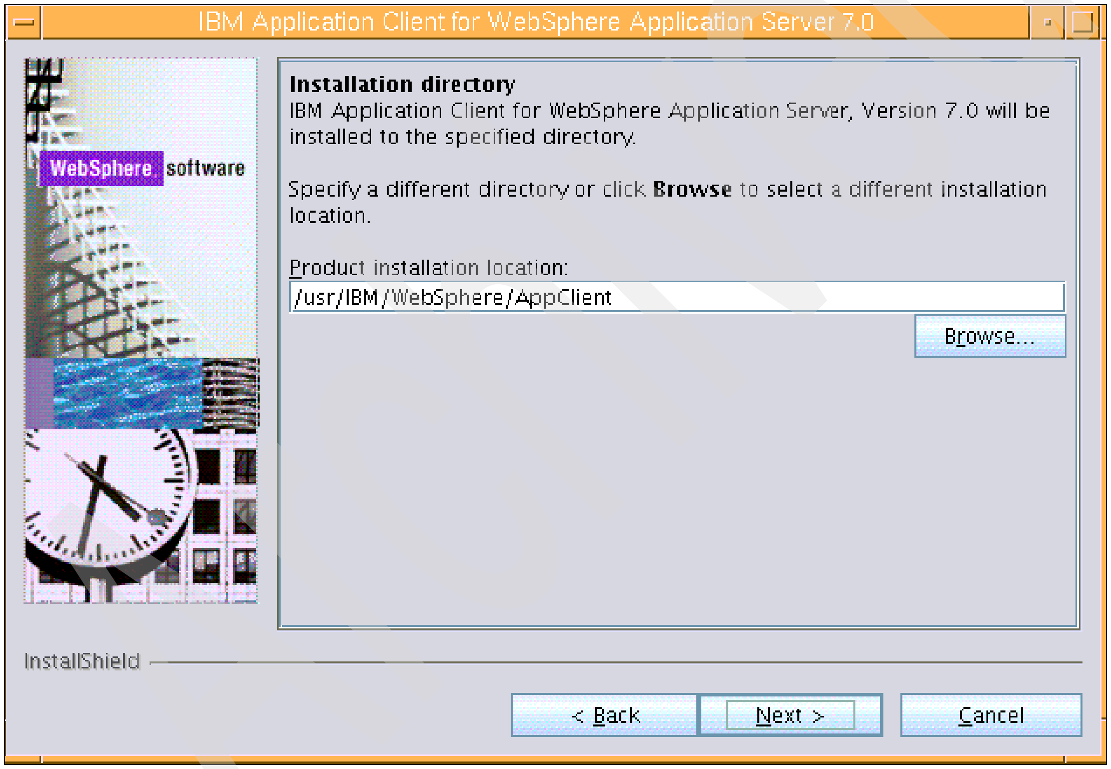
- IBM Thin Client for Java API for XML-based Web Services (JAX-WS)
- IBM Thin Client for Java API for XML-based RPC (JAX-RPC)
- IBM Thin Client for Java Messaging Service (JMS)
- IBM Resource Adapter for JMS
- IBM Thin Client for JPA
- IBM Thin Client for EJB.

Additional information can be found at the Information Centers and Support sites for WebSphere and related products homepage.

Figure D-1 Welcome panel for the Application Client

4. Select **I accept the terms in the license agreement** and then click **Next** to continue. The installation wizard checks the system for prerequisites.

5. If you see a Passed message on the wizard panel, click **Next** to continue. If you see a warning message on the wizard panel, click **Cancel** to exit the installation wizard, then install the prerequisites before restarting the installation wizard.
6. Specify the destination directory `/usr/IBM/WebSphere/AppClient` as shown in Figure D-2.
   a. If you do not want to use the default destination directory, specify a different target directory. The Product installation location field must contain a valid target directory for you to continue with the installation.
   b. Ensure that there is adequate space available in the target directory.
   c. Click **Next** to continue.


IBM Application Client for WebSphere Application Server 7.0

Installation directory
IBM Application Client for WebSphere Application Server, Version 7.0 will be installed to the specified directory.

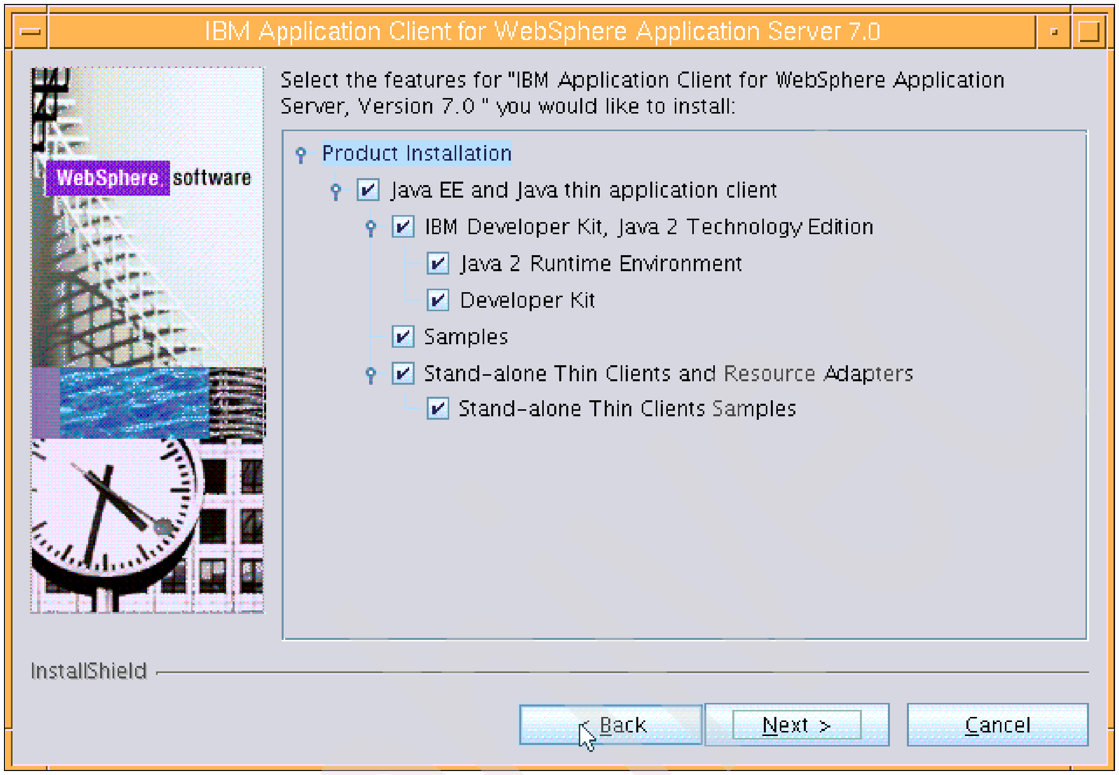
Specify a different directory or click **Browse** to select a different installation location.

Product installation location:
/usr/IBM/WebSphere/AppClient

Figure D-2 Installation directory for Application Client

7. Choose **Custom** as the setup type to allow you to select the features that you want. Click **Next**.

8. Select all features as shown in Figure D-3, and click **Next**.


IBM Application Client for WebSphere Application Server 7.0

Select the features for "IBM Application Client for WebSphere Application Server, Version 7.0 " you would like to install:

- Product Installation
  - [✓] Java EE and Java thin application client
    - [✓] IBM Developer Kit, Java 2 Technology Edition
      - [✓] Java 2 Runtime Environment
      - [✓] Developer Kit
    - [✓] Samples
    - [✓] Stand-alone Thin Clients and Resource Adapters
      - [✓] Stand-alone Thin Clients Samples

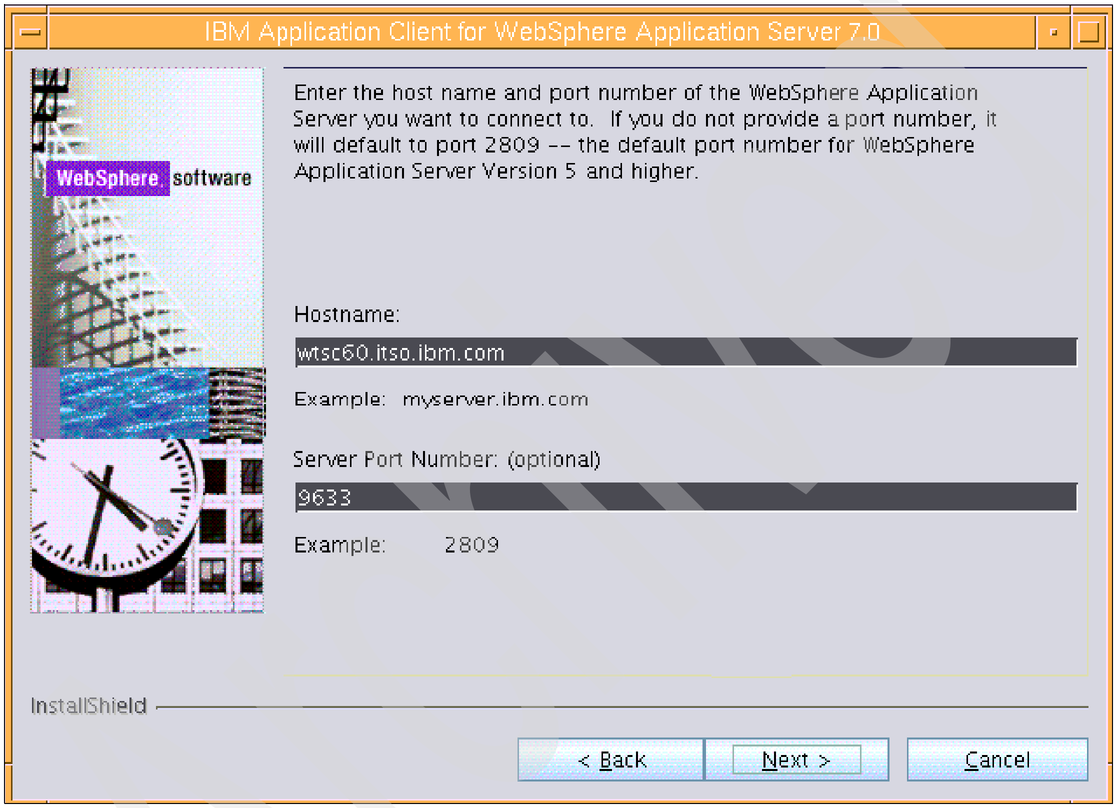
Figure D-3 Selecting the features for the Application Client

9. Enter the host name and port number of the WebSphere Application Server for z/OS server to which you want to connect, as shown in Figure D-4.

   These settings are used by the Java EE and Java thin application clients that have the Application Client as their runtime environment.

   The default bootstrap port number is 2809.

   Click **Next**.



Enter the host name and port number of the WebSphere Application Server you want to connect to. If you do not provide a port number, it will default to port 2809 -- the default port number for WebSphere Application Server Version 5 and higher.

Hostname:

wtsc60.itso.ibm.com

Example: myserver.ibm.com

Server Port Number: (optional)

9633

Example: 2809

Figure D-4 Connecting the Application Client to the WebSphere Application Server

10. Click **Next**.

11. Click **Finish** to exit the wizard.

E

Additional material

This book refers to additional material that you can download from the Internet as described in this appendix.

Locating the Web material

The Web material that 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/SG24-7771-00

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select **Additional materials** and open the directory that corresponds with the IBM Redbooks form number, SG247771.

Using the Web material

The additional Web material that accompanies this book includes the following files:

- `WCFSamples` folder

  This folder contains the WCFClient that we use in Chapter 11, "WS-SecurityKerberos with a .NET Web services client" on page 241 and WCFService applications that we use in Chapter 12, "WS-SecurityKerberos with a Thin Client for JAX-WS and .NET provider" on page 273.

- `krbSample` folder

  This folder contains the `KrbSample.ear` file and the `appdb.sql` file that we use in Chapter 13, "Single sign-on to WebSphere Application Server and DB2 from a Java application client" on page 305 and Chapter 15, "SSO to WebSphere Application Server for z/OS and DB2 using Microsoft Kerberos KDC and z/OS KDC trust" on page 391.

- `Weather` folder

  This folder contains the following folders and files:

  – The `WeatherJavaBean` folder

    This folder contains the `WeatherWebService.zip` project interchange file that includes the Web service provider (WeatherJavaBeanServer application) and a Web service client (WeatherJavaBeanWebClientEar).

  – The `Weather Derby Database` folder

    This folder contains a compressed file of the Weather database, `weather_database.zip`.

  You can use these files to build the sample applications that we use in Chapter 10, "WS-SecurityKerberos with a J2EE Web services client" on page 217.

Related publications

We consider the publications that we list in this section particularly suitable for a more detailed discussion of the topics that we cover in this book.

IBM Redbooks publications

For information about ordering these publications, see “How to get IBM Redbooks publications” on page 528. Note that some of the documents referenced here might be available in softcopy only.

- *IBM WebSphere Application Server V7.0 Web Services Guide*, SG24-7758
- *WebSphere Application Server V7.0 Security Guide*, SG24-7660
- *WebSphere Application Server V7 Administration and Configuration Guide*, SG24-7615

Online resources

The following Web sites are also relevant as further information sources:

- *How do I install Active Directory on my Windows Server 2003 server*

  `http://www.petri.co.il/how_to_install_active_directory_on_windows_2003.htm`
- OASIS Web Services Security (WSS) Kerberos Token Profile 1.1

  `http://www.oasis-open.org/committees/download.php/16788/wss-v1.1-spec-os-KerberosTokenProfile.pdf`
- WebSphere Application Server V7 Information Center

  `http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp`
- Security Server RACF Security Administrator’s Guide

  `http://publibz.boulder.ibm.com/epubs/pdf/ichza780.pdf`
- Integrated Security Services Network Authentication Service Administration

  `http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/download/euvb3a50.pd`

- Wikipedia, Domain Name System

  http://en.wikipedia.org/wiki/Domain_Name_System
- Microsoft TechNet

  http://technet.microsoft.com/en-us/library/default.aspxDescription3
- Microsoft Download Center

  http://www.microsoft.com/downloads/en/default.aspx
- MSDN .NET Framework Developer Center

  http://msdn.microsoft.com/en-us/library/default.aspx
- IBM developerWorks Technical Library

  http://www.ibm.com/developerworks/data/library/
- OASIS Web Services Security: SOAP message Security 1.0

  http://docs.oasis-open.org/wss/

How to get IBM Redbooks publications

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

IBM

Redbooks

Implementing Kerberos in a WebSphere Application Server Environment

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

Implementing Kerberos in a WebSphere Application Server Environment

Discusses how to implement Kerberos in a WebSphere environment

Provides information on using single sign-on

Includes detailed scenarios and examples

This IBM Redbooks publication discusses Kerberos technology with IBM WebSphere Application Server V7.0.0.5 on distributed platforms. IBM WebSphere Application Server V7.0.0.5 Kerberos Authentication and single sign-on (SSO) features enable interoperability and identity propagation with other applications (such as .NET, DB2, and others) that support the Kerberos authentication mechanism. With this feature, a user can log in once and then can access other applications that support Kerberos Authentication without having to log in a second time. It provides an SSO, end-to-end interoperability solution and preserves the original requester identity.

This book provides a set of common examples and scenarios that demonstrate how to use the Kerberos with WebSphere Application Server. The scenarios include configuration information for WebSphere Application Server V7 when using a KDC from Microsoft, AIX, and z/OS as well as considerations when using these products. The intended audience for this book is system administrators and developers who use IBM WebSphere Application Server V7 on distributed platforms.

SG24-7771-00 ISBN 0738433489