

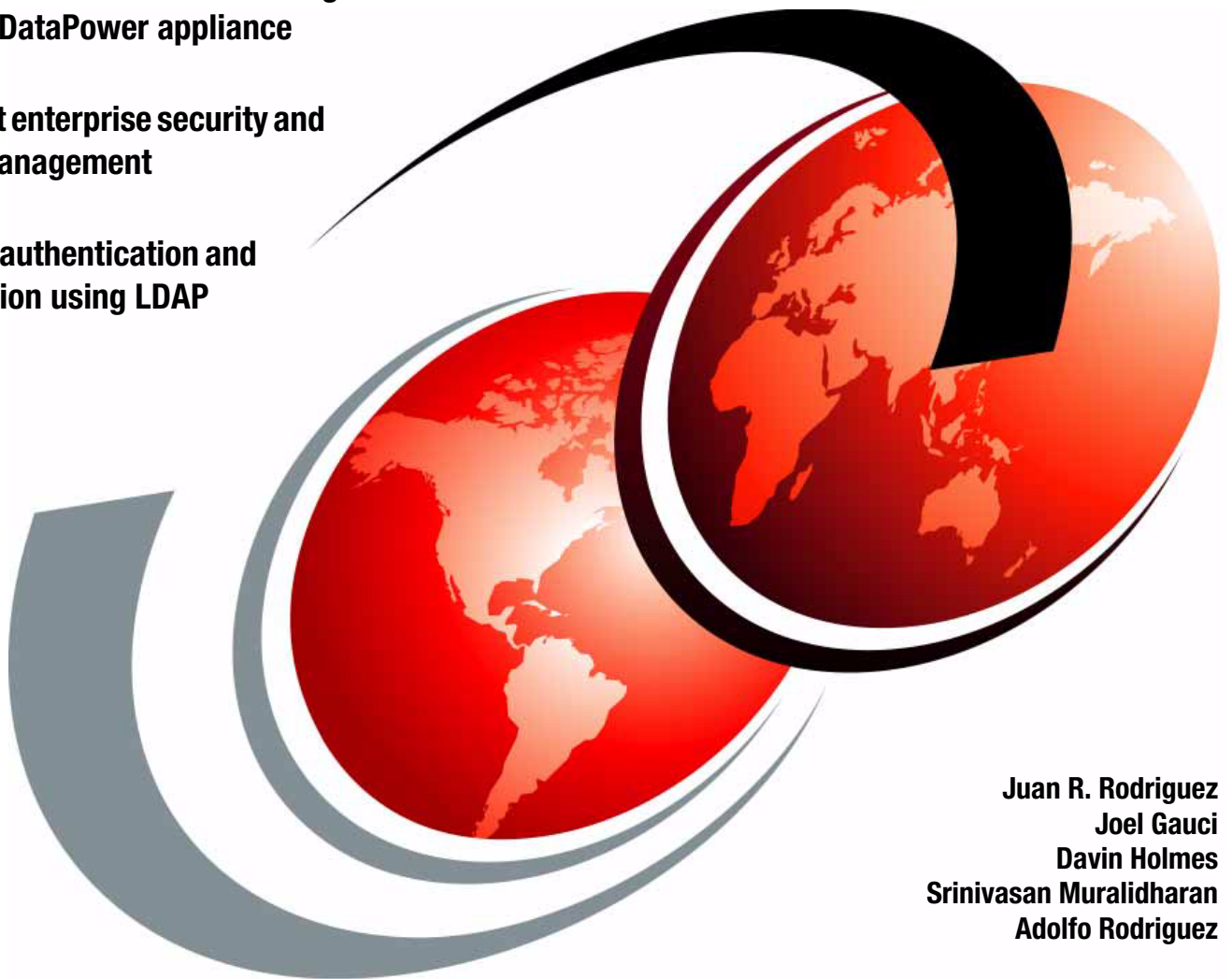
IBM WebSphere DataPower SOA Appliances

Part II: Authentication and Authorization

Integrate IBM Tivoli Access Manager
with your DataPower appliance

Implement enterprise security and
identity management

Configure authentication and
authorization using LDAP



Juan R. Rodriguez
Joel Gauci
Davin Holmes
Srinivasan Muralidharan
Adolfo Rodriguez



International Technical Support Organization

**IBM WebSphere DataPower SOA Appliances
Part II: Authentication and Authorization**

April 2008

Note: Before using this information and the product it supports, read the information in “Notices” on page v.

First Edition (April 2008)

This edition applies to Version 3, Release 6, Modification 0 of IBM WebSphere DataPower Integration Appliance.

This document created or updated on March 27, 2008.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	v
Trademarks	vi
Preface	vii
The team that wrote this paper	vii
Become a published author	viii
Comments welcome	ix
Chapter 1. Authentication and authorization	1
1.1 Concepts	2
1.2 Demonstration environment	2
1.2.1 Primes Web service	2
1.2.2 Securing the Web service	3
1.2.3 Preliminary DataPower configuration	5
1.3 Summary	17
Chapter 2. AAA concepts and policy creation	19
2.1 AAA concepts	20
2.1.1 AAA policy	20
2.1.2 AAA action	20
2.1.3 Configuring a AAA policy	20
2.1.4 Processing a AAA policy	23
2.2 Configuring the Primes Web service proxy policy	23
2.2.1 Creating a processing rule and AAA action	23
2.2.2 Adding a Results action	27
2.2.3 Committing the processing rule	28
2.3 Summary	28
Chapter 3. AAA with Tivoli Access Manager	29
3.1 IBM Tivoli Access Manager for e-business	30
3.1.1 Tivoli Access Manager and DataPower	30
3.1.2 Tivoli Access Manager and Web services overview	31
3.1.3 Designing the Tivoli Access Manager object space	32
3.1.4 Configuring the Tivoli Access Manager object space	33
3.1.5 Applying the security policy	34
3.2 Creating the Tivoli Access Manager AAA policy	35
3.2.1 Creating the AAA policy object	36
3.2.2 Extracting the identity	38
3.2.3 Authenticating requests	39
3.2.4 Mapping the credentials	44
3.2.5 Extracting the resource	44
3.2.6 Mapping the resources	45
3.2.7 Configuring the authorization	46
3.2.8 Performing the post processing	46
3.2.9 Committing the AAA policy	47
3.3 Updating the DataPower Tivoli Access Manager client object	48
3.3.1 Verifying the file creation of the Tivoli Access Manager configuration	48
3.3.2 Updating the Tivoli Access Manager object	49

3.4 Testing	54
3.4.1 Testing the support material	54
3.4.2 Testing the prerequisites	54
3.4.3 Testing requests to the DataPower appliance by using cURL	56
3.5 Summary	66
Chapter 4. AAA with the LDAP directory	67
4.1 IBM Tivoli Directory Server	68
4.1.1 Tivoli Directory Server and DataPower	68
4.1.2 Configuring the Tivoli Directory Server	69
4.2 Creating a Web service proxy for Tivoli Directory Server	70
4.3 Configuring the Web service proxy policy	71
4.3.1 Creating the processing rules	71
4.4 Creating Tivoli Directory Server AAA policies	75
4.4.1 Creating the Primes division AAA policy object	76
4.4.2 Creating the Random division AAA policy object	81
4.5 Applying the AAA policies to the Web service proxy	82
4.5.1 Configuring the getPrime and nextPrime operations	82
4.5.2 Configuring the random operation	83
4.6 Testing	85
4.7 Summary	85
Appendix A. External keys	87
GSKit ikeyman	88
Creating a keystore	88
Generating a certificate	90
Uploading the certificate and key to the DataPower appliance	91
Uploading the certificate	91
Uploading the key	93
Creating the DataPower Crypto object	95
Appendix B. Additional material	101
Locating the Web material	101
Using the Web material	101
System requirements for downloading the Web material	101
How to use the Web material	102
Related publications	103
IBM Redbooks	103
Other publications	103
Online resources	104
How to get Redbooks	104
Help from IBM	104

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:


This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

CICS®
DataPower device®
DataPower®

IBM®
IMS™
Redbooks®

Redbooks (logo) ®
Tivoli®
WebSphere®

The following terms are trademarks of other companies:

Java, J2EE, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

IBM® WebSphere® DataPower® SOA Appliances represent an important element in the holistic approach of IBM to service-oriented architecture (SOA). IBM SOA appliances are purpose-built, easy-to-deploy network devices that simplify, secure, and accelerate your XML and Web services deployments while extending your SOA infrastructure. These appliances offer an innovative, pragmatic approach to harness the power of SOA. By using them, you can simultaneously use the value of your existing application, security, and networking infrastructure investments.

This series of IBM Redpaper publications is written for architects and administrators who need to understand the implemented architecture in WebSphere DataPower appliances to successfully deploy it as a secure and efficient enterprise service bus (ESB) product. These papers give a broad understanding of the new architecture and traditional deployment scenarios. They cover details about the implementation to help you identify the circumstances under which you should deploy DataPower appliances. They also provide a sample implementation and architectural best practices for an SOA message-oriented architecture in an existing production ESB environment.

Part 2 of the series, this part, provides information about ways to integrate the DataPower appliance with other products such as IBM Tivoli® Access Manager and Tivoli Directory Server. The entire IBM WebSphere DataPower SOA Appliances series includes the following papers:

- ▶ *IBM WebSphere DataPower SOA Appliances Part I: Overview and Getting Started*, REDP-4327
- ▶ *IBM WebSphere DataPower SOA Appliances Part II: Authentication and Authorization*, REDP-4364
- ▶ *IBM WebSphere DataPower SOA Appliances Part III: XML Security Guide*, REDP-4365
- ▶ *IBM WebSphere DataPower SOA Appliances Part IV: Management and Governance*, REDP-4366

The team that wrote this paper

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

Juan R. Rodriguez is a Consulting IT professional and Project Leader at the IBM ITSO Center, Raleigh. He has an Master of Science (MS) degree in Computer Science from Iowa State University. He writes extensively and teaches IBM classes worldwide on Web technologies and information security. Before joining the IBM ITSO, Juan worked at the IBM laboratory in Research Triangle Park, North Carolina, as a designer and developer of networking products.

Joel Gauci has been working for IBM Global Business Services in France since 2001, as an Advisory IT Specialist. He mainly works in design and implementation on portal and Web services platforms. He has worked on SOA and WebSphere DataPower projects since May 2006, for leading firms in the mobile telephony area. Joel holds a master degree in computer science and engineering from l'Ecole d'Ingénieurs du Pas-de-Calais School (EIPC), France.

Davin Holmes is a Staff Software Engineer for IBM Software Group, Tivoli. He has worked in software development for seven years in a variety of technical areas, which include smartcards, enterprise software integration, and Web services, with a particular focus on information security. Davin is the team lead for the DataPower and Tivoli Security integration effort located at the Gold Coast, Australia site of the Australia Development Laboratory (ADL). He has degrees in electrical and computer engineering and optoelectronics from Queensland University of Technology and Macquarie University.

Srinivasan Muralidharan is an Advisory Engineer with 15 years of industrial experience and nine years with IBM. He is currently working on DataPower related projects at the IBM WebSphere Technology Institute. He is widely experienced in SOA-related technologies in all tiers of the software development stack. He has studied SOA performance with DataPower appliances. Srinivasan has also investigated integrating DataPower with other mid-tier and back-end traditional components, such as WebSphere Application Server, MQ, CICS®, and IMS™, in the SOA context of reusing existing systems and enterprise modernization.

Adolfo Rodriguez is a Software Architect within the IBM WebSphere Technology Institute (WSTI). He leads a team of engineers who focus on emerging technologies in WebSphere products and, more recently, DataPower SOA appliances. His recent contributions include projects in the areas of Web services enablement, XML processing, SOA management, and application-aware networking. His primary interests are networking and distributed systems, application middleware, overlays, and J2EE™ architecture. Adolfo is also an Assistant Adjunct Professor of Computer Science at Duke University, where he teaches networking courses. He has written 12 books and numerous research articles. He holds four degrees from Duke University: a Bachelor of Science in Computer Science, a Bachelor of Arts in Mathematics, MS in Computer Science, and a Ph.D. in Computer Science (Systems).

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

Robert Callaway
John Graham
Marcel Kinard
IBM Research Triangle Park, North Carolina, USA

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 papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks® in one of the following ways:

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

ibm.com/redbooks

- Send your comments in an e-mail to:

redbooks@us.ibm.com

- Mail your comments to:

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



Authentication and authorization

In this paper, we explain how you can use DataPower SOA appliances with Tivoli security software to support Web service authentication and authorization. The Tivoli software products, Tivoli Access Manager for e-business and Tivoli Directory Server, provide enterprise security and identity management. You can use them to augment the powerful capabilities of the DataPower SOA appliances.

In this chapter, we introduce the authentication and authorization features that are provided by the DataPower SOA appliances. We also set up the example environment that is used to demonstrate these features. This chapter includes the following topics:

- ▶ Authentication and authorization concepts
- ▶ Security policy discussion for demonstration environment
- ▶ Preliminary DataPower configuration to support this security policy

In the subsequent chapters of this paper, we configure and integrate the DataPower SOA appliance features with Tivoli Access Manager and Tivoli Directory Server.

1.1 Concepts

We begin by briefly discussing the concepts of authentication and authorization for two reasons. The first reason is that the differences between authentication and authorization are often misunderstood or confused. The second reason is that a discussion of the topic better illustrates how the concepts apply to the DataPower appliances.

The process of *authentication* verifies that an entity is in fact the entity that it is claiming to be. Conversely, *authorization* is used to determine whether a client is allowed to access services (or content as documents or files). In most security systems, authorization checking is only performed if authentication is successful.

1.2 Demonstration environment

To illustrate how to implement a complete authentication and authorization solution by using a DataPower appliance, we describe the existing environment that must be protected.

1.2.1 Primes Web service

The ITSOCorp corporation has made an internal application that generates prime and random numbers available as a Web service on the company intranet. The Web service uses the SOAP specification as the message protocol and HTTP as the transport protocol.

ITSOCorp has implemented a DataPower XI50 appliance to validate and route requests to the Primes Web service, as illustrated in Figure 1-1.

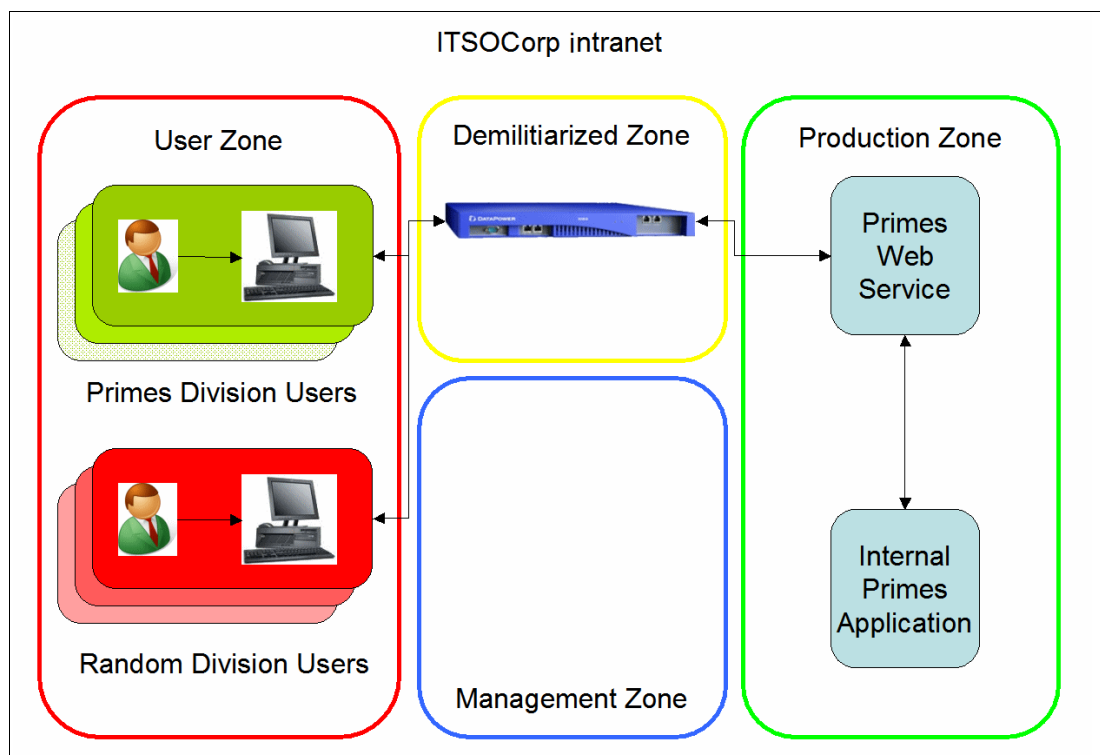


Figure 1-1 ITSOCorp Web services architecture

1.2.2 Securing the Web service

ITSOCorp requires that access to the Primes Web service be secured. In this section, we explain the security policy for the Web service.

ITSOCorp has mandated that the system demonstrate the following security characteristics:

- ▶ Authentication
- ▶ Authorization
- ▶ Confidentiality

Authentication

User identity is verified by using a Web Services Security (WS-Security) UsernameToken. WS-Security is a suite of specifications that define mechanisms for transporting a user identity in a Web service environment. The UsernameToken profile describes the transportation of a user name and a password in the header of a SOAP message. The password field of UsernameTokens must be protected by token encryption or by the use of transport level security (Secure Sockets Layer (SSL)).

Example 1-1 shows UsernameToken contained within a SOAP header.

Example 1-1 WS-Security header with UsernameToken

```
...
<soapenv:Header>
...
<wsse:Security>
<wsse:UsernameToken>
<wsse:Username>johndoe</wsse:Username>
<wsse:Password>passw0rd</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
...
</soapenv:Header>
...
```

wsse namespace: The wsse namespace may be defined in the SOAP envelope element by using the following value:

```
xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
```

In this discussion, we demonstrate the use of two different authentication services:

- ▶ IBM Tivoli Access Manager
- ▶ IBM Tivoli Directory Server

Authorization

Defining a clear policy with respect to authorization is an important part of any overall security policy. An unambiguous authorization policy can greatly simplify deployments and reduce the risk of creating vulnerabilities. In broad terms, the process of creating a security policy consists of the following steps:

- ▶ Define the *resources* that need to be protected.
- ▶ Decide which *users, groups*, or users and groups require access to these resources.
- ▶ Decide which *level* or *type* of access these users should have to the resources.

The policy must then be *applied* to the resources, so that only the correct users have access to the resources that they require.

For our example scenario, the policy can be broken down as follows:

1. The resource is the Primes Web service, specifically the operations *getPrime*, *nextPrime*, and *random*.
2. Employees of the Primes division of ITSOCorp require access to the *getPrime* and *nextPrime* operations to retrieve prime numbers.
3. Employees of the Random division of ITSOCorp require access to the *random* operation to retrieve random numbers.

The user's authenticated credential is used to determine whether they are permitted access to the Web service.

Confidentiality

Transport level security is used for requests sent to the Web service. Since the DataPower appliance is mediating these requests, connections to the appliance are SSL terminated. SSL ensures the confidentiality of user passwords as information transmitted over the SSL channel is encrypted. This protects the password in the UsernameToken of the incoming request from being discovered in transit.

For the SSL connection, only the server, for example the DataPower appliance, is required to supply a certificate for verification. Clients are not required to present a certificate for verification.

XML encryption: XML encryption can be applied to the SOAP message request to provide message-level confidentiality. The use of XML encryption in this case is outside the scope of the discussion of authentication, authorization, and audit. For more information regarding DataPower and XML encryption, refer to the companion paper *IBM WebSphere DataPower SOA Appliances Part III: XML Security Guide*, REDP-4365.

1.2.3 Preliminary DataPower configuration

In this section, we describe the initial configuration that is required on the DataPower XI50 appliance to support the Web service. We describe the authentication and authorization services in subsequent chapters.

Configuring the SSL

Connections to the DataPower appliance are required to be SSL terminated. Data that is transmitted over SSL connections is encrypted by using session keys that are secured with public key cryptography. Public key cryptography requires a public key (stored in a certificate) and a private key. The DataPower appliance provides powerful tools to generate public and private keys. For this scenario, we use these tools to create a key and a self-signed certificate. In the following sections, we explain how to generate keys and certificates and how to use them to support SSL.

Keystore: A keystore that is created by using other cryptographic utilities can be used by the DataPower appliance. Refer to Appendix A, “External keys” on page 87, for instructions to create and use externally generated keystores with the DataPower objects that are described here.

Generating keys

To begin, we must create a public and private key pair. In this scenario, the public key is stored in a self-signed X.509 certificate. Note that in a production or real-world scenario, the certificate is signed by a trusted Certificate Authority (CA).

To create the certificate and private key:

1. By using the DataPower WebGUI, select **Administration** → **Crypto Tools** to navigate to the cryptographic tools page.
2. On the Generate Key tab of the Crypto Tools page, enter the details as shown in Figure 1-2 on page 6. Use the password `itsocorp` for the key. Then click the **Generate Key** button.

The screenshot shows the 'Crypto Tools' page with the 'Generate Key' tab selected. The page contains various input fields and radio buttons for configuring a key and certificate. The 'Generate Key' button is at the bottom left.

Field/Option	Value/State
LDAP (reverse) Order of RDNs	<input type="radio"/> on <input checked="" type="radio"/> off
Country Name (C)	US
State or Province (ST)	
Locality (L)	
Organization (O)	ITSOCorp
Organizational Unit (OU)	DP
Organizational Unit 2 (OU)	
Organizational Unit 3 (OU)	
Organizational Unit 4 (OU)	
Common Name (CN)	itsocorp-ssl *
RSA Key Length	1024 bits
File Name	itsocorp-ssl
Validity Period	365 days
Password	itsocorp
Password Alias	
Export Private Key	<input type="radio"/> on <input checked="" type="radio"/> off
Generate Self-Signed Certificate	<input checked="" type="radio"/> on <input type="radio"/> off
Export Self-Signed Certificate	<input checked="" type="radio"/> on <input type="radio"/> off
Generate Key and Certificate Objects	<input checked="" type="radio"/> on <input type="radio"/> off
Object Name	
Using Existing Key Object	

Generate Key

Figure 1-2 Generating a private key and self-signed certificate

3. In the confirmation window that opens, click **Confirm** to create the key and certificate. The appliance creates the key.

4. The window (Figure 1-3) that opens indicates successful completion of the operation. Click **Close** to return to the Generate Key page.



Figure 1-3 Successful key generation

Creating a Crypto Identification Credential

DataPower uses a *Crypto Identification Credential* to associate or match a public key and private key for use in cryptographic operations such as establishing SSL connections. In this section, we use the certificate and private key that we created in the previous section to build a new Crypto Identification Credential.

1. With the DataPower WebGUI, navigate to **Objects** → **Crypto Identification Credentials**.
2. On the Configure Crypto Identification Credentials page, click **Add** and configure a Crypto Identification Credential as shown in Figure 1-4 on page 8.

Values: The Crypto Key and Certificate fields use the certificate and key values from Figure 1-3 on page 7. In our example, this is `itsocorp-ssl`.

3. Click **Apply** to save the changes.

The screenshot shows the 'Configure Crypto Identification Credentials' page in the DataPower WebGUI. The page has a 'Main' tab and a title 'Configure Crypto Identification Credentials'. Below the title, there is a section for 'Crypto Identification Credentials'. At the top left of this section are 'Apply' and 'Cancel' buttons. The configuration area contains the following fields:

- Name:** A text field containing 'itsocorp-ssl' with a '*' icon to its right.
- Admin State:** Radio buttons for 'enabled' (selected) and 'disabled'.
- Crypto Key:** A dropdown menu showing 'itsocorp-ssl' with a '+' and '...' button to its right and a '*' icon.
- Certificate:** A dropdown menu showing 'itsocorp-ssl' with a '+' and '...' button to its right and a '*' icon.
- Intermediate CA Certificate:** An empty text area with a 'Delete' button and an 'Add' button with a '+' and '...' button below it.

Figure 1-4 Creating a Crypto Identification Credential

Creating a Crypto Validation Credential

DataPower uses *Validation Credentials* (ValCred) to validate digital signatures and received certificates. A Validation Credential is a list of certificate objects and is required by the DataPower SSL configuration.

To create a Validation Credential:

1. Select **Objects** → **Crypto Validation Credentials** to navigate to the Validation Credential creation page.
2. On the Configure Crypto Validation Credentials page, click **Add**.
3. Configure a new Validation Credential as shown in Figure 1-5 on page 9. Click **Apply** to save the changes.



Configure Crypto Validation Credentials

Main

Crypto Validation Credentials

Apply Cancel
Add from directory

Name	itsocorp-ssl *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
Certificates	<div><div></div><div>Delete itsocorp-ssl Add + ...</div></div>
Certificate Validation Mode	Match exact certificate or immediate issuer
Use CRLs	<input type="radio"/> on <input checked="" type="radio"/> off

Figure 1-5 Creating a Crypto Validation Credential

Creating a Crypto Profile

A Crypto Profile in the DataPower appliance associates a Crypto Identification with a Crypto Validation Credential. A Crypto Profile is required when configuring an SSL Proxy Profile, which we create in “Creating an SSL Proxy profile” on page 11.

To create a Crypto Profile:

1. In the DataPower WebGUI, select **Objects** → **Crypto** → **Crypto Profile**.
2. On the Configure Crypto Profile page, click **Add**.
3. Create a new Crypto Profile as shown in Figure 1-6 on page 10. Click **Apply** to save the changes.

Configure Crypto Profile

Main

Crypto Profile

Apply **Cancel**

Name	itsocorp-ssl *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
Identification Credentials	itsocorp-ssl + ...
Validation Credentials	itsocorp-ssl + ...
Ciphers	DEFAULT
Options	<input checked="" type="checkbox"/> OpenSSL default settings <input type="checkbox"/> Disabled SSL version 2 <input type="checkbox"/> Disabled SSL version 3 <input type="checkbox"/> Disabled TLS version 1 *
Send Client CA List	<input type="radio"/> on <input checked="" type="radio"/> off

Figure 1-6 Crypto Profile

Creating an SSL Proxy profile

To manage the SSL connection, the identification and validation objects are grouped together to form an *SSL Proxy Profile*.

To create an SSL Proxy Profile:

1. In the DataPower WebGUI, select **Objects** → **Crypto** → **SSL Proxy Profile**.
2. On the Configure SSL Proxy Profile page, click **Add**.
3. Create a new SSL Proxy Profile as shown in Figure 1-7.

Client Authentication Is Optional field: We select *on* for the Client Authentication Is Optional field to ensure that the SSL Proxy Profile does *not* prompt the requesting client to present a certificate for verification when the SSL connection is initiated.

Click **Apply**.

The screenshot shows the 'Configure SSL Proxy Profile' page in the DataPower WebGUI. The 'Main' tab is selected. The 'SSL Proxy Profile' section contains the following fields and values:

Field	Value
Name	itsocorp-ssl *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
Direction	reverse *
Reverse (Server) Crypto Profile	itsocorp-ssl + ... *
Server-side Session Caching	<input checked="" type="radio"/> on <input type="radio"/> off
Server-side Session Cache Timeout	300 seconds
Server-side Session Cache Size	20 entries (x 1024)
Client Authentication Is Optional	<input checked="" type="radio"/> on <input type="radio"/> off

Buttons: Apply, Cancel

Figure 1-7 Creating an SSL Proxy Profile

4. Click **Save Config** to save the running configuration.

Configuring a Web service proxy

In this section, we explain how to create a Web service proxy on the DataPower X150 appliance that will support the example scenario. With a Web service proxy, a Web service can be rapidly integrated with a DataPower appliance. By using only the Web Service Description Language (WSDL) file that describes the Web service, a nearly complete implementation that proxies the actual Web service can be constructed.

The following steps are required to build the Web service proxy:

1. Create the Web service proxy object.
2. Create and configure an HTTPS listener.

Web service proxy objects: Two Web service proxy objects are used in this paper for reasons of clarity and demonstrability. The first object uses Tivoli Access Manager for authentication and authorization. The second object uses Tivoli Directory Server. The use of separate objects helps to simplify the scenario. The objects are also configured by using different DataPower mechanisms to illustrate that alternative approaches can be used. The steps that are used in the subsequent sections are used again in Chapter 4, “AAA with the LDAP directory” on page 67.

Creating a Web service proxy

To create the Web service proxy:

1. From the main page of the DataPower WebGUI (Figure 1-8), select **Control Panel** → **Web Service Proxy** to navigate to the Web service proxy creation object.

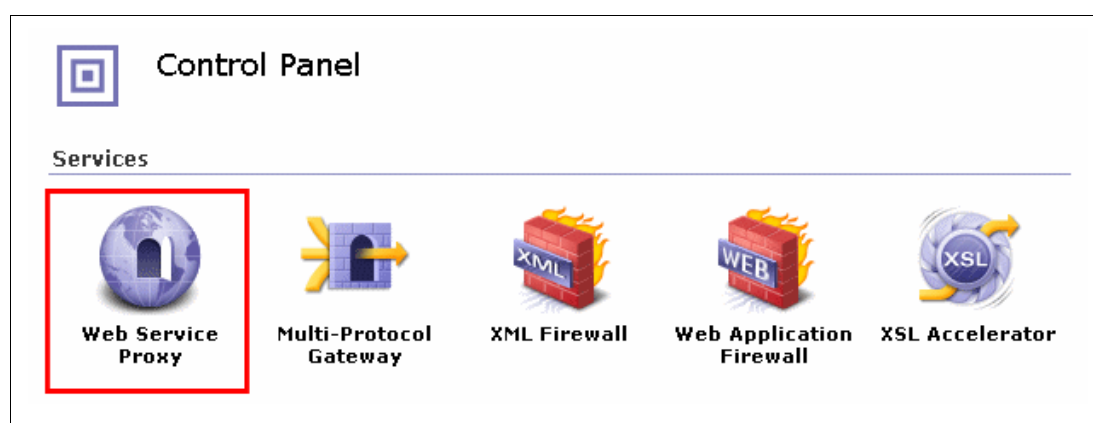


Figure 1-8 Selecting the Web Service Proxy icon

2. Click the **Add** button to create a new Web service proxy object.
3. On the Configure Web Service Proxy page (Figure 1-9), in the Web Service Proxy Name field, type ITS0_PRIMES. Click the **Upload** button to upload the WSDL file for the Primes Web service to the appliance.

Configure Web Service Proxy

SLM **WSDLs** Services Policy Proxy Settings Advanced Proxy Settings

Web Service Proxy Name
ITSO_PRIMES *

Apply Cancel

Web Service Proxy WSDLs

☐ Edit WSDL/Subscription
☒ Add WSDL
☐ Add UDDI Subscription
☐ Add WSRR Subscription

WSDL File URL

Configure Endpoints

local: (none) Upload... Fetch...

Browse UDDI

Figure 1-9 Uploading the WSDL file for the Primes Web service

- In the File Management window (Figure 1-10), click the **Browse...** button and select the file to upload. Then click **Upload**.

https://9.42.170.230:9090 - DataPower XI50 - File Management: - Microsoft Internet Explorer

Troubleshooting Enabled (The performance of the device may be impacted!)

File Management

Upload File to Directory **local:**

Source: ☒ File ☐ Java Key Store (Requires Sun JRE 1.4.2 or better)

File to upload:
C:\PrimeService.wsdl Browse... *

Save as:
PrimeService.wsdl *

☐ Overwrite Existing File

Upload Cancel

Figure 1-10 Uploading the Primes Web service WSDL file

- Ensure that the file uploads successfully and click **Continue** to proceed.

The Web service proxy that represents the Primes Web service is now created.

Creating an HTTPS listener

The Web service proxy must accept SSL connections. Therefore, we must create an SSL-enabled object called an *HTTPS Front Side Handler* to manage SSL connections to the Web service.

To build an HTTPS Front Side Handler:

1. Navigate to the **WSDLs** tab of the Web service proxy object that was just created.
2. If necessary, expand the plus (+) sign (circled in Figure 1-11) beneath “WSDL Source Location” to display the Web service’s local and remote endpoints.

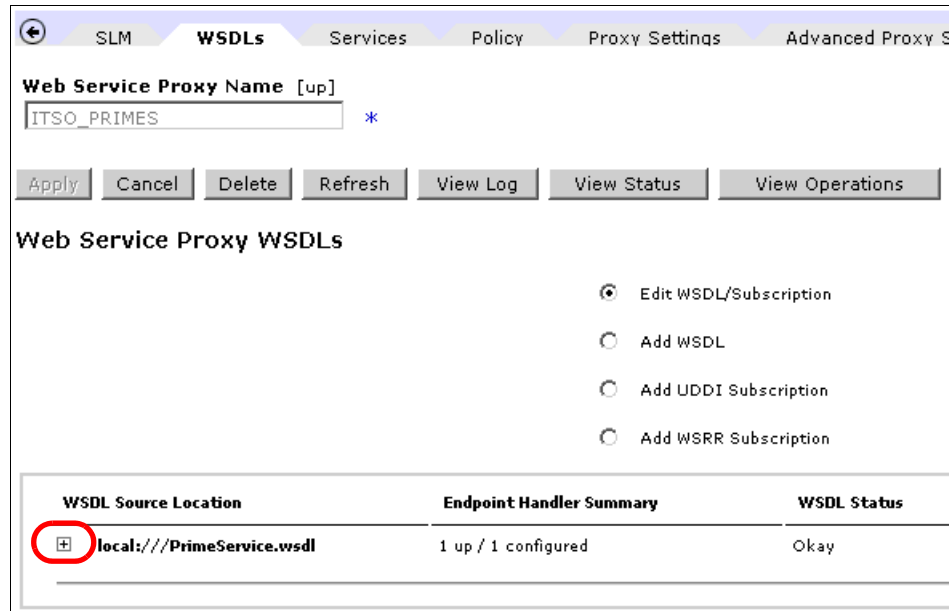


Figure 1-11 Displaying the Web service proxy endpoints

3. Edit the Hostname and Port field of the remote endpoint to be the host name or IP address of the machine that is hosting the Primes Web service.

4. Click the + button for Local Endpoint Handler to create a new SSL-enabled HTTP listener for the Web service proxy, and select the **HTTPS (SSL) Front Side Handler** option (Figure 1-12).

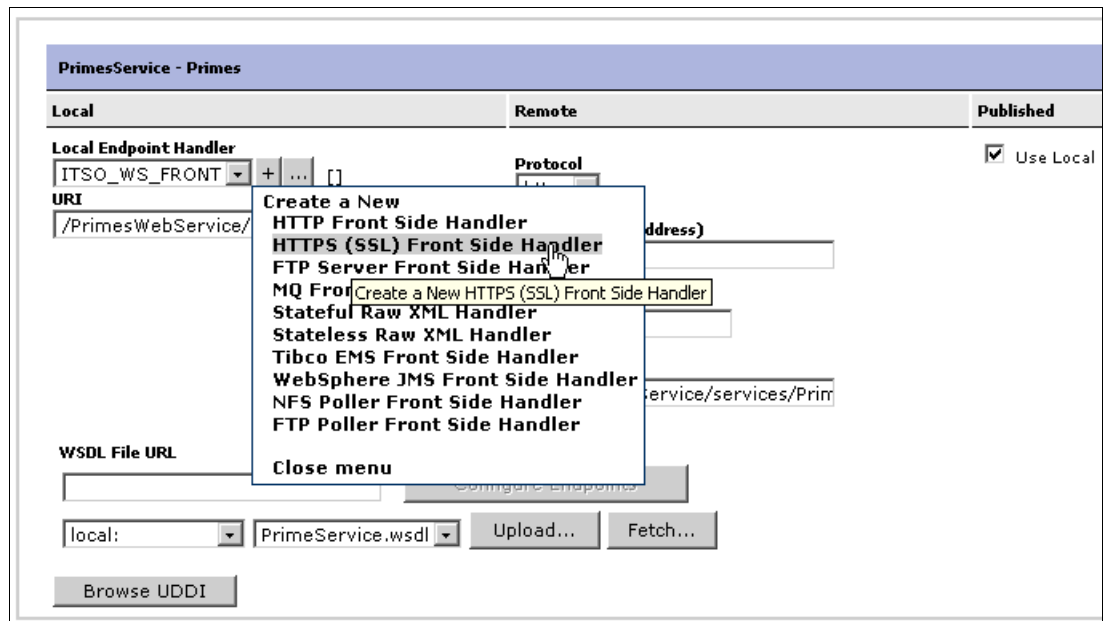


Figure 1-12 Creating a new HTTPS Front Side Handler

5. In the Configure HTTPS (SSL) Front Side Handler window, complete the fields as shown in Figure 1-13.

Port number: If you choose a port number for the HTTPS Front Side Handler that is already in use by another object, the DataPower WebGUI will display an error.

https://9.42.170.230:9090 - DataPower XI50 - Configure:HTTPS (SSL) Front Side Handler

Troubleshooting Enabled (The performance of the device may be impacted!)

Configure HTTPS (SSL) Front Side Handler

Main

HTTPS (SSL) Front Side Handler

Apply Cancel

Name	ITSO_WS_HTTPS_FSH *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
Comments	
Local IP Address	0.0.0.0 Select Alias *
Port Number	8443 *
HTTP Version to Client	HTTP/1.1
	<input checked="" type="checkbox"/> HTTP/1.0

Internet

Figure 1-13 Configuring the name and port of the HTTPS Front Side Handler

6. Scroll down until you see SSL Proxy. Select the SSL Proxy that was created in the previous section from the menu as shown in Figure 1-14.

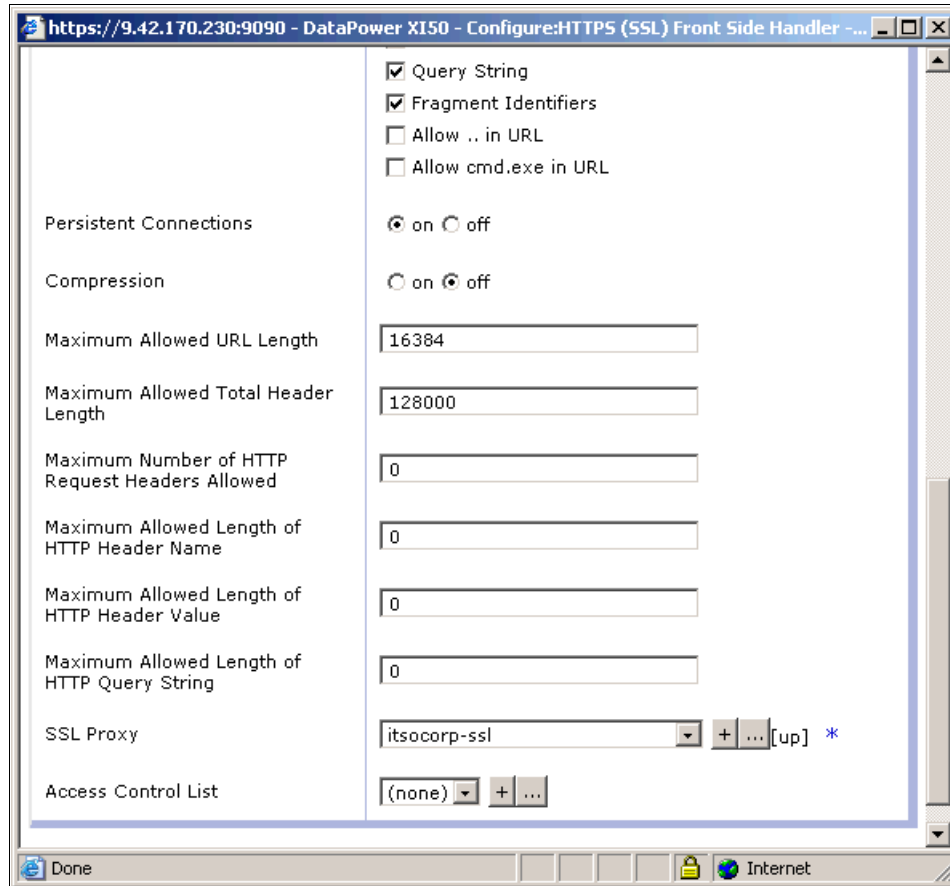


Figure 1-14 Selecting the SSL Proxy Profile

7. Click **Apply**.

The Front Side Handler of the Web service proxy is now configured.

1.3 Summary

In this chapter, we outlined the concepts of authentication and authorization. We also briefly demonstrated how you can use a DataPower appliance to perform these actions. A Web service proxy object was created to represent the Primes Web service of ITSOCorp.

The Web service proxy can now accept SSL connections from clients and proxy the request to the actual Primes Web service. However, these requests are only checked for general XML validity and conformance to the Primes Web service WSDL. No user identity or authorization checking is performed.

To regulate access to the service, we must define a processing policy for the Web service proxy. In subsequent chapters in this paper, we implement the remainder of the security policy for the Web service.



AAA concepts and policy creation

In this chapter, we describe the authentication, authorization, and audit (AAA) capabilities that are provided by the DataPower XS40 and XI50 devices.

We discuss the following topics:

- ▶ Concepts of AAA
- ▶ Creation of a processing policy for a Web service
- ▶ Configuration of the processing policy to use AAA

We use the processing policy that we create in this chapter to integrate the Tivoli software products of Tivoli Access Manager for e-business and Tivoli Directory Server in subsequent chapters.

2.1 AAA concepts

Authentication refers to the verification of a client to ensure that the client is who they claim to be. Conversely, *authorization* refers to the process of determining whether a client is allowed to access services or content as documents or files. In most security systems, authorization checking is performed only if authentication is successful. In Chapter 1, “Authentication and authorization” on page 1, we discuss these concepts in more detail, especially how they correspond to our example scenario.

Auditing refers to the recording of various portions of a transaction for different purposes such as tracking, error recovery, and so on. As part of the AAA policy, we consider logging information that can assert whether authentication and authorization steps have been successfully passed.

DataPower appliances provide different means for implementing AAA. For more detailed information, consult the DataPower XS40 or XI50 WebGUI documentation, which is available at the following Web addresses:

- ▶ XS40

<http://www-1.ibm.com/support/docview.wss?rs=2362&uid=swg24016091>

- ▶ XI50

<http://www-1.ibm.com/support/docview.wss?rs=2362&uid=swg24014405>

2.1.1 AAA policy

By having a *AAA policy*, you define the authentication, authorization, and auditing stages on a DataPower device®. The AAA policy is invoked by using a AAA action of a processing rule for dedicated service objects such as XML firewalls, multiprotocol gateways, or Web Service Proxies.

2.1.2 AAA action

A *AAA action* is a DataPower object that references a specific AAA policy. The action can be used on an existing processing rule. It can be implemented on every kind of DataPower service, including XML firewall, multiprotocol gateway, Web service proxy, or XSL proxy.

While creating a AAA action, a reference to an existing AAA policy must be provided. If a AAA policy does not exist, you must create one, in keeping with the requirements and constraints of the current application.

AAA policy and actions: A AAA policy can be referenced by different AAA actions, provided that they all, including the policies and actions, belong to the same domain.

2.1.3 Configuring a AAA policy

The configuration of a AAA policy entails the following steps. Notice that we identify those steps that are optional, depending upon the requirements of the application. The other steps are required.

1. Extract identity.

The purpose of this extraction is to answer the question: Who are you? Data may be extracted from different sources:

- SOAP message
- XML content
- HTTP header
- Automatically by DataPower or through a custom template (XSL stylesheet)

You can select several extraction methods. The DataPower AAA framework executes these methods in the following order:

- a. HTTP authentication header
- b. Password-carrying UsernameToken from the WS-Security header
- c. Derived-key UsernameToken element from the WS-Security header
- d. BinarySecurityToken element from the WS-Security header

Therefore, it is possible to use a single AAA policy that is in charge of extracting identity from different sources. These sources can implement different kinds of identification methods.

2. Authenticate.

In this step, you define the method that is used to authenticate the asserted identity. There are many methods for authentication, including the following options:

- Using a Lightweight Directory Access Protocol (LDAP)-enabled directory server, such as IBM Tivoli Directory Server
- Using IBM Tivoli Access Manager for e-business
- Using a DataPower AAA Info file

This is an XML file that contains authenticated identities and values related to them.

- Using a custom template

Authentication may be managed through an XSL stylesheet that is responsible for authenticating the asserted identity. An empty output of this XSL means “authentication failure” to the DataPower AAA framework.

3. (Optional) Map credentials.

It may not be possible to use the extracted identity to authorize the use of the requested resource, especially when authorization is managed by an external authority. Interoperability between systems (DataPower device and system in charge of authorization) might require mapping between credentials. The following supported methods can be used for this mapping:

- IBM Tivoli Federated Identity Manager
- AAA Info file
- XPath query
- Custom template
- WS-SecureConversation

4. Extract resource.

The goal of this step is to retrieve the requested resource service that is essential to complete authorization. The purpose of this extraction is to answer the question: What do you want to do? The following items, among others, are proposed to identify a resource:

- Recover resource from local name of the requested of the message
- HTTP operation
- XPath expression

5. (Optional) Map resource.

Interoperability between systems (DataPower device and system in charge of authorization) might require mapping between resources. The following supported methods and formats can be used for this mapping:

- IBM Tivoli Access Manager for e-business objectspace representation
- AAA Info file
- XPath query
- Custom template

6. Authorize.

Credentials and resources, which both might have been remapped, are submitted to the authority in charge of dealing with authorization. This authority might be the DataPower device itself or an external entity, such as the following most prominent ones:

- IBM Tivoli Access Manager for e-business
- An LDAP server, as IBM Tivoli Directory Server
- DataPower AAA Info file
- Custom template

Authorization might be managed through an XSL stylesheet. Two different outputs are also possible for this XSL:

- The <approved/> element, which means “authorization success” to the DataPower AAA framework
- The <declined/> element, which means “authorization failure”

7. (Optional) Perform post processing.

Tasks might be set as a final step of the AAA policy. For instance, it is possible to add a WS-Security UserName token in the message.

8. (Optional) Audit.

An audit consists of logging information that might assert that both authentication and authorization have succeeded or failed.

2.1.4 Processing a AAA policy

Figure 2-1 shows a typical scheme of processing a AAA policy:

- ▶ The input is an XML or SOAP message.
- ▶ The output is either an error message or the XML/SOAP message that is provided as input of the processing of the AAA policy.

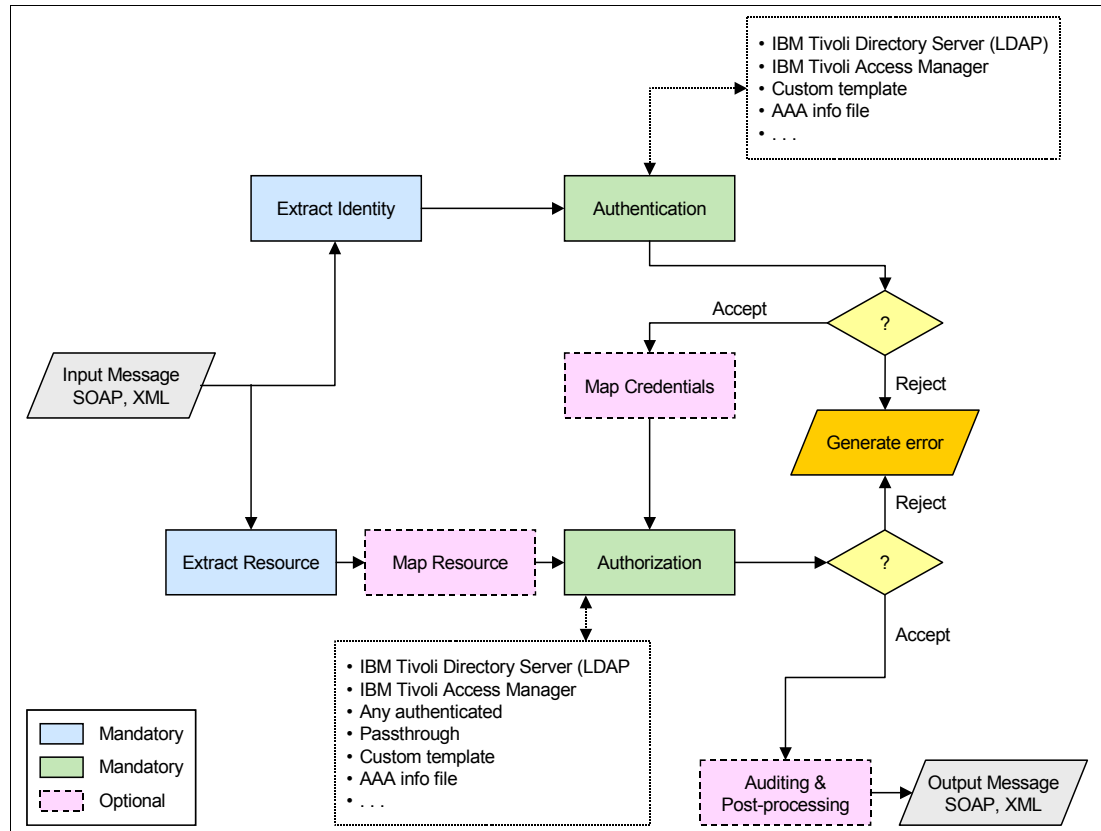


Figure 2-1 Processing a AAA policy

2.2 Configuring the Primes Web service proxy policy

To describe how a AAA policy can be used in a DataPower appliance, we consider the Web service proxy that is described in Figure 1-1 on page 2.

2.2.1 Creating a processing rule and AAA action

When a Web service proxy object is created, default request and response processing rules are created. As with other DataPower service objects, such as XML firewalls, additional processing rules can be created. A key difference of Web service proxy objects is that rules can be created to process messages at the WSDL, service, port, and operation levels of the Web service. Additional rules can also be created at the Web service proxy level.

Rule per message: Only one rule is executed per message. The *SLM* and *validate* actions are always executed before any user-defined rules.

In this scenario, we define a processing rule at the WSDL level. To create the new rule:

1. In the DataPower WebGUI, navigate to the **Policy** tab of the ITSO_PRIMES Web service proxy.
2. On the Policy tab (Figure 2-2), click the **Add Rule** button.

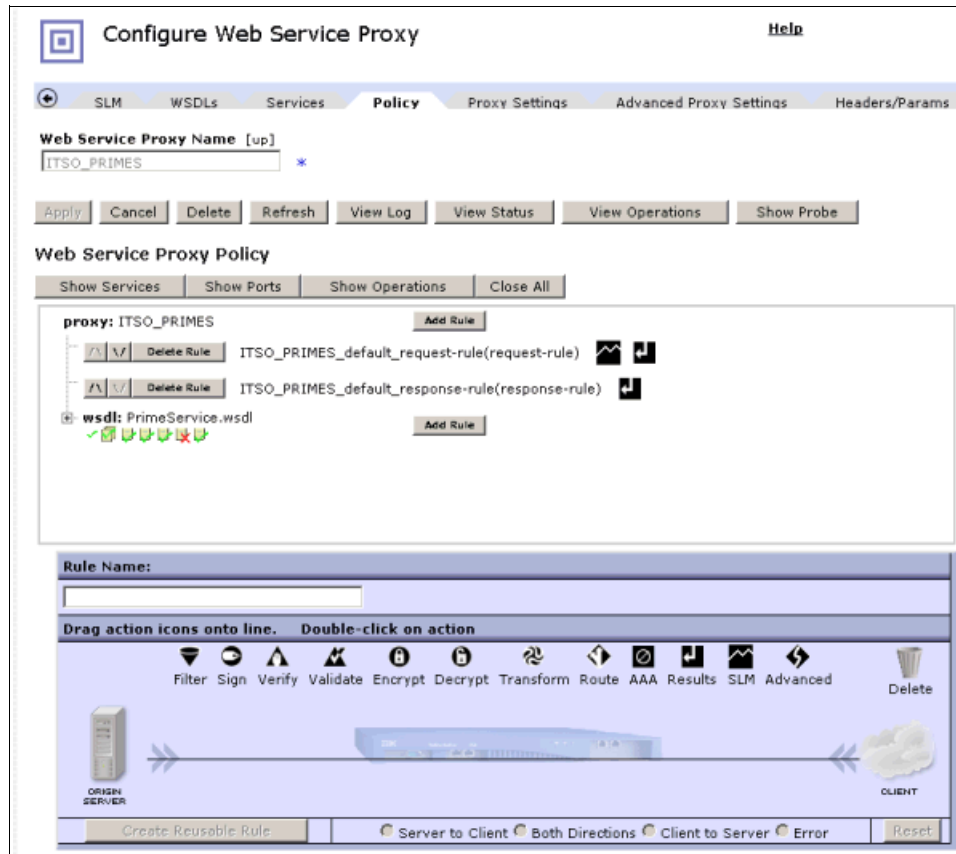


Figure 2-2 Web Service Proxy policy page

3. In the shaded section as shown in Figure 2-3, complete these steps:
 - a. Drag the **AAA icon** onto the new processing rule.
 - b. Select the **Client to Server** radio button. The Client to Server setting ensures that the AAA action is only used on *incoming* requests.
 - c. Double-click the **AAA action** to configure it.

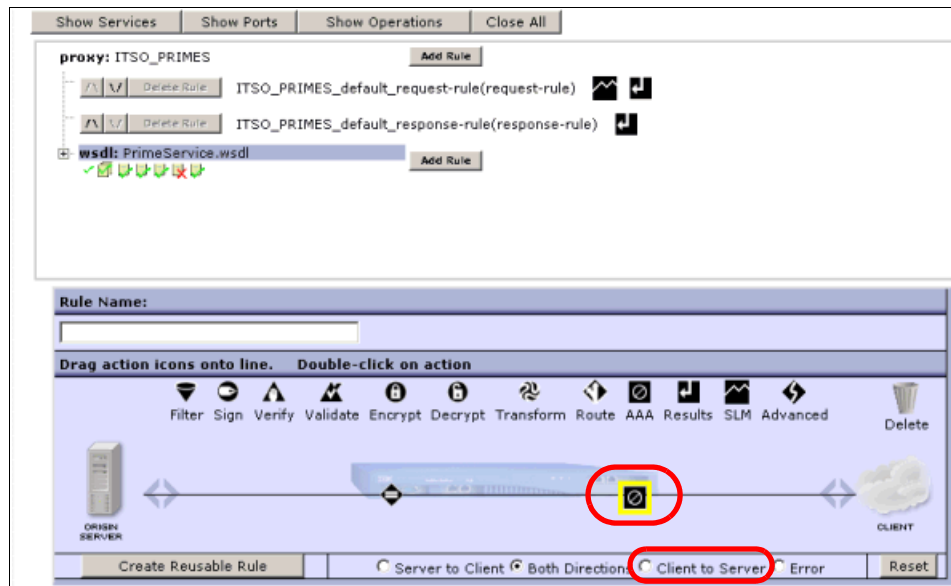


Figure 2-3 Adding a AAA action to the processing rule

4. In the Configure AAA Action window, configure the AAA action as shown in Figure 2-4 and click **Done**.

https://9.42.170.230:9090 - DataPower XI50 - Configure AAA Action - Microsoft Internet ...

DATAPOWER XI50

Configure AAA Action

Basic Advanced

Input

Input | (auto) (auto)

Options

☒ AAA

AAA Policy | (none) + ...

Output

Output | (auto) (auto)

Delete Done Cancel

Done Internet

Figure 2-4 Configuring the AAA action

The AAA action has been added to the new processing rule, as shown in Figure 2-5.

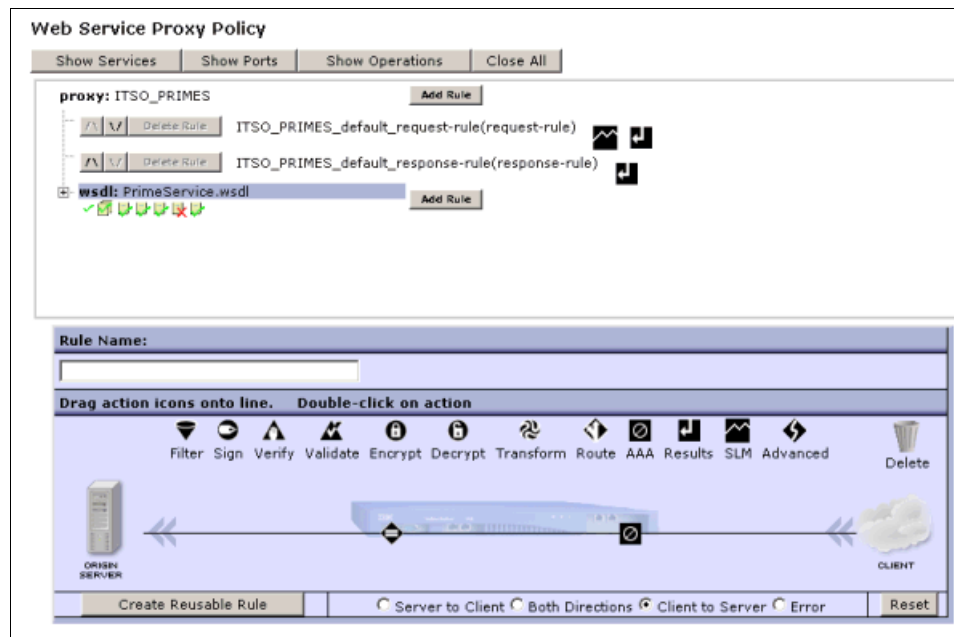


Figure 2-5 AAA action added to the processing rule

Note: The AAA action has not been associated with a AAA policy. We perform this action in the following chapters.

2.2.2 Adding a Results action

The final step in creating the processing rule is to add a Results action:

1. Drag the **Results** icon to the processing rule, as shown in Figure 2-6. Double-click the **Results** icon.

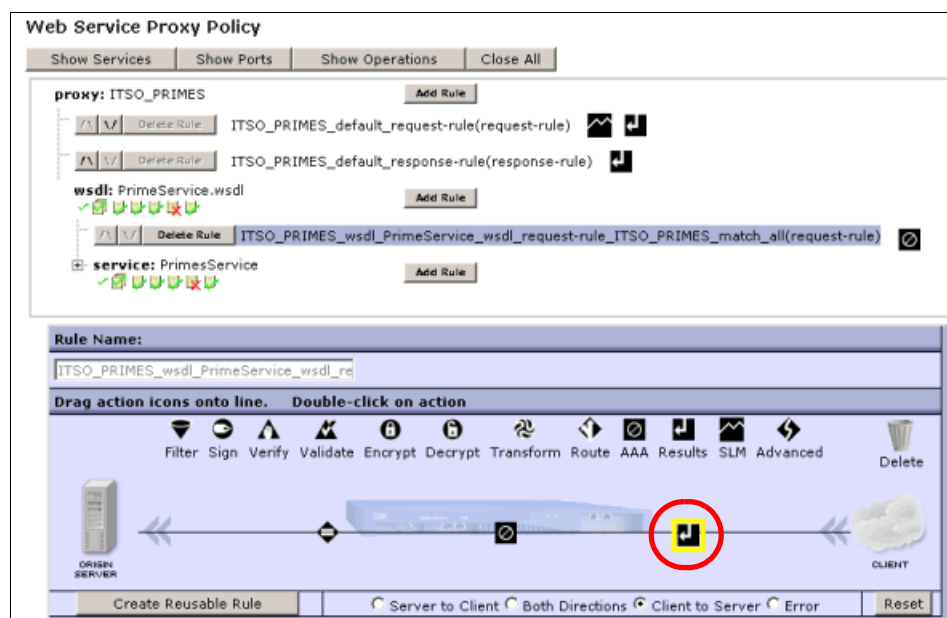


Figure 2-6 Adding a Results action

2. On the Configure Results Action page (Figure 2-7), click **Done**.

The screenshot shows the 'Configure Results Action' configuration page in the DataPower X150 interface. The page is divided into two tabs: 'Basic' (selected) and 'Advanced'. The 'Input' section contains a text field with the value '(auto)' and a dropdown menu also set to '(auto)'. The 'Options' section features a 'Results' icon and a 'Send Results Asynchronously' section with two radio buttons: 'on' and 'off', with 'off' being the selected option. Below this is a 'Destination' text field. The 'Output' section includes a text field and a dropdown menu set to '(auto)'. At the bottom of the page are three buttons: 'Delete', 'Done', and 'Cancel'.

Figure 2-7 Configuring the Results action

2.2.3 Committing the processing rule

The processing rule for the Primes Web service has been created. Click **Apply** and then click **Save Config** to commit the changes that you made to the ITSO_PRIMES Web service proxy of the domain configuration.

2.3 Summary

In this chapter, we described the concept of a AAA policy with respect to DataPower SOA appliances. In short, DataPower AAA policy objects provide a powerful mechanism to enforce access control for SOA deployments. The wide ranging support of user identification, authentication, and authorization techniques of the AAA policy object offers a great deal of flexibility.

To support ITSOCorp's Primes Web service, a processing rule that contained a AAA action was created. This AAA action is associated with a AAA policy in subsequent chapters.



AAA with Tivoli Access Manager

The powerful access control mechanisms of Tivoli Access Manager for e-business offer a comprehensive access management solution and complement the strong Web services capabilities of the DataPower appliance. In this chapter, we explain how to configure a representation of the Primes Web service in Tivoli Access Manager as well as build a Tivoli Access Manager access control policy to regulate access to the service.

In addition, we explain the steps that are required to configure the DataPower appliance to use a Tivoli Access Manager infrastructure. This task includes generating the necessary configuration files and keystores, as well as integrating these configuration artifacts into a DataPower processing policy.

3.1 IBM Tivoli Access Manager for e-business

Tivoli Access Manager for e-business is the flagship enterprise security product for IBM and is recognized as an industry leader. Tivoli Access Manager offers centralized policy management, scalability, and flexibility to provide enterprises with a secure and easily managed network infrastructure.

3.1.1 Tivoli Access Manager and DataPower

To use Tivoli Access Manager to perform authentication and authorization, a Tivoli Access Manager (client) object must be configured on the DataPower appliance. When Tivoli Access Manager is specified in a AAA processing policy, the Tivoli Access Manager client is used to communicate with the Tivoli Access Manager server infrastructure.

Usage of a Tivoli Access Manager client entails the following steps:

1. Creating the necessary Tivoli Access Manager configuration files
2. Using the files to create a DataPower Tivoli Access Manager client

The following Tivoli Access Manager files are created:

- ▶ A configuration file that contains the Tivoli Access Manager infrastructure information
- ▶ A keystore file that contains the certificates that are necessary for communication with the Tivoli Access Manager server
- ▶ A stash file that contains the password for the keystore

A DataPower Tivoli Access Manager client object uses these files to interact with the Tivoli Access Manager server that is specified in the configuration file. The Tivoli Access Manager client configuration files can be used by multiple Tivoli Access Manager clients in the same DataPower domain, but cannot be shared among other domains.

In the remainder of this section, we discuss the Tivoli Access Manager infrastructure of the Management zone as illustrated in Figure 3-1.

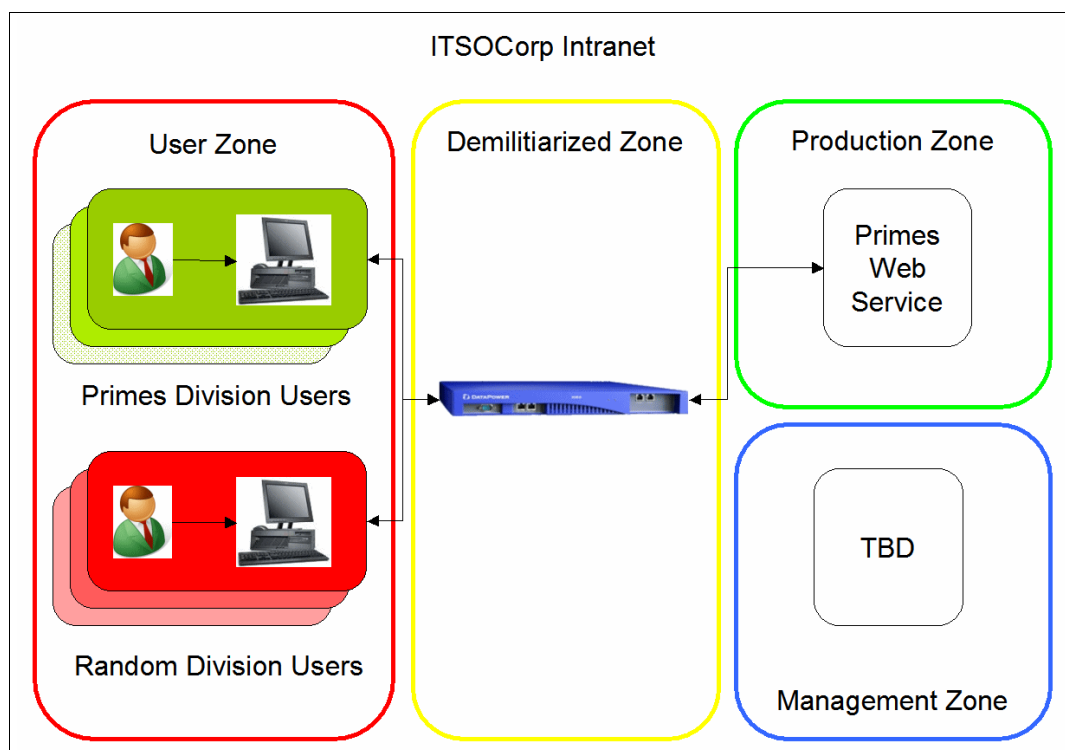


Figure 3-1 Intranet deployment architecture

3.1.2 Tivoli Access Manager and Web services overview

To enforce the security policy defined in Chapter 1, “Authentication and authorization” on page 1, by using Tivoli Access Manager, you must design a virtual representation the Primes Web service. In a Tivoli Access Manager environment, the entity that represents resources like the Web service is called a *protected object space*. Tivoli Access Manager applies the defined security policy to the protected object space by associating appropriate access control policies to the objects in the object space.

Functional Tivoli Access Manager environment: In the following sections, we assume that you have a functional Tivoli Access Manager environment installed and configured. A functional Tivoli Access Manager environment consists of a configured Tivoli Access Manager Policy Server and at least one configured Tivoli Access Manager Authorization Server. For more information about installing and configuring Tivoli Access Manager, refer to the Tivoli Access Manager Information Center at the following Web address:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame.doc/welcome.htm>

Example 3-1 shows the relevant sections of the Primes Web service Web Service Description Language (WSDL) file.

Example 3-1 Relevant sections of the Primes Web service WSDL file

```
...
<wsdl:portType name="Primes">
<wsdl:operation name="getPrime">
...
</wsdl:operation>
<wsdl:operation name="random">
...
</wsdl:operation>
</wsdl:portType>
...
<wsdl:binding name="PrimesSoapBinding type="intf:Primes">
...
<wsdl:service name="PrimesService">
<wsdl:port binding="intf:PrimesSoapBinding" name="Primes">
<wsdlsoap:address location="http://itsolab1:9080/PrimesWebService/services/Primes"/>
</wsdl:port>
</wsdl:service>
...
```

By inspecting the WSDL for the Primes Web service, you can see that it defines a *service* called *PrimesService*. There is a single network endpoint or *port* for the service, named *Primes*. The Primes portType defines multiple operations, called *getPrime*, *random*, and *nextPrime*.

3.1.3 Designing the Tivoli Access Manager object space

In this scenario, you begin by creating a root object space called */itso/services*. This object space contains the Web service objects that represent the services that are provided by ITSO Corp. The subsequent sections explain how to create the Tivoli Access Manager objectspace for the Web service.

Service

To represent the Primes Web service, a Tivoli Access Manager object container called *PrimesService* is created under */itso/services*. The container */itso/services/PrimesService* stores the objects that represent the *services* that are offered by the Primes Web service. A container object is used since a Web service might define multiple service bindings.

Port

The service defines a single port called *Primes*. This port is represented as an object container called */itso/services/PrimesService/Primes*. Again, you use an object container to represent the port since a service can define multiple network endpoints.

Operations

The service defines the operations *getPrime*, *random*, and *nextPrime*. The objects *getPrime*, *random* and *nextPrime* are created under the container object */itso/services/Primes/PrimesService/Primes*.

Table 3-1 lists the object space elements that correspond to the Primes Web service.

Table 3-1 Primes Web service WSDL elements and Tivoli Access Manager object space

WSDL element	Primes Web service definition	Tivoli Access Manager object space
Service	PrimesService	/itso/services/PrimesService
Port	Primes	/itso/services/PrimesService/Primes
Operation or operations	getPrime	/itso/services/PrimesService/Primes/getPrime
	random	/itso/services/PrimesService/Primes/random
	nextPrime	/itso/services/PrimesService/Primes/nextPrime

3.1.4 Configuring the Tivoli Access Manager object space

You use the **pdadmin** configuration utility of Tivoli Access Manager to create the object space. Example 3-2 shows the commands to create the object space that represents the Primes Web service. Note that an *application object container* is represented in the Tivoli Access Manager object space notation by the number 14. Conversely, Tivoli Access Manager designates an *application object* by using the number 15.

Alternative: You can also perform the operations in Example 3-2 by using the Tivoli Access Manager Web Portal Manager (WPM) application. WPM is a WebSphere Application Server-based application that enables Web-based Tivoli Access Manager administration.

Example 3-2 Tivoli Access Manager object space commands for the Primes Web service

```
pdadmin sec_master> objectspace create /itso "ITS0 object space" 14
pdadmin sec_master> object create /itso/services "ITS0 Web service container" 14
ispolicyattachable yes
pdadmin sec_master> object create /itso/services/Primes "Primes Web Service
container" 14 ispolicyattachable yes
pdadmin sec_master> object create /itso/services/Primes/PrimesService "Primes Web
Service service container" 14 ispolicyattachable yes
pdadmin sec_master> object create /itso/services/Primes/PrimesService/Primes
"Primes Web Service port container" 14 ispolicyattachable yes
pdadmin sec_master> object create
/itso/services/Primes/PrimesService/Primes/getPrime "Primes Web service getPrime
operation object" 15 ispolicyattachable yes
pdadmin sec_master> object create /itso/services/Primes/PrimesService/Primes/random
"Primes Web service random operation object" 15 ispolicyattachable yes
pdadmin sec_master> object create
/itso/services/Primes/PrimesService/Primes/nextPrime "Primes Web service nextPrime
operation object" 15 ispolicyattachable yes
```

User and group creation

Next you create the user identities that are defined in 1.2.2, “Securing the Web service” on page 3, in Tivoli Access Manager. You also add them to appropriate Tivoli Access Manager user groups. Example 3-3 displays the commands to create the users and groups.

Example 3-3 Tivoli Access Manager user and group creation by using pdadmin

```
pdadmin sec_master> user create johndoe cn=johndoe,dc=itso,dc=ibm,dc=com johndoe
“John Doe” passwd0rd
pdadmin sec_master> user modify johndoe account-valid yes
pdadmin sec_master> user create janedoe cn=janedoe,dc=itso,dc=ibm,dc=com janedoe
“Jane Doe” passwd1rd
pdadmin sec_master> user modify janedoe account-valid yes
pdadmin sec_master> group create itso-primes cn=itso-primes,dc=itso,dc=ibm,dc=com
itso-primes
pdadmin sec_master> group modify itso-primes description “Primes division users”
pdadmin sec_master> group modify itso-primes add johndoe
pdadmin sec_master> group create itso-random cn=itso-random,dc=itso,dc=ibm,dc=com
itso-random
pdadmin sec_master> group modify itso-random description “Random division users”
pdadmin sec_master> group modify itso-random add janedoe
```

3.1.5 Applying the security policy

Now that you have a Tivoli Access Manager object space that corresponds to the Primes Web service, you can define a security policy and configure Tivoli Access Manager (and DataPower) to enforce that policy.

In our example scenario, the employees of the Primes division of ITSOCorp require access to the operations *getPrime* and *nextPrime* of the Primes Web service. Conversely, the employees of the Random division require access to the *random* operation of the Web service. By using Tivoli Access Manager, you can provide this fine-grained access control.

Tivoli Access Manager provides powerful mechanisms for managing access to resources. With access control list (ACL) policies, the system administrator can specify which users and groups are permitted to access resources as well as the operations they can perform on those resources. By using ACLs, highly configurable and flexible security policies can be enforced.

Example 3-4 shows the Tivoli Access Manager **pdadmin** commands that define the policy for the *getPrime* operation.

Example 3-4 Tivoli Access Manager pdadmin commands to apply the security policy

```
pdadmin sec_master> action group create WebService
pdadmin sec_master> action create i execute services WebService
pdadmin sec_master> acl create getPrime
pdadmin sec_master> acl modify getPrime set group itso-primes T[WebService]i
pdadmin sec_master> acl attach /itso/services/Primes/PrimesService/Primes/getPrime
getPrime
pdadmin sec_master> acl create random
pdadmin sec_master> acl modify random set group itso-random T[WebService]i
pdadmin sec_master> acl attach /itso/services/Primes/PrimesService/Primes/random
random
```

3.2 Creating the Tivoli Access Manager AAA policy

In this section, we explain the steps to create and configure a AAA policy. The AAA policy is created by using a wizard that takes the user through the stages of AAA processing that is described in Chapter 2, “AAA concepts and policy creation” on page 19. The wizard is launched by configuring a processing rule’s AAA *action*. The processing rule and AAA action that are created in Chapter 2, “AAA concepts and policy creation” on page 19, are used for the remainder of this chapter.

Workflow options: The AAA policy can be created via two separate workflows. In this chapter, we build the AAA policy by using a wizard. The alternative approach is explained in 4.4, “Creating Tivoli Directory Server AAA policies” on page 75.

During the process of creating the AAA policy, the initial configuration of a Tivoli Access Manager client object is completed. Specifically, the AAA policy wizard initiates the creation of the required Tivoli Access Manager configuration files. Since the file creation process takes time to complete, the following process occurs:

1. Initiate the Tivoli Access Manager configuration file process.
2. Create the DataPower Tivoli Access Manager client object. The DataPower Tivoli Access Manager client object is unconfigured at this point to use the Tivoli Access Manager configuration files since they do not exist at this stage.
3. Complete the AAA policy.
4. Ensure that the Tivoli Access Manager configuration files are created successfully.
5. Update the DataPower Tivoli Access Manager client object to use these files.

3.2.1 Creating the AAA policy object

To create the Tivoli Access Manager AAA policy for the Web service proxy of the Primes Web service:

1. Navigate to the **Policy** tab of the ITSO_PRIMES Web service proxy in the DataPower WebGUI. Expand **wsdl: PrimeService.wsdl** to display the processing rule with the AAA action (Figure 3-2).

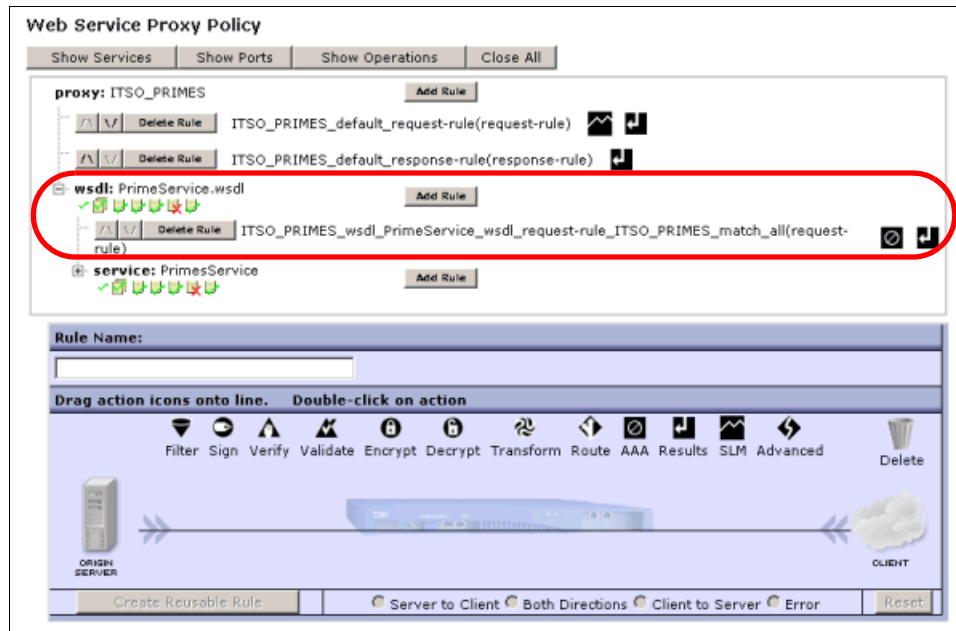


Figure 3-2 WSDL level processing rule

2. Click the **WSDL level processing rule** and then double-click the **AAA action** as shown in Figure 3-3.

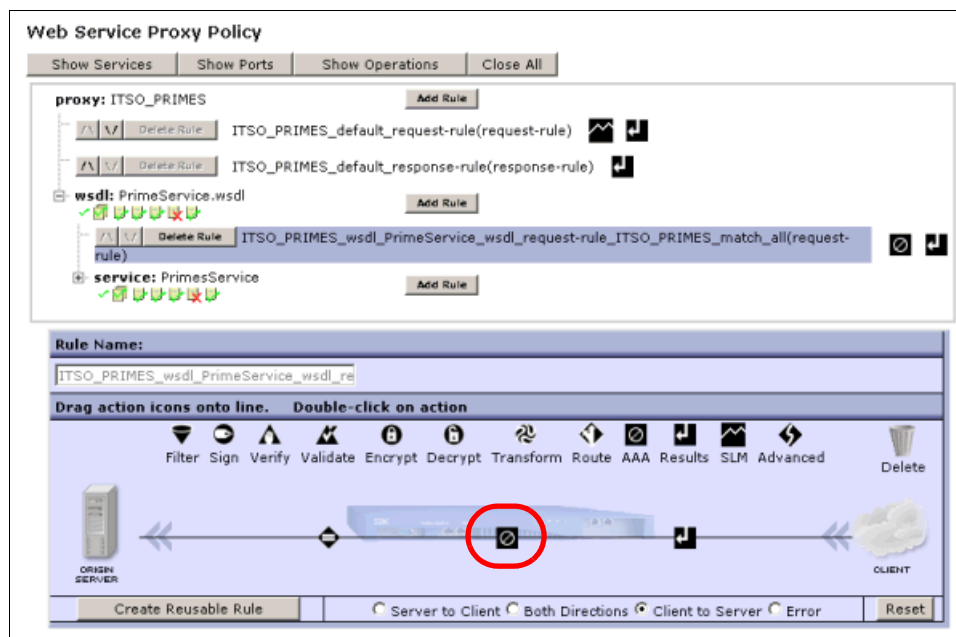


Figure 3-3 Editing the AAA action

3. On the Configure AAA Action page (Figure 3-4), click the + button to create a new AAA policy.

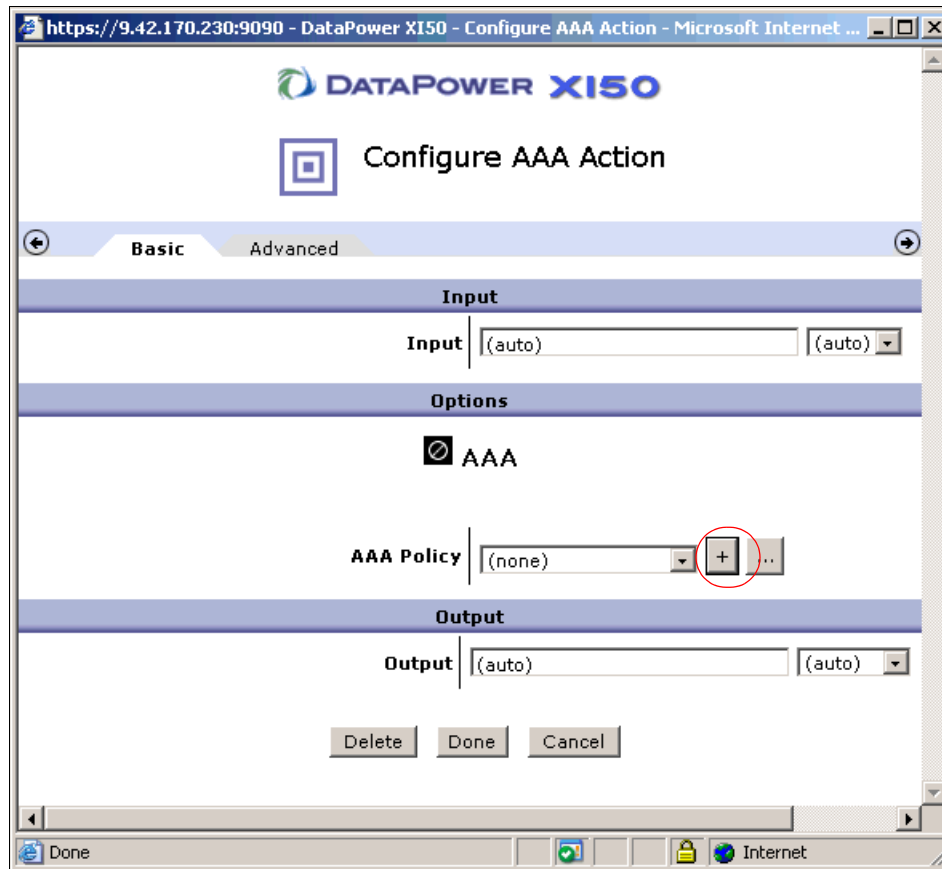


Figure 3-4 Creating a new AAA policy

4. On the Configure an Access Control Policy page (Figure 3-5), enter a name for the policy. We type the name `ITSO_PRIMES_TAM_AAA`. Then click **Create**.

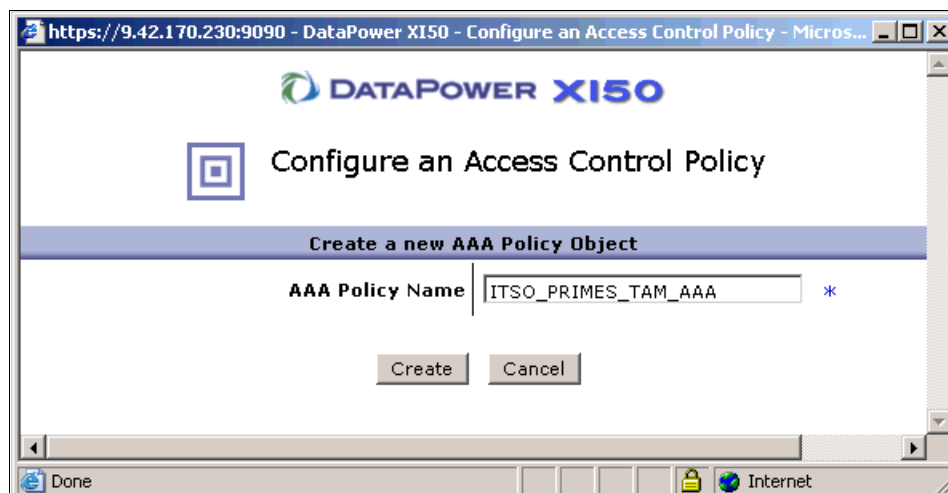


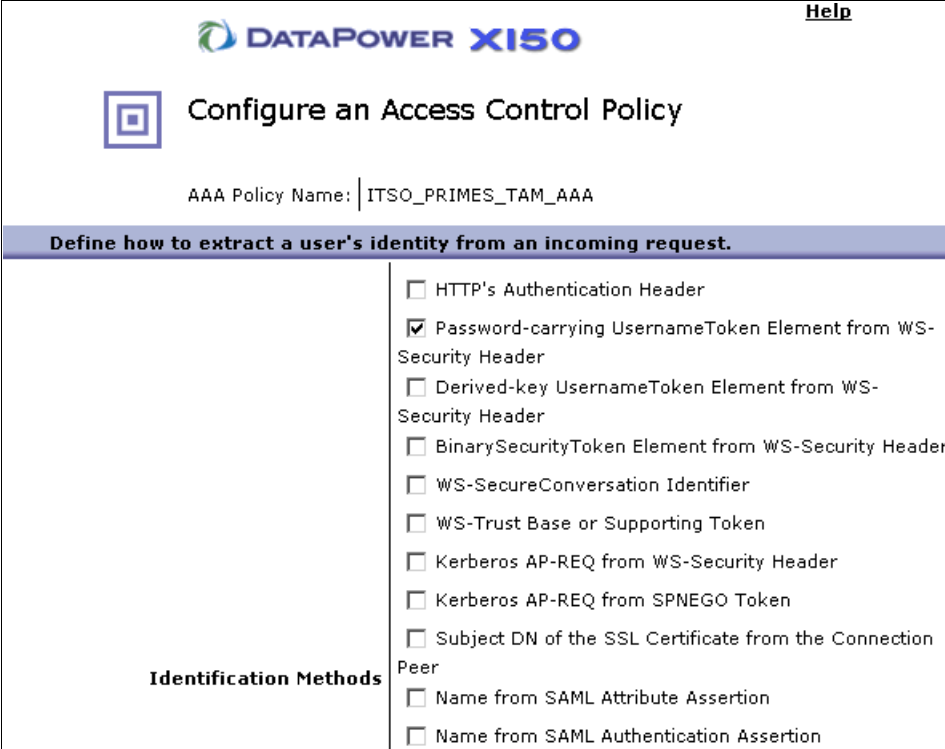
Figure 3-5 Entering the name of the AAA policy

The Extract Identity page opens.

3.2.2 Extracting the identity

Defining the mechanism that is needed to extract an identity from the message is the first stage in configuring a AAA policy. In our scenario, the user's credentials are stored in a WS-Security UsernameToken. The UsernameToken is transported in the header of the SOAP message.

Following on from the previous section, on the Configure an Access Control Policy page (Figure 3-6), select the **Password-carrying UsernameToken Element from WS-Security Header** option. Then click **Next**.



DATAPOWER X150 [Help](#)

Configure an Access Control Policy

AAA Policy Name:

Define how to extract a user's identity from an incoming request.

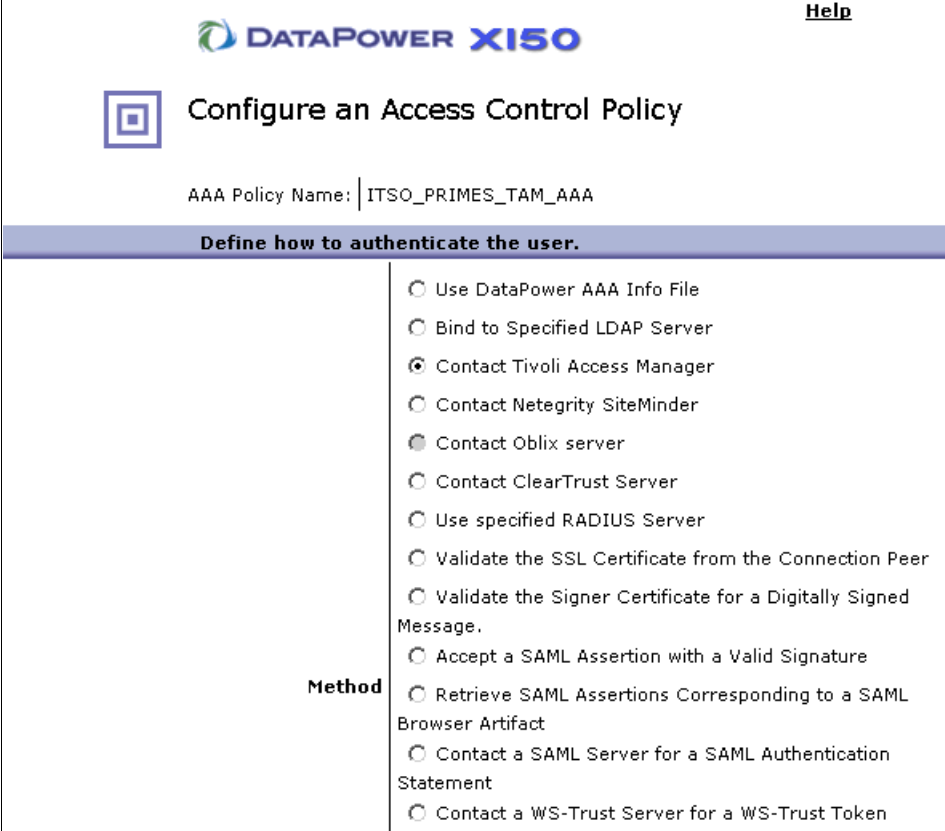
Identification Methods
<input type="checkbox"/> HTTP's Authentication Header
<input checked="" type="checkbox"/> Password-carrying UsernameToken Element from WS-Security Header
<input type="checkbox"/> Derived-key UsernameToken Element from WS-Security Header
<input type="checkbox"/> BinarySecurityToken Element from WS-Security Header
<input type="checkbox"/> WS-SecureConversation Identifier
<input type="checkbox"/> WS-Trust Base or Supporting Token
<input type="checkbox"/> Kerberos AP-REQ from WS-Security Header
<input type="checkbox"/> Kerberos AP-REQ from SPNEGO Token
<input type="checkbox"/> Subject DN of the SSL Certificate from the Connection Peer
<input type="checkbox"/> Name from SAML Attribute Assertion
<input type="checkbox"/> Name from SAML Authentication Assertion

Figure 3-6 Defining the mechanism for extracting identity

3.2.3 Authenticating requests


The next stage of processing performed by the AAA policy is authentication. As described in 2.1.3, “Configuring a AAA policy” on page 20, the DataPower appliance provides several means for authenticating a request. In this section, on the Define authentication page (Figure 3-7), complete the following actions:

1. Select the **Contact Tivoli Access Manager** option.



DATAPOWER X150

Help

 Configure an Access Control Policy

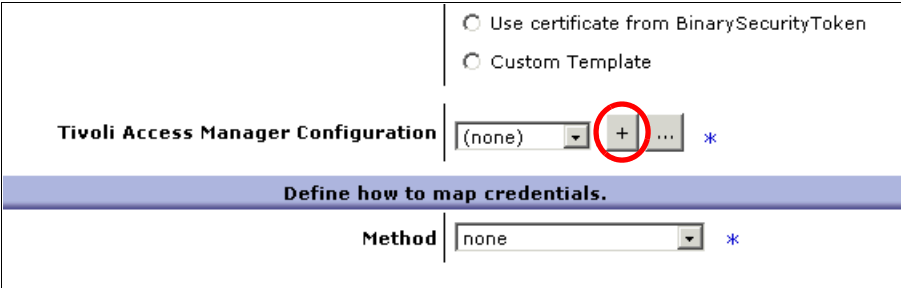
AAA Policy Name: ITSQ_PRIMES_TAM_AAA

Define how to authenticate the user.

Method	
<input type="radio"/>	Use DataPower AAA Info File
<input type="radio"/>	Bind to Specified LDAP Server
<input checked="" type="radio"/>	Contact Tivoli Access Manager
<input type="radio"/>	Contact Netegrity SiteMinder
<input type="radio"/>	Contact Oblix server
<input type="radio"/>	Contact ClearTrust Server
<input type="radio"/>	Use specified RADIUS Server
<input type="radio"/>	Validate the SSL Certificate from the Connection Peer
<input type="radio"/>	Validate the Signer Certificate for a Digitally Signed Message.
<input type="radio"/>	Accept a SAML Assertion with a Valid Signature
<input type="radio"/>	Retrieve SAML Assertions Corresponding to a SAML Browser Artifact
<input type="radio"/>	Contact a SAML Server for a SAML Authentication Statement
<input type="radio"/>	Contact a WS-Trust Server for a WS-Trust Token

Figure 3-7 Selecting Tivoli Access Manager for authentication

2. Click the + button to select the Tivoli Access Manager configuration (Figure 3-8).



☐ Use certificate from BinarySecurityToken

☐ Custom Template

Tivoli Access Manager Configuration: (none) + ... *

Define how to map credentials.

Method: none *

Figure 3-8 Defining the Tivoli Access Manager configuration

The Configure IBM Tivoli Access Manager page (Figure 3-9) opens on which you can both create the Tivoli Access Manager configuration files and configure the DataPower Tivoli Access Manager client to use these files.

Figure 3-9 Creating the Tivoli Access Manager configuration files

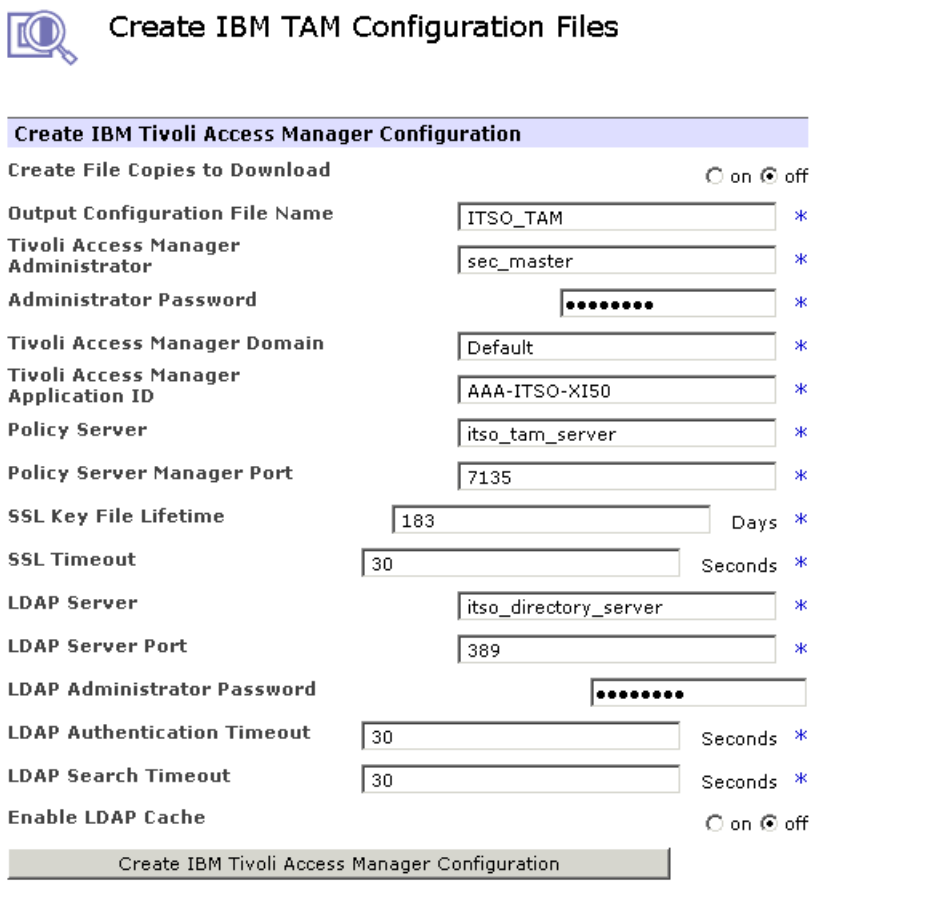
In the following subsections, we describe the process of creating and configuring a Tivoli Access Manager client object on the DataPower appliance. The process entails the following actions:

- ▶ Create the Tivoli Access Manager configuration files for the DataPower Tivoli Access Manager client. These files contain the information that is required by the client to communicate with the Tivoli Access Manager server.
- ▶ Create the DataPower Tivoli Access Manager client object itself.

Creating the Tivoli Access Manager configuration files

To create the Tivoli Access Manager configuration files:

1. On the Configure IBM Tivoli Access Manager page (Figure 3-9), in the Name field, type ITSO_TAM. Then click **Create IBM TAM Configuration Files**.
2. On the Create IBM TAM Configuration Files page (Figure 3-10), enter the necessary information and click the **Create IBM Tivoli Access Manager Configuration** button.



Create IBM TAM Configuration Files

Create IBM Tivoli Access Manager Configuration

Create File Copies to Download ☐ on ☒ off

Output Configuration File Name *

Tivoli Access Manager Administrator *

Administrator Password *

Tivoli Access Manager Domain *

Tivoli Access Manager Application ID *

Policy Server *

Policy Server Manager Port *

SSL Key File Lifetime Days *

SSL Timeout Seconds *

LDAP Server *

LDAP Server Port *

LDAP Administrator Password *

LDAP Authentication Timeout Seconds *

LDAP Search Timeout Seconds *

Enable LDAP Cache ☐ on ☒ off

Create IBM Tivoli Access Manager Configuration

Figure 3-10 Creating the Tivoli Access Manager configuration files

More information: For more information about these configuration fields, refer to the DataPower reference support Web page at the following address or click the **Help** link:
<http://www-306.ibm.com/software/integration/datapower/support/>

3. In the confirmation window (Figure 3-11) that opens, click **Confirm** to begin the Tivoli Access Manager file creation process.

Note: When this confirmation window opens, the Create IBM TAM Configuration Files page (Figure 3-10) becomes unavailable.

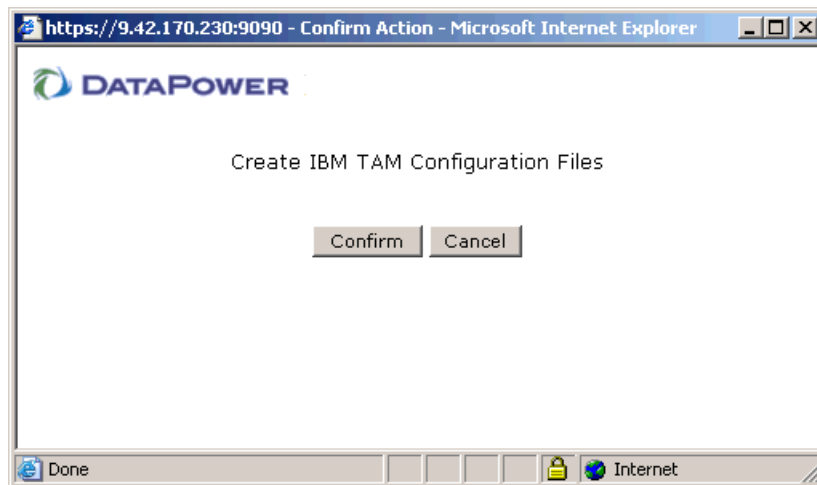


Figure 3-11 Confirming the creation of Tivoli Access Manager configuration files

4. In the Action completed successfully window (Figure 3-12), click **Close**.

Important: The message in the Action completed successfully window refers only to the fact that the configuration process *started* successfully. The process of creating the Tivoli Access Manager configuration files takes some time to complete.



Figure 3-12 Tivoli Access Manager configuration file creation started successfully

The DataPower WebGUI returns focus to the Configure IBM Tivoli Access Manager page (Figure 3-9 on page 40).

Configuring the DataPower Tivoli Access Manager client object

While the creation of the Tivoli Access Manager configuration is in progress, you can (partially) complete the configuration of the DataPower Tivoli Access Manager client object. Click **Apply** to save the client object, as shown in Figure 3-13.

Configure IBM Tivoli Access Manager

Main Authorization Server Replicas

IBM Tivoli Access Manager

Apply Cancel

Create IBM TAM Configuration Files

Name	ITSO_TAM *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
TAM Configuration File	local: (none) Upload... Fetch... *
SSL Key File	cert: (none) Upload... Fetch... *
SSL Key Stash File	cert: (none) Upload... Fetch... *
Return User Attributes	<input checked="" type="radio"/> on <input type="radio"/> off
Use LDAP over SSL	<input type="radio"/> on <input checked="" type="radio"/> off

Figure 3-13 Creating the DataPower Tivoli Access Manager client object

You now return to the AAA policy wizard Authenticate stage. The Tivoli Access Manager configuration that was just created is now displayed, as shown in Figure 3-14.

☐ Custom Template

Tivoli Access Manager Configuration | ISOTO_TAM + ... *

Define how to map credentials.

Method | none *

Back Next Advanced Cancel

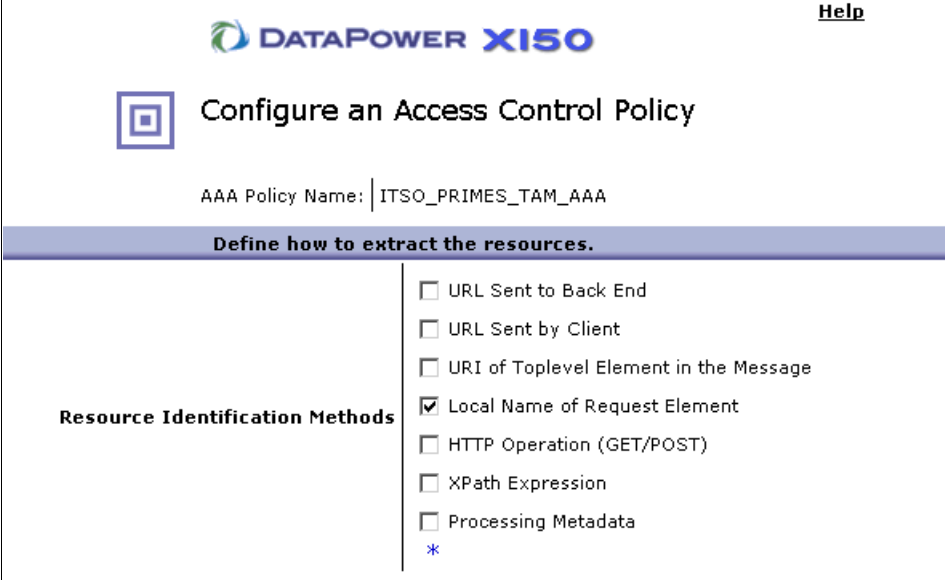
Figure 3-14 New Tivoli Access Manager configuration

3.2.4 Mapping the credentials

You do not need to perform any credential mapping in this scenario. On the Define mapping credentials page (Figure 3-14 on page 43), click **Next** to continue to the resource extraction page.

3.2.5 Extracting the resource

You must define the resource that the request is trying to access. Several mechanisms are available in this step. On the Resource Identification Methods page, select the **Local Name of Request Element** option.



The screenshot shows the 'Configure an Access Control Policy' interface. At the top, there's a 'Help' link. Below it, the title 'Configure an Access Control Policy' is displayed. The 'AAA Policy Name' is set to 'ITSO_PRIMES_TAM_AAA'. A section titled 'Define how to extract the resources.' contains a list of 'Resource Identification Methods'. The methods listed are: 'URL Sent to Back End', 'URL Sent by Client', 'URI of Toplevel Element in the Message', 'Local Name of Request Element' (which is checked), 'HTTP Operation (GET/POST)', 'XPath Expression', and 'Processing Metadata'. A blue asterisk is visible below the list.

Figure 3-15 Extracting resources by using the Local Name of Request Element

By selecting the Local Name of Request Element option, the `getPrime` field is extracted from the body of the SOAP message, as shown in Example 3-5.

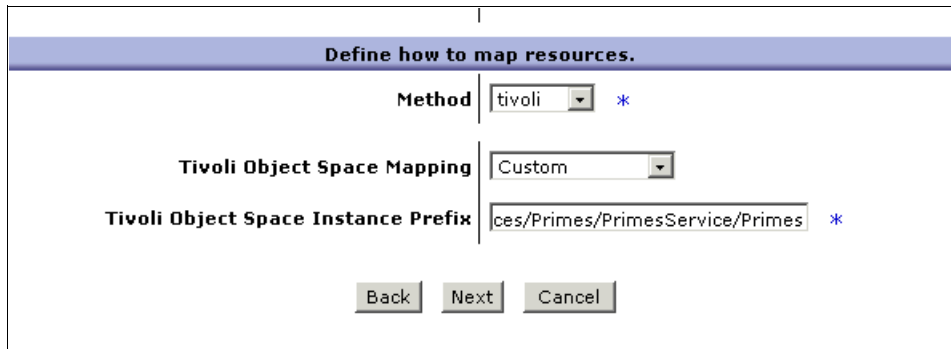
Example 3-5 Local name field of SOAP message

```
...
<soapenv:Body>
<q0:getPrime>
<numDigits>4</numDigits>
</q0:getPrime>
</soapenv:Body>
...
```


3.2.6 Mapping the resources

Next you map the extracted resource to a *resource string* that is recognized by, and corresponds to, the objectspace that was created for the Primes Web service. In the Define how to map resources section (Figure 3-16), complete the following steps:

1. For Method, select **tivoli**.
2. For Tivoli Object Space Mapping, select **Custom**.
3. In the Tivoli Object Space Instance Prefix field, enter
/itso/services/Primes/PrimesService/Primes.
4. Click **Next** to continue to the Authorize stage.



Define how to map resources.	
Method	tivoli *
Tivoli Object Space Mapping	Custom
Tivoli Object Space Instance Prefix	/itso/services/Primes/PrimesService/Primes *
<div>Back Next Cancel</div>	

Figure 3-16 Defining the resource mapping

3.2.7 Configuring the authorization

Now that you have identified and mapped the resource for the request, you can configure the authorization stage to use Tivoli Access Manager. In the Authorization configuration section (Figure 3-17), complete the following steps:

1. For Method, select **Contact Tivoli Access Manager**.
2. For Tivoli Access Manager Configuration, select the DataPower Tivoli Access Manager client object that was created in “Configuring the DataPower Tivoli Access Manager client object” on page 43. In this scenario, the Tivoli Access Manager client object is called **ITSO_TAM**.
3. For Default Action, select the **[WebService]i (TFIM Action)**, which corresponds with the Tivoli Access Manager configuration that is described in 3.1.5, “Applying the security policy” on page 34.
4. Click **Next** to proceed to the Post Processing stage.

The screenshot shows the 'Configure an Access Control Policy' window in the DataPower X150 management console. At the top, the 'AAA Policy Name' is set to 'ITSO_PRIMES_TAM_AAA'. Below this, a section titled 'Define how to authorize a request.' contains a 'Method' list with radio buttons. The 'Contact Tivoli Access Manager' option is selected. Below the method list, there are two configuration fields: 'Tivoli Access Manager Configuration' set to 'ITSO_TAM' and 'Default Action' set to '[WebService]i (TFIM Action)'. A 'Help' link is visible in the top right corner.

Figure 3-17 Configuring the authorization mechanism

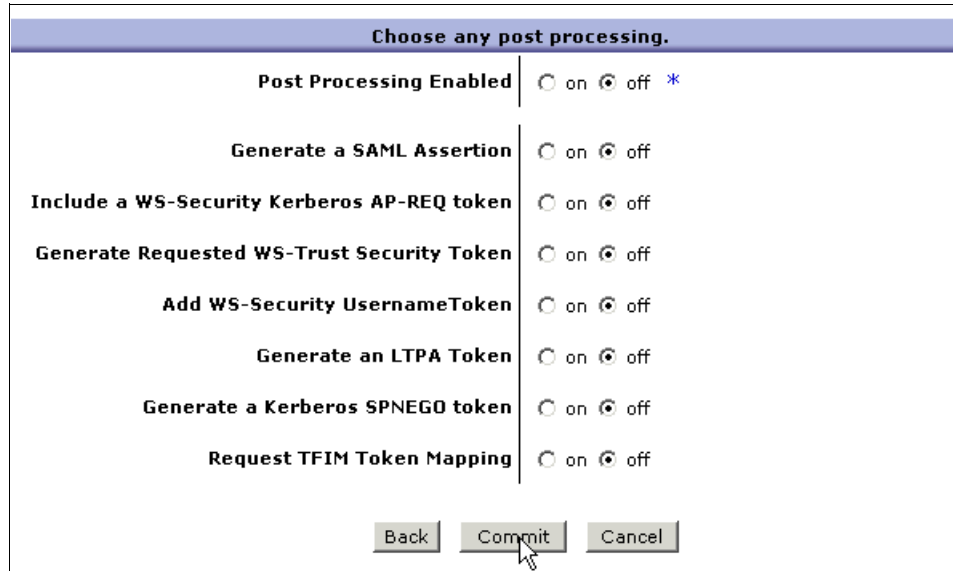
3.2.8 Performing the post processing

No post processing is required for this sample scenario.

3.2.9 Committing the AAA policy

You can now commit the changes to the AAA policy as explained in the following steps:

1. In the Post Processing section (Figure 3-18), click **Commit**.

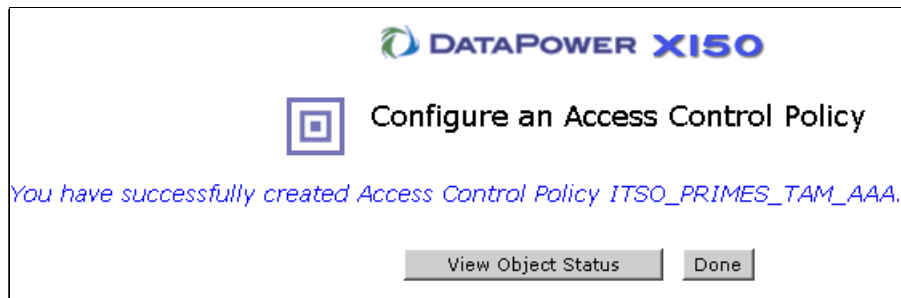


Choose any post processing.	
Post Processing Enabled	<input type="radio"/> on <input checked="" type="radio"/> off *
Generate a SAML Assertion	<input type="radio"/> on <input checked="" type="radio"/> off
Include a WS-Security Kerberos AP-REQ token	<input type="radio"/> on <input checked="" type="radio"/> off
Generate Requested WS-Trust Security Token	<input type="radio"/> on <input checked="" type="radio"/> off
Add WS-Security UsernameToken	<input type="radio"/> on <input checked="" type="radio"/> off
Generate an LTPA Token	<input type="radio"/> on <input checked="" type="radio"/> off
Generate a Kerberos SPNEGO token	<input type="radio"/> on <input checked="" type="radio"/> off
Request TFIM Token Mapping	<input type="radio"/> on <input checked="" type="radio"/> off


Back Commit Cancel

Figure 3-18 Saving the newly created AAA policy

2. In the confirmation window (Figure 3-19) that opens, you see a message that indicates successful AAA policy creation. Click **Done** to return to the AAA Action configuration page.



DATAPOWER X150

 **Configure an Access Control Policy**

You have successfully created Access Control Policy ITSO_PRIMES_TAM_AAA.

View Object Status Done

Figure 3-19 Successful AAA policy creation

3. On the Configure AAA Action page (Figure 3-20), click **Done**.

The screenshot shows the 'Configure AAA Action' page in the DataPower X150 WebGUI. The page is divided into sections: 'Input', 'Options', and 'Output'. The 'Input' section contains a text field with the value 'INPUT' and a dropdown menu also set to 'INPUT'. The 'Options' section features a checked checkbox labeled 'AAA'. Below this, the 'AAA Policy' section shows a dropdown menu with 'ITSO_PRIMES_TAM_AAA' selected, accompanied by '+' and '-' buttons. The 'Output' section contains a text field with 'OUTPUT' and a dropdown menu also set to 'OUTPUT'. At the bottom of the page are three buttons: 'Delete', 'Done', and 'Cancel'.

Figure 3-20 AAA Action with configured AAA policy

4. On the Policy page for the ITSO_PRIMES Web service proxy, click **Apply** to commit the changes to the Web service proxy.
5. Click **Save Config** to write the running configuration for the domain to file.

3.3 Updating the DataPower Tivoli Access Manager client object

In “Configuring the DataPower Tivoli Access Manager client object” on page 43, we create a new DataPower Tivoli Access Manager client object. This object is incomplete and not ready for use because we must create the Tivoli Access Manager configuration files and then associate the files with the object. In the following sections, you update the Tivoli Access Manager client object with the appropriate configuration files.

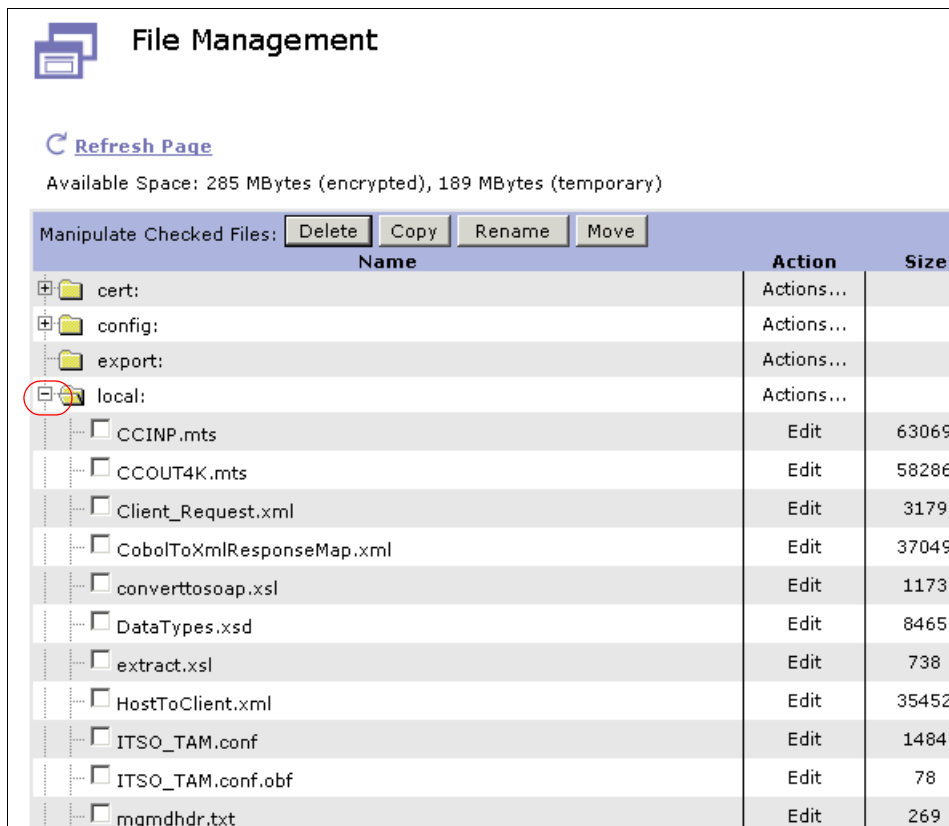
3.3.1 Verifying the file creation of the Tivoli Access Manager configuration

Before you attempt to update the Tivoli Access Manager client object, you must ensure that the Tivoli Access Manager files were generated successfully. Through the WebGUI, you can only verify that the plain-text configuration file or files were created, because the key and stash files are stored in the cert: directory, which is not accessible through the WebGUI.

To verify that the plain-text configuration files are present:

1. With the DataPower WebGUI, navigate to **Administration** → **File Management**.
2. On the File Management page (Figure 3-21), expand the **local:** folder.
3. Ensure that the files **ITSO_TAM.conf** and **ITSO_TAM.conf.obf** are present.

Missing files: If these files are not present, then it is likely that an error occurred during Tivoli Access Manager configuration. Check the DataPower system logs for more information.



File Management

[Refresh Page](#)

Available Space: 285 MBytes (encrypted), 189 MBytes (temporary)

Manipulate Checked Files:

Name	Action	Size
cert:	Actions...	
config:	Actions...	
export:	Actions...	
local:	Actions...	
CCINP.mts	Edit	63069
CCOUT4K.mts	Edit	58286
Client_Request.xml	Edit	3179
CobolToXmlResponseMap.xml	Edit	37049
converttosoap.xsl	Edit	1173
DataTypes.xsd	Edit	8465
extract.xsl	Edit	738
HostToClient.xml	Edit	35452
ITSO_TAM.conf	Edit	1484
ITSO_TAM.conf.obf	Edit	78
mqmdhdr.txt	Edit	269

Figure 3-21 Displaying files under the local folder

3.3.2 Updating the Tivoli Access Manager object

Now you can complete the configuration of the DataPower Tivoli Access Manager client object by performing the following actions:

- ▶ Updating the configuration, key, and stash file fields with the correct information
- ▶ Adding an authorization server replica

You update the configuration files first:

1. With the WebGUI, navigate to **Objects** → **Access** → **IBM Tivoli Access Manager**.
2. On the Configure IBM Tivoli Access Manager page (Figure 3-22), notice that the object state is *down*, because the object does not have any configuration files associated with it, as you see in the following sections. To proceed, click the object's name (circled in Figure 3-22).

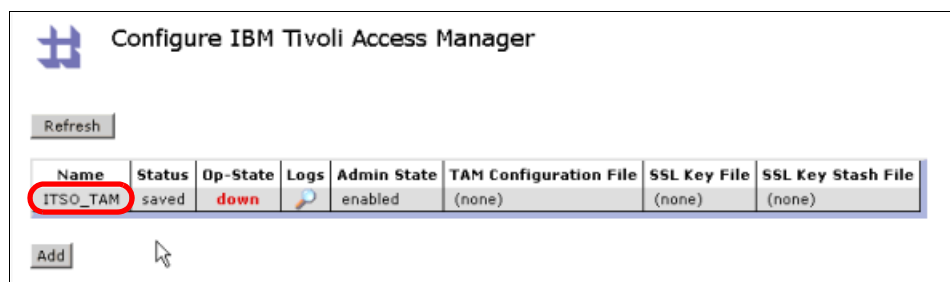


Figure 3-22 DataPower Tivoli Access Manager client objects

The ITSO_TAM object configuration page opens.

Updating the configuration files

On the ITSO_TAM object configuration page (Figure 3-23), we update the dependent file fields with the appropriate values. We enter the following values:

- ▶ For TAM Configuration File, enter `local:ITSO_TAM.conf`.
- ▶ For SSL Key File, enter `cert:ITSO_TAM.kdb`.
- ▶ For SSL Key Stash File, enter `cert:ITSO_TAM.sth`.

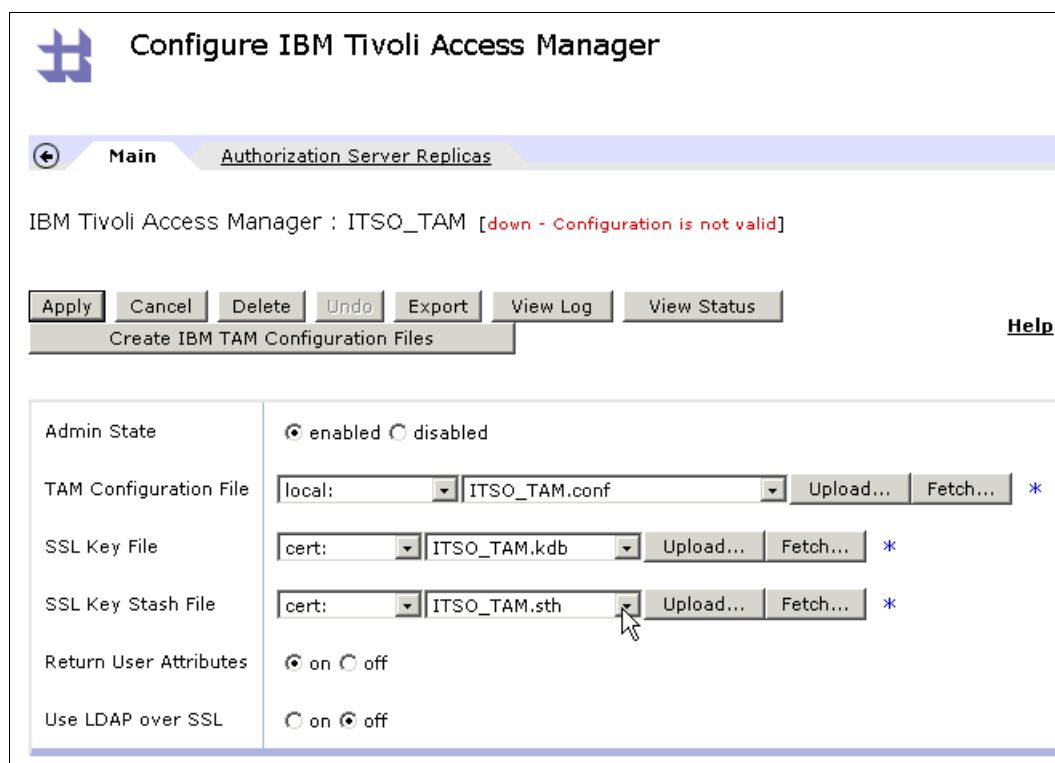


Figure 3-23 Completed configuration file for the ITSO_TAM object

Adding an authorization server replica

In the final configuration step, we add an authorization server replica. This replica is used by the DataPower Tivoli Access Manager client to determine where to send requests for authorization. It directs the client to a Tivoli Access Manager authorization server for the domain. Multiple authorization server replicas for the Tivoli Access Manager domain can be added here, each with different rankings. The client attempts to contact replicas with higher rankings before others replicas with lower rankings.

To configure this replica:

1. On the ITSO_TAM configuration page (Figure 3-24), click the **Authorization Server Replicas** tab.

Configure IBM Tivoli Access Manager

← Main **Authorization Server Replicas**

IBM Tivoli Access Manager : ITSO_TAM [down - Configuration is not valid]

Apply Cancel Delete Undo Export View Log View Status

Create IBM TAM Configuration Files [Help](#)

Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled		
TAM Configuration File	local: <input type="text"/> ITSO_TAM.conf	Upload...	Fetch... *
SSL Key File	cert: <input type="text"/> ITSO_TAM.kdb	Upload...	Fetch... *
SSL Key Stash File	cert: <input type="text"/> ITSO_TAM.sth	Upload...	Fetch... *
Return User Attributes	<input checked="" type="radio"/> on <input type="radio"/> off		
Use LDAP over SSL	<input type="radio"/> on <input checked="" type="radio"/> off		

Figure 3-24 Navigating to the Authorization Server Replica configuration

2. On the Authorization Server Replicas page (Figure 3-25), click **Add** to add an Authorization Server Replica object.

Configure IBM Tivoli Access Manager

← Main **Authorization Server Replicas**

IBM Tivoli Access Manager : ITSO_TAM [down - Configuration is not valid]

Apply Cancel Delete Undo Export View Log View Status

Create IBM TAM Configuration Files [Help](#)

Authorization Server Replica Host	Authorization Server Replica Port	Authorization Server Replica Weight
(no properties defined)		

Add

Figure 3-25 Adding an Authorization Server Replica object

3. On the Adding new Authorization Server Replicas property of IBM Tivoli Access Manager page (Figure 3-26), configure the authorization server replica. Enter the correct Tivoli Access Manager authorization server host name, port, and weight for your environment. Then click **Save**.

Adding new Authorization Server Replicas property of IBM Tivoli Access Manager [Help](#)

Authorization Server Replica Host	itso_tam_server *
Authorization Server Replica Port	7136 *
Authorization Server Replica Weight	10 *

Save Cancel

Figure 3-26 Entering the host name and port of Tivoli Access Manager authorization server

- On the Authorization Server Replicas tab of the Tivoli Access Manager object configuration page (Figure 3-27), the new replica is displayed in the list of hosts. Click **Apply** to save the changes for the ITSO_TAM DataPower Tivoli Access Manager client object.

Configure IBM Tivoli Access Manager

Main **Authorization Server Replicas**

IBM Tivoli Access Manager : ITSO_TAM [down - Configuration is not valid]

Apply Cancel Delete Undo Export View Log View Status Help

Create IBM TAM Configuration Files

Authorization Server Replica Host	Authorization Server Replica Port	Authorization Server Replica Weight	
itso_tam_server	7136	10	Delete Edit

Add

Figure 3-27 Authorization server replica added to configuration

- Ensure that the Tivoli Access Manager client starts without any error messages by checking the following items:
 - The state of the object is up. To verify this, look at the list of DataPower Tivoli Access Manager client objects. Navigate to **Objects** → **Access** → **IBM Tivoli Access Manager** and verify that the Object field of the ITSO_TAM client has a status of *up* (Figure 3-28).

Configure IBM Tivoli Access Manager

Refresh

Name	Status	Op-State	Logs	Admin State	TAM Configuration File
ITSO_TAM	saved	up		enabled	local:///ITSO_TAM.conf

Add

Figure 3-28 Verifying DataPower Tivoli Access Manager client object state

- The log file for the object indicates that the Tivoli Access Manager client thread started without an error. Under Logs, click the magnifying glass icon (Figure 3-28) to view the system log for the object. The log should look similar to the example in Figure 3-29.

msgid	message	Show last 50 100 all
0x00350014	tam (ITSO_TAM): Operational state up	
0x81a0001d	tam (ITSO_TAM): Tivoli Access Manager client successfully enabled	
0x8100000b	tam (ITSO_TAM): Event-Code (0x360001) - Pending tam (ITSO_TAM): starting tam thread	
0x81a00038	tam (ITSO_TAM): The Tivoli Access Manager object has been configured; attempting to start client authorization endpoint	
0x81a0003b	tam (ITSO_TAM): Successfully added replica to Tivoli Access Manager configuration - hostname 'itso_tam_server', port 7136, weight 10	
0x81a00036	tam (ITSO_TAM): 1 authorization replica configured	

Figure 3-29 System log for Tivoli Access Manager client object showing successful startup

3.4 Testing

In this section, we outline simple tests that you can and should perform to determine that the system is working correctly. Performing the tests also gives you insight into where to look for potential problems.

3.4.1 Testing the support material

This paper has additional materials that include the exported domain (AAA) that was built for this paper. They also include the WS-Security example messages that are used in the testing process. For more information about locating the additional material, see Appendix B, “Additional material” on page 101.

3.4.2 Testing the prerequisites

In this section, we explain the actions that you must take before you can begin testing.

HTTP command line utility cURL

The tests that we perform require the command line HTTP utility cURL. This tool is freely available for download from the cURL Web site at the following address:

<http://curl.haxx.se>

You can also find usage instructions and other information on this Web site.

DataPower logging

To obtain a full picture of the processes at work during testing, you must set the DataPower logging level to *debug*. Navigate to **Control Panel** → **Troubleshooting** → **Logging** → **Set Log Level**. On the Set Log Level page, set the log level to **debug** (Figure 3-30).

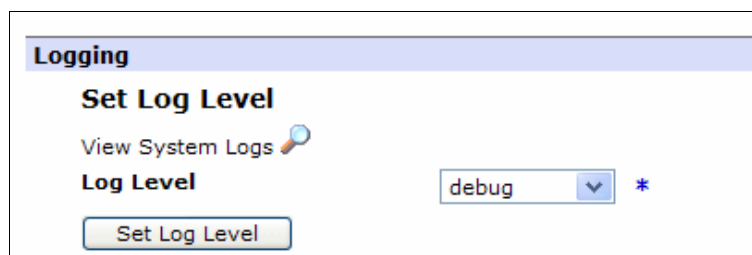


Figure 3-30 Enabling debug level logging

DataPower appliances also include a troubleshooting utility called *probe*. The probe displays detailed information about the processing of requests at every step in a processing rule. It is an invaluable testing tool. By using this tool, you can pinpoint where an error or unexpected behavior is occurring. To enable the probe for the Web service proxy:

1. Using the DataPower WebGUI, navigate to **Control Panel** → **Services** → **Web Service Proxy** → **ITSO_PRIMES**. Click **Show Probe** (Figure 3-31).

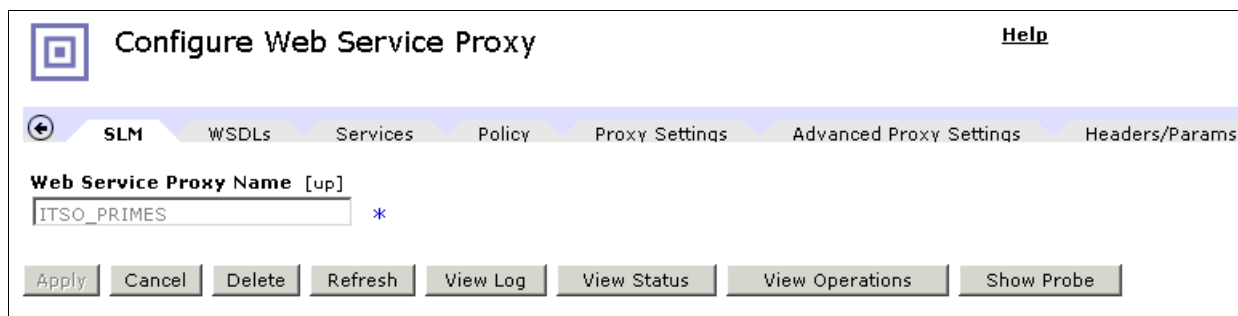


Figure 3-31 Displaying the probe for the Web service proxy ITSO_PRIMES

2. On the Transaction List page for ITSO_PRIMES (Figure 3-32), click the **Enable Probe** button.

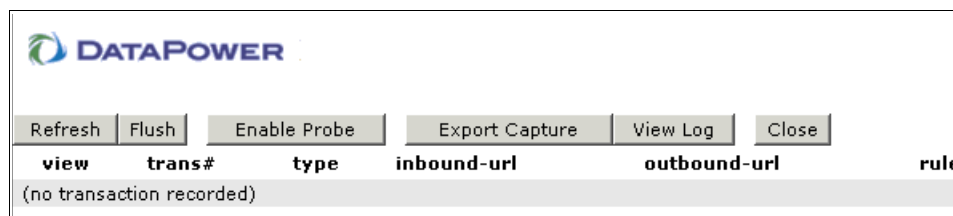


Figure 3-32 Enable Probe for the Web service proxy ITSO_PRIMES

3. In the confirmation window, ensure that the message states that debug mode was entered successfully. Click **Close**.

On the Transaction List page, notice that the Enable Probe button has changed to *Disable Probe*, as shown in Figure 3-33. You leave the Transaction List page open, because you will use it in the testing.

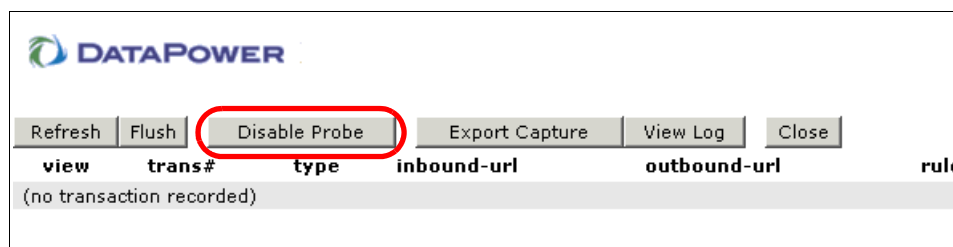


Figure 3-33 Probe enabled for Web service proxy ITSO_PRIMES

3.4.3 Testing requests to the DataPower appliance by using cURL

In this section, you send several requests to the DataPower appliance by using cURL and observe the behavior.

Requests for getPrime

Example 3-6 shows the SOAP body that requests a four-digit prime number from the Primes Web service operation *getPrime*.

Example 3-6 Request for a prime number from the getPrime operation

```
<soapenv:Body>
<q0:getPrime>
<numDigits>4</numDigits>
</q0:getPrime>
</soapenv:Body>
```

Requests for nextPrime

Example 3-7 shows the SOAP body that requests the next four-digit prime number from the Primes Web service operation *nextPrime*.

Example 3-7 Request for the next prime number from the nextPrime operation

```
<soapenv:Body>
<q0:nextPrime>
<start>4</start>
</q0:nextPrime>
</soapenv:Body>
```

Requests for random

Example 3-8 shows the SOAP body that requests a four-digit random number from the Primes Web service operation *random*.

Example 3-8 Request for a random number from the random operation

```
<soapenv:Body>
<q0:random>
<numDigits>4</numDigits>
</q0:random>
</soapenv:Body>
```

Primes division user tests

For the Primes division user *johndoe*, you must ensure that he is able to invoke the *getPrime* and *nextPrime* operations of the Primes Web service, but is unable to use the *random* operation of the Web service.

Example 3-9 shows the WS-Security element that the SOAP header of the messages has for these tests.

Example 3-9 WS-Security UsernameToken

```
<wsse:Security>
<wsse:UsernameToken>
<wsse:Username>johndoe</wsse:Username>
<wsse:Password>passw0rd</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
```

The additional material that accompanies this paper includes the following files that contain the requests to *getPrime*, *nextPrime*, and *random* with the user name *johndoe*:

- ▶ *primes-request-getPrime.xml*
- ▶ *primes-request-nextPrime.xml*
- ▶ *primes-request-random.xml*

For information about how to download the additional materials, see Appendix B, “Additional material” on page 101.

getPrime

To test access to the *getPrime* operation, enter the following command by using cURL:

```
curl -k -H "SOAPAction: getPrime" --data-binary @primes-request-getPrime.xml
https://<dp_host>:<ws-proxy_port>/PrimesWebService/services/Primes
```

Example 3-10 shows the SOAP response to the request.

Example 3-10 SOAP response to curl command to getPrime operation for johndoe

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header/>
  <soapenv:Body>
```

```

    <p234:getPrimeResponse
      xmlns:p234="http://com.itso">
      <getPrimeReturn>2647</getPrimeReturn>
    </p234:getPrimeResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

Figure 3-34 shows an example of the information that the log for the ITSO_PRIMES Web service proxy should display.

info	1400146	>	9.42.170.199	0x80c00002	wsgw (ITSO_PRIMES): rule (ITSO_PRIMES_wsdI_PrimeService_wsdI_request-rule_ITSO_PRIMES_match_all): #1 aaa: 'INPUT ITSO_PRIMES_TAM_AAA stored in tempvar1' completed ok.
info	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Message allowed
info	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authorization succeeded with credential 'cn=johndoe,dc=itso,dc=ibm,dc=com' for resource '/itso/services/Primes/PrimesService/Primes/getPrime'
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AZ entry
debug	1400146	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authorization and attribute retrieval succeeded
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): The operation for the Tivoli authorization request is [WebService]i
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authorizing with "tivoli"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AZ cache check with key="authz<?xml version="1.0" encoding="UTF-8"?> <container><mapped-credentials type="none" au-success="true"><entry type="tivoli">cn=johndoe,dc=itso,dc=ibm,dc=com</entry></mapped-credentials><mapped-resource type="tivoli"><entry type="tivoli">/itso/services/Primes/PrimesService/Primes/getPrime</entry></mapped-resource><identity><entry type="wssec-username"><username>johndoe</username><password type="" sanitize="true">passw0rd</password></entry></identity><au-ancillary-info/><az-ancillary-info><default-action>_WebService_i</default-action><dynamic-action/><server>ITSO_TAM</server></az-ancillary-info></container>authztivoli"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Finished parsing store:///dp/aaapolicy.xml
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Parsing document 'store:///dp/aaapolicy.xml'
info	1400146	>	9.42.170.199	0x80000001	wsgw (ITSO_PRIMES): The Tivoli Access Manager MapResource prefix is itso/services/Primes/PrimesService/Primes
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping resources using tivoli
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping credentials using none
info	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authentication succeeded with (wssec-username, username='johndoe' password='*****')
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AU entry
debug	1400146	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authentication succeeded
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authenticating with "tivoli"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AU cache check with key="policyname<?xml version="1.0" encoding="UTF-8"?> <identity><entry type="wssec-username"><username>johndoe</username><password type="" sanitize="true">passw0rd</password></entry></identity><?xml version="1.0" encoding="UTF-8"?> <au-ancillary-info/>tivoli"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authentication Caching is on: true
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Resource "getPrime"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting resources using "request-opname"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting identity using "wssec-username"
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Stylesheet URL to compile is 'store:///dp/aaapolicy.xml'
debug	1400146	>	9.42.170.199		wsgw (ITSO_PRIMES): Finished parsing http://127.0.0.1:63504/

Figure 3-34 Log output for johndoe request to getPrime

You can see that authentication and authorization succeeded for the getPrime request, as highlighted by the boxes in Figure 3-34. In particular, notice that the mapped resource string

in the upper box is `/itso/sevices/Primes/PrimesService/Primes/getPrime`. That is to say that the AAA policy has mapped the extracted resource `getPrime` to the required resource string for Tivoli Access Manager. Authorization has succeeded because the user `johndoe` belongs to the `itso-primes` group. The `itso-primes` group has the correct access permissions to the object `/itso/sevices/Primes/PrimesService/Primes/getPrime`. Refer to 3.1.5, “Applying the security policy” on page 34, for details about the security policy that is applied in Tivoli Access Manager.

In addition, on the Transaction List page (Figure 3-35), click the **Refresh** button to display the transaction request.

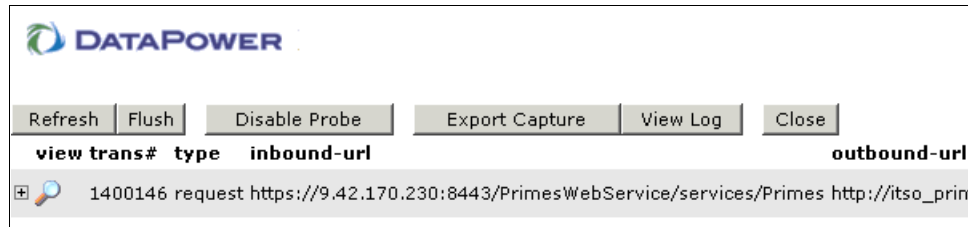


Figure 3-35 Transaction list for the `getPrime` request

You can click the magnifying glass icon to display the probe interface for the transaction, as shown in Figure 3-36.

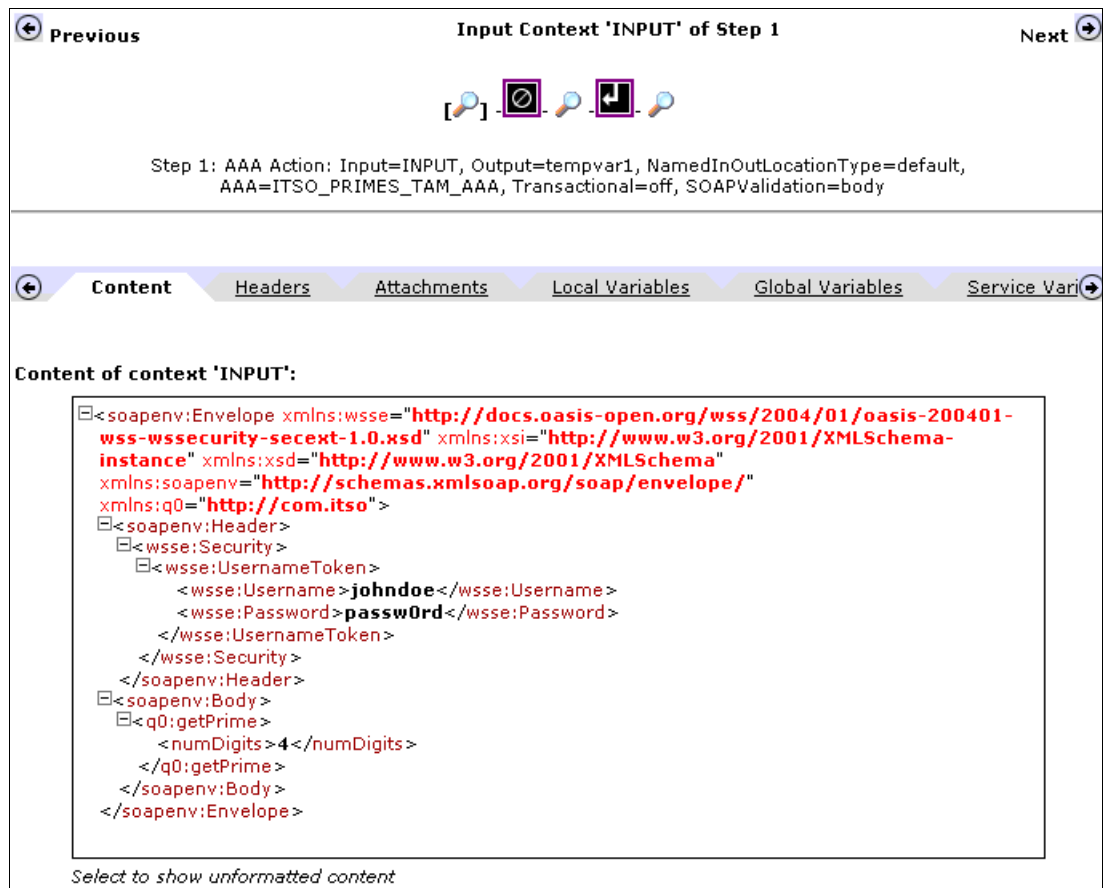


Figure 3-36 Probe capture for the `getPrime` request

nextPrime

To test access to the nextPrime operation, enter the following command by using cURL:

```
curl -k -H "SOAPAction: nextPrime" --data-binary @primes-request-nextPrime.xml  
https://<dp_host>:<ws-proxy_port>/PrimesWebService/services/Primes
```

Example 3-11 shows the SOAP response to the request.

Example 3-11 SOAP response to curl command to nextPrime operation for johndoe

```
<?xml version="1.0" encoding="UTF-8"?>  
<soapenv:Envelope  
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"  
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  
  <soapenv:Header/>  
  <soapenv:Body>  
    <p234:nextPrimeResponse  
      xmlns:p234="http://com.itso">  
      <nextPrimeReturn>3</nextPrimeReturn>  
    </p234:nextPrimeResponse>  
  </soapenv:Body>  
</soapenv:Envelope>
```

Figure 3-37 shows the information that the log for the ITSO_PRIMES Web service proxy should display.

info	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Message allowed
info	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authorization succeeded with credential 'cn=johndoe,dc=itso,dc=ibm,dc=com' for resource '/itso/services/Primes/PrimesService/Primes/nextPrime'
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AZ entry
debug	1401620	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authorization and attribute retrieval succeeded
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): The operation for the Tivoli authorization request is [WebService]i
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authorizing with "tivoli"
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AZ cache check with key="authz<?xml version="1.0" encoding="UTF-8"?> <container><mapped-credentials type="none" au-success="true"><entry type="tivoli">cn=johndoe,dc=itso,dc=ibm,dc=com</entry></mapped-credentials><mapped-resource type="tivoli"><entry type="tivoli">/itso/services/Primes/PrimesService/Primes/nextPrime</entry></mapped-resource><identity><entry type="wssec-username"><username>johndoe</username><password type="" sanitize="true">password</password></entry></identity><au-ancillary-info/><az-ancillary-info><default-action>_WebService_i</default-action><dynamic-action/><server>ITSO_TAM</server></az-ancillary-info></container>authztivoli"
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Finished parsing store:///dp/aaapolicy.xml
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Parsing document 'store:///dp/aaapolicy.xml'
info	1401620	>	9.42.170.199	0x80000001	wsgw (ITSO_PRIMES): The Tivoli Access Manager MapResource prefix is itso/services/Primes/PrimesService/Primes
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping resources using tivoli
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping credentials using none
info	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authentication succeeded with (wssec-username, username='johndoe' password='*****')
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AU entry
debug	1401620	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authentication succeeded
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authenticating with "tivoli"
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AU cache check with key="policyname<?xml version="1.0" encoding="UTF-8"?> <identity><entry type="wssec-username"><username>johndoe</username><password type="" sanitize="true">password</password></entry></identity><?xml version="1.0" encoding="UTF-8"?> <au-ancillary-info/>tivoli"
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authentication Caching is on: true
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Resource "nextPrime"
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting resources using "request-opname"
debug	1401620	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting identity using "wssec-username"

Figure 3-37 Log output for johndoe request to nextPrime

random

To test access to the random operation, enter the following command by using cURL:

```
curl -k -H "SOAPAction: random" --data-binary @primes-request-random.xml  
https://<dp_host>:<ws-proxy_port>/PrimesWebService/services/Primes
```

Example 3-12 shows the SOAP response to the request.

Example 3-12 SOAP response to the curl command to random operation for johndoe

```
<?xml version="1.0" encoding="UTF-8"?>  
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">  
  <env:Body>  
    <env:Fault>  
      <faultcode>env:Client</faultcode>  
      <faultstring>Rejected by policy. (from client)</faultstring>  
    </env:Fault></env:Body>  
</env:Envelope>
```

Figure 3-38 shows the information that the log for the ITSO_PRIMES Web service proxy should display.

p	error	1403788	>	9.42.170.199	0x80c00009	wsgw (ITSO_PRIMES): request ITSO_PRIMES_wsdI_PrimeService_wsdI_request-rule_ITSO_PRIMES_match_all #1 aaa: 'INPUT ITSO_PRIMES_TAM_AAA stored in tempvar1' failed: Rejected by policy.
	error	1403788	>	9.42.170.199	0x80c00010	wsgw (ITSO_PRIMES): Execution of 'store:///dp/aaapolicy.xml' aborted: Rejected by policy.
	warn	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Message rejected
	warn	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Reject set: Rejected by policy.
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Reject called (with override): Rejected by policy.
	error	1403788	>	9.42.170.199	0x01d30002	wsgw (ITSO_PRIMES): AAA Authorization Failure
	warn	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authorization failed with credential 'cn=johndoe,dc=itso,dc=ibm,dc=com' for resource '/itso/services/Primes/PrimesService/Primes/random'
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AZ entry
	info	1403788	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message:
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): The operation for the Tivoli authorization request is [WebService]i
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authorizing with "tivoli"
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AZ cache check with key="authz<?xml version="1.0" encoding="UTF-8"?> <container><mapped-credentials type="none" au-success="true"><entry type="tivoli">cn=johndoe,dc=itso,dc=ibm,dc=com</entry></mapped-credentials><mapped-resource type="tivoli"><entry type="tivoli">/itso/services/Primes/PrimesService/Primes/random</entry></mapped-resource><identity><entry type="wssec-username"><username>johndoe</username><password type="" sanitize="true">passwd0rd</password></entry></identity><au-ancillary-info><az-ancillary-info><default-action>_WebService_i</default-action><dynamic-action/><server>ITSO_TAM</server></az-ancillary-info></container>authztivoli"
e	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Finished parsing store:///dp/aaapolicy.xml
e	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Parsing document 'store:///dp/aaapolicy.xml'
	info	1403788	>	9.42.170.199	0x80000001	wsgw (ITSO_PRIMES): The Tivoli Access Manager MapResource prefix is itso/services/Primes/PrimesService/Primes
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping resources using tivoli
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping credentials using none
	info	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authentication succeeded with (wssec-username, username='johndoe' password='*****')
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AU entry
	debug	1403788	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authentication succeeded
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authenticating with "tivoli"
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AU cache check with key="polycname<?xml version="1.0" encoding="UTF-8"?> <identity><entry type="wssec-username"><username>johndoe</username><password type="" sanitize="true">passwd0rd</password></entry></identity><?xml version="1.0" encoding="UTF-8"?> <au-ancillary-info>tivoli"
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authentication Caching is on: true
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Resource "random"
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting resources using "request-opname"
	debug	1403788	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting identity using "wssec-username"

Figure 3-38 Log output for johndoe request to random

From the log shown in Figure 3-38, you can see that authentication succeeded for the user *johndoe* as indicated by the lower box. However, authorization failed for the request to the random operation, as highlighted by the upper box in Figure 3-38. In particular, notice that the mapped resource string in the upper box is `/itso/sevices/Primes/PrimesService/Primes/random`. That is to say that the AAA policy has mapped the extracted resource *random* to the required resource string for Tivoli Access Manager. Based on the security policy applied in the Tivoli Access Manager environment, the request was rejected.

Random division user tests

For the Random division user *janedoe*, you must ensure that she is able to invoke the random operation of the Primes Web service, but is unable to use the `getPrime` and `nextPrime` operations of the Web service.

For these tests, the SOAP header of the messages has the WS-Security element shown in Example 3-13.

Example 3-13 WS-Security UsernameToken

```
<wsse:Security>
<wsse:UsernameToken>
<wsse:Username>janedoe</wsse:Username>
<wsse:Password>passw1rd</wsse:Password>
</wsse:UsernameToken>
</wsse:Security>
```

The additional material that accompanies this paper includes the following files that contain the requests to `getPrime`, `nextPrime` and `random` with the user name *janedoe*:

- ▶ random-request-getPrime.xml
- ▶ random-request-nextPrime.xml
- ▶ random-request-random.xml

For information about how to download the additional materials, see Appendix B, “Additional material” on page 101.

random

To test access to the random operation, enter the following command by using cURL:

```
curl -k -H "SOAPAction: random" --data-binary @random-request-random.xml
https://<dp_host>:<ws-proxy_port>/PrimesWebService/services/Primes
```

Example 3-14 shows the SOAP response to the request.

Example 3-14 SOAP response to the curl command to the random operation for janedoe

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header/>
  <soapenv:Body>
    <p234:randomResponse
      xmlns:p234="http://com.itso">
      <randomReturn>9321</randomReturn>
    </p234:randomResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

Figure 3-39 shows the information that the log for the ITSO_PRIMES Web service proxy should display.

	info	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Message allowed
	info	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authorization succeeded with credential 'cn=janedoe,dc=itso,dc=ibm,dc=com' for resource '/itso/services/Primes/PrimesService/Primes/random'
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AZ entry
	debug	1405712	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authorization and attribute retrieval succeeded
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): The operation for the Tivoli authorization request is [WebService]i
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authorizing with "tivoli"
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AZ cache check with key="authz<?xml version="1.0" encoding="UTF-8"?> <container><mapped-credentials type="none" au-success="true"><entry type="tivoli">cn=janedoe,dc=itso,dc=ibm,dc=com</entry></mapped-credentials><mapped-resource type="tivoli"><entry type="tivoli">/itso/services/Primes/PrimesService/Primes/random</entry></mapped-resource><identity><entry type="wssec-username"><username>janedoe</username><password type="" sanitize="true">passw1rd</password></entry></identity><au-ancillary-info/><az-ancillary-info><default-action>_WebService_i</default-action><dynamic-action/><server>ITSO_TAM</server></az-ancillary-info></container>authztivoli"
se	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Finished parsing store:///dp/aaapolicy.xml
se	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Parsing document 'store:///dp/aaapolicy.xml'
g	info	1405712	>	9.42.170.199	0x80000001	wsgw (ITSO_PRIMES): The Tivoli Access Manager MapResource prefix is itso/services/Primes/PrimesService/Primes
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping resources using tivoli
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Mapping credentials using none
	info	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): tivoli authentication succeeded with (wssec-username, username='janedoe' password='*****')
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Cached AU entry
	debug	1405712	>	9.42.170.199	0x81a00006	wsgw (ITSO_PRIMES): Tivoli Access Manager authorization client log message: TAM authentication succeeded
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authenticating with "tivoli"
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): AU cache check with key="policyname<?xml version="1.0" encoding="UTF-8"?> <identity><entry type="wssec-username"><username>janedoe</username><password type="" sanitize="true">passw1rd</password></entry></identity><?xml version="1.0" encoding="UTF-8"?> <au-ancillary-info/>tivoli"
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Authentication Caching is on: true
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Resource "random"
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting resources using "request-opname"
	debug	1405712	>	9.42.170.199		wsgw (ITSO_PRIMES): Policy(ITSO_PRIMES_TAM_AAA): Extracting identity using "wssec-username"

Figure 3-39 Log output for the janedoe request to random

You can see that authentication and authorization succeeded for the random request, as highlighted by the boxes in Figure 3-39. In particular, notice that the mapped resource string in the upper box is /itso/services/Primes/PrimesService/Primes/random. That is to say that the AAA policy mapped the extracted resource *random* to the required resource string for Tivoli Access Manager. Authorization succeeded because the user *janedoe* belongs to the *itso-random* group. The *itso-random* group has the correct access permissions to the object /itso/sevices/Primes/PrimesService/Primes/random.

Refer to 3.1.5, “Applying the security policy” on page 34, for the details about the security policy that was applied in Tivoli Access Manager.

3.5 Summary

Implementing a AAA policy on a DataPower service object, such as a multiprotocol gateway or a Web service proxy, provides a real enforcement point. A AAA policy supports various identity extraction methods.

In addition, the following authentication and authorization authorities are supported among others:

- ▶ IBM Tivoli Directory Server
- ▶ IBM Tivoli Access Manager for e-business
- ▶ AAA info file or basic XML file and so on

In this chapter, we explained how to create a DataPower Tivoli Access Manager configuration and use it in a AAA policy. The AAA policy enforces an access control policy that is defined for a Web service by using Tivoli Access Manager as the decision point.



AAA with the LDAP directory

In this chapter, we describe the methods that are used to perform authentication and authorization operations by using an Lightweight Directory Access Protocol (LDAP)-enabled directory server. Building on the concepts that we define in the previous chapters, in this chapter, we explain how to build additional processing rules at the operation level of a Web service proxy. In addition, we demonstrate an alternative method of creating authentication, authorization, and audit (AAA) policies.

4.1 IBM Tivoli Directory Server

Tivoli Directory Server is the primary LDAP-accessible directory server from IBM. Tivoli Directory Server is a robust and scalable enterprise identity management product.

In this section, we discuss the Tivoli Directory Server infrastructure of the management zone that is illustrated in Figure 4-1.

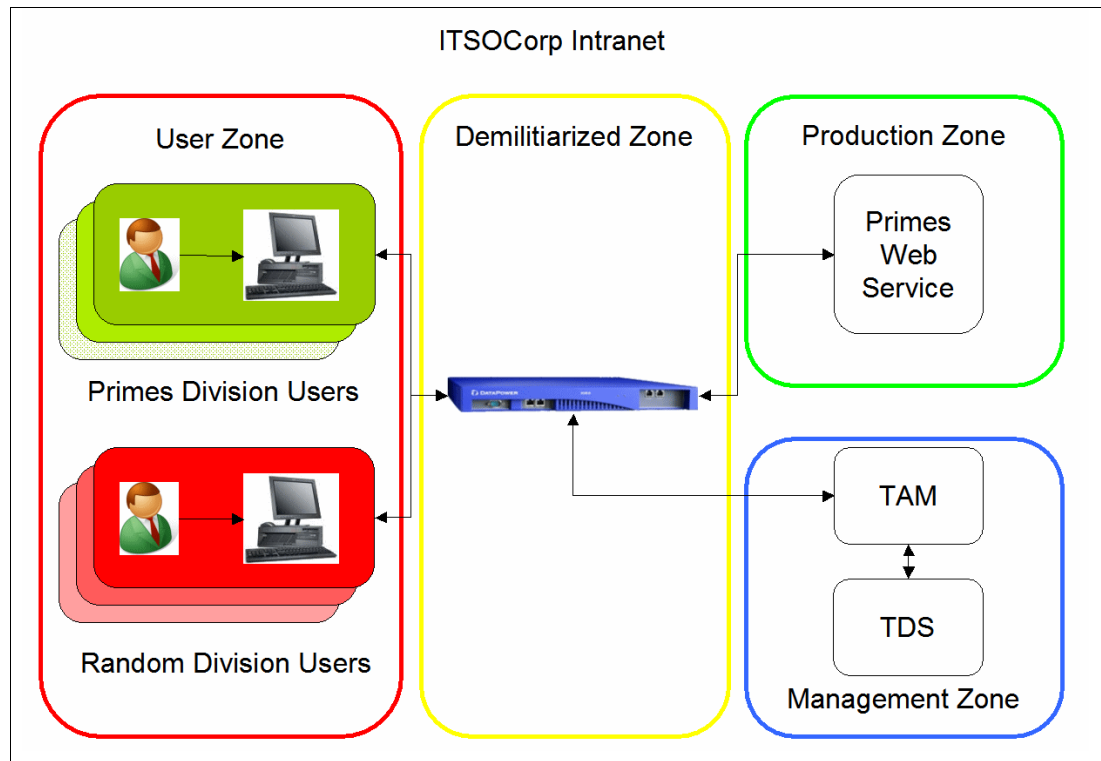


Figure 4-1 Deployment architecture of Tivoli Directory Server

4.1.1 Tivoli Directory Server and DataPower

A DataPower AAA policy can use Tivoli Directory Server (or any LDAP-accessible directory server) for authentication and authorization. The AAA policy retrieves the user's password from the directory server and compares it with the password that is extracted from the request. Conversely, for authorization, the DataPower appliance checks for the user's membership of a specified group in the directory.

More information: In the following sections, we assume that you have configured a functional Tivoli Directory Server. For more information about installing and configuring Tivoli Directory Server, refer to the Tivoli Directory Server Information Center at the following address:

<http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?toc=/com.ibm.IBMDS.doc/toc.xml>

4.1.2 Configuring the Tivoli Directory Server

The Tivoli Directory Server requires various entries for both authentication and authorization management. As discussed in 1.2.2, “Securing the Web service” on page 3, ITSOCorp wants to restrict access to the operations of the Primes Web service to certain employees. In the following section, we discuss the steps to configure Tivoli Directory Server to support this policy.

Users and groups: If you configured Tivoli Access Manager as described in 3.1.5, “Applying the security policy” on page 34, and are using the Tivoli Directory Server that is used by Tivoli Access Manager for this section, then the users and groups already exist.

Configuring the users and groups

You can create the users and groups by using the Lightweight Directory Interchange Format (LDIF) commands. To begin, we create the users that are defined in LDIF notation shown in Example 4-1.

Example 4-1 ITSOCorp users in LDIF representation

```
dn: cn=johndoe,dc=itso,dc=ibm,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: ePerson
userPassword: passw0rd
sn: John Doe
cn: johndoe
```

```
dn: cn=janedoe, dc=itso,dc=ibm,dc=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: ePerson
userPassword: passw1rd
sn: Jane Doe
cn: janedoe
```

The ITSOCorp divisions Primes and Random are represented in LDIF notation in Example 4-2.

Example 4-2 ITSOCorp divisions in LDIF representation

```
dn: cn=itso-primes,dc=itso,dc=ibm,dc=com
description: Primes division users
objectclass: top
objectclass: groupOfNames
member: cn=johndoe,dc=itso,dc=ibm,dc=com
cn: itso-primes
```

```
dn: cn=itso-random, dc=itso,dc=ibm,dc=com
description: Random division users
objectclass: top
objectclass: groupOfNames
member: cn=janedoe,dc=itso,dc=ibm,dc=com
cn: itso-random
```

In addition to the LDIF definitions, you can create the users and groups by using the directory modification tools that ship with Tivoli Directory Server. You can find the LDIF definitions in the additional material that accompanies this paper in the itsocorp-tds.ldif file. For information about how to download the additional materials, see Appendix B, “Additional material” on page 101.

Example 4-3 shows the commands to apply the content of the LDIF to Tivoli Directory Server. In the command, we assume that the suffix **dc=itso,dc=ibm,dc=com** exists.

Example 4-3 Creating users and groups in Tivoli Directory Server

```
ldapadd -h <tds_server> -p <port> -D <bind_dn> -w <bind_password> -f
itsocorp-tds.ldif
```

Figure 4-2 shows the configuration of the user and group base Distinguished Name (DN), dc=itso,dc=ibm,dc=com.

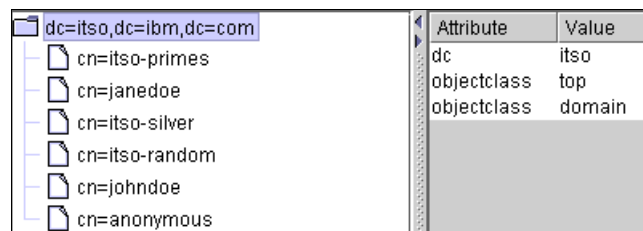


Figure 4-2 Tivoli Directory Server server entries

4.2 Creating a Web service proxy for Tivoli Directory Server

In this chapter, we use a slightly different methodology to authorize requests. You use the multiple layer policy feature of the Web service proxy to enforce access to the Primes Web service. With this feature, processing rules can be configured at the service, port, and operation levels of a Web service. If the object policy of a Web service proxy has multiple processing rules, the processing rule that can be selected by the rule’s match action. The processing rule that is hierarchically closest to the operation being requested is the rule that is executed.

With this in mind, authorization checks by using Tivoli Directory Server are performed with individual AAA policies. The AAA policies are specified at the operation level or levels of the Web service proxy. This means that a separate AAA policy exists for each of the getPrime, nextPrime, and random operations of the Primes Web service. In Chapter 3, “AAA with Tivoli Access Manager” on page 29, recall the single DataPower AAA policy that was configured at the Web Service Description Language (WSDL) level of the Web service proxy. It used Tivoli Access Manager (access control list (ACL) policies) to control access to the Primes Web service rather than individual DataPower AAA policies.

To create the Web service proxy in this chapter, repeat the steps used in “Configuring a Web service proxy” on page 11, with the following information:

- ▶ For Name the Web Service Proxy, type ITSO_PRIMES_TDS.
- ▶ Create a new HTTPS listener that listens on port 8444.

4.3 Configuring the Web service proxy policy

In order to define the access control policy using Tivoli Directory Server, the ITSO_PRIMES_TDS Web service proxy must have processing rules created for the getPrime, nextPrime, and random operations. We explain how to add these processing rules and add the AAA policies for the operations.

4.3.1 Creating the processing rules

To create the operation level processing rules for the ITSO_PRIMES_TDS Web service proxy:

1. Select **Control Panel** → **Services** → **Web Service Proxy** → **ITSO_PRIMES_TDS** → **Policy** and click the **ITSO_PRIMES_TDS Web Service Proxy Policy** tab.
2. On the Web Service Proxy Policy page (Figure 4-3), click the **Show Operations** button. Then for the getPrime operation, click the **Add Rule** button.

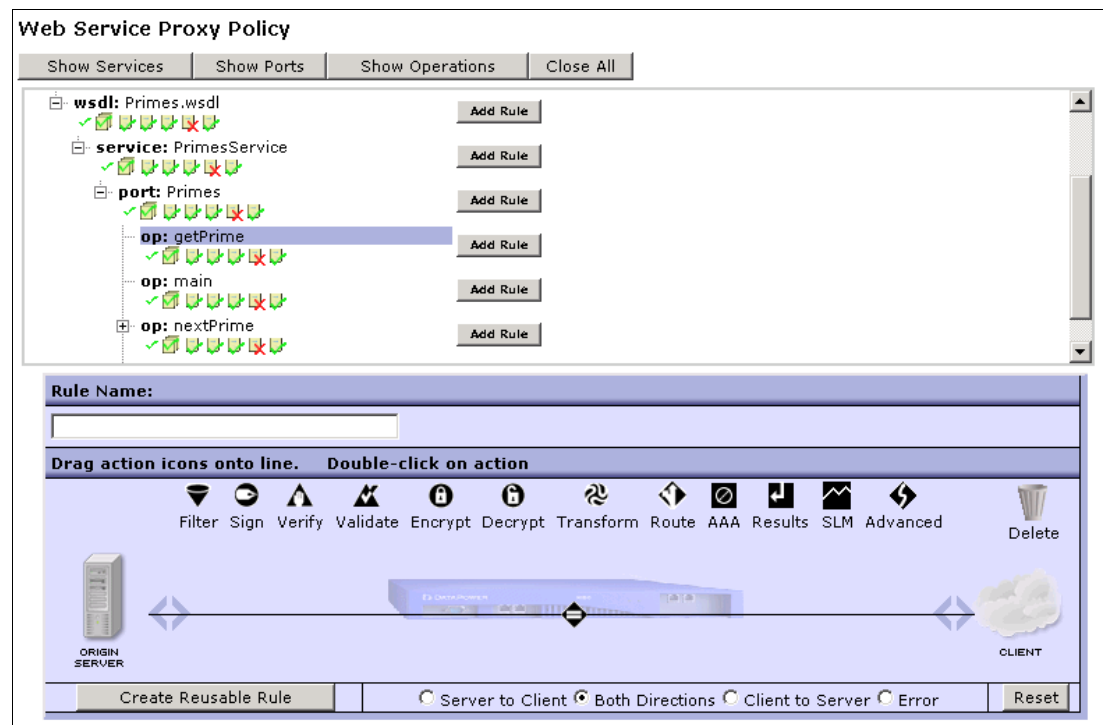


Figure 4-3 Adding a rule

3. In the shaded area of the Web Service Proxy Policy page, drag the **AAA** action onto the processing rule line and select the **Client to Server** radio button (Figure 4-4). Double-click the **AAA** action.

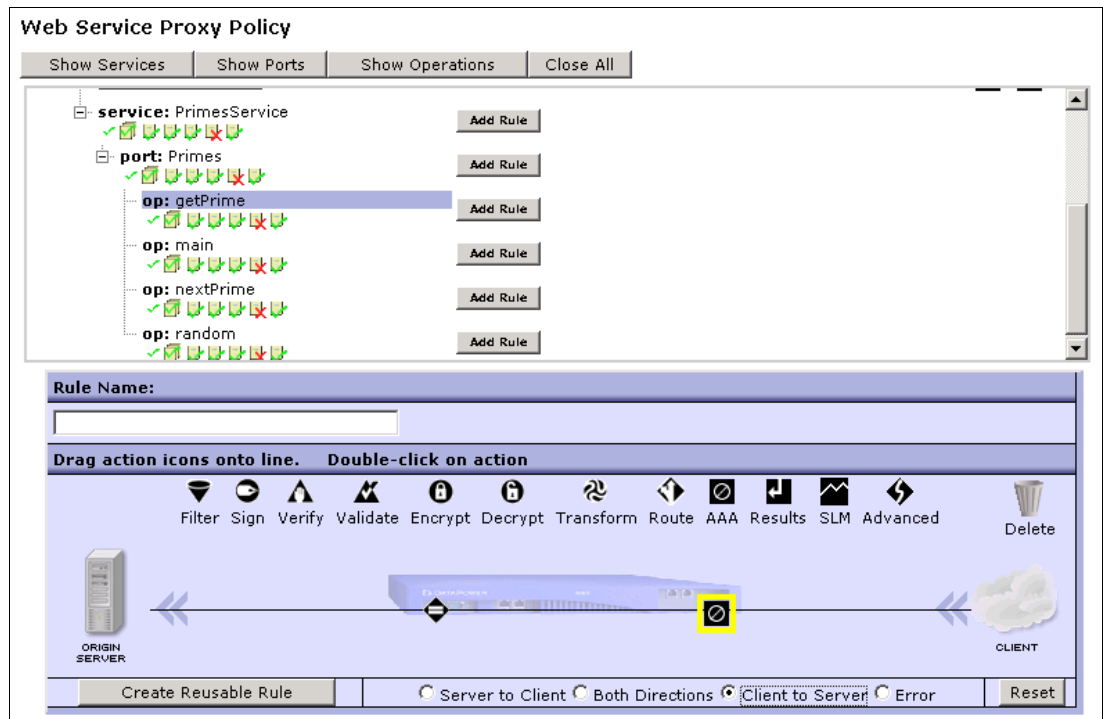


Figure 4-4 Dragging the AAA action to the processing rule line and selecting the Client to Server button

4. In the Configure AAA Action window (Figure 4-5), verify the settings and click **Done**.

The screenshot shows a web browser window titled "https://9.42.170.230:9090 - DataPower XI50 - Configure AAA Action - Microsoft Internet ...". The main content area has the "DATAPOWER XI50" logo and the title "Configure AAA Action". Below the title is a tabbed interface with "Basic" and "Advanced" tabs. The "Basic" tab is active and contains the following sections:

- Input**: A label "Input" followed by a text box containing "(auto)" and a dropdown menu also showing "(auto)".
- Options**: A section with a checked radio button labeled "AAA".
- AAA Policy**: A label "AAA Policy" followed by a dropdown menu showing "(none)", a "+" button, and a "... button".
- Output**: A label "Output" followed by a text box containing "(auto)" and a dropdown menu also showing "(auto)".

At the bottom of the form are three buttons: "Delete", "Done", and "Cancel". The browser's status bar at the bottom shows "Done" and "Internet".

Figure 4-5 Configuring the AAA action

- Back on the Web Service Proxy Policy page (Figure 4-6), drag a **Results action** onto the processing rule line. Double-click the **Results action**.

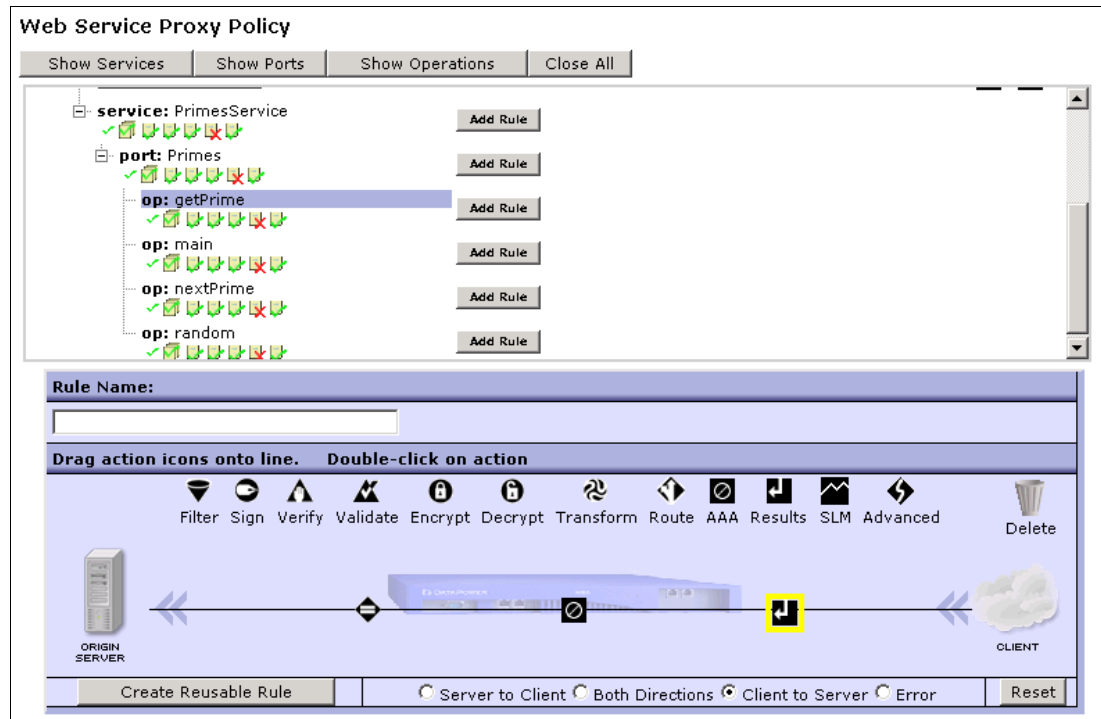


Figure 4-6 Dragging the Results action to the processing rule line

- On the Configure Results Action page, verify the configuration as shown in Figure 4-7 and click **Done**.

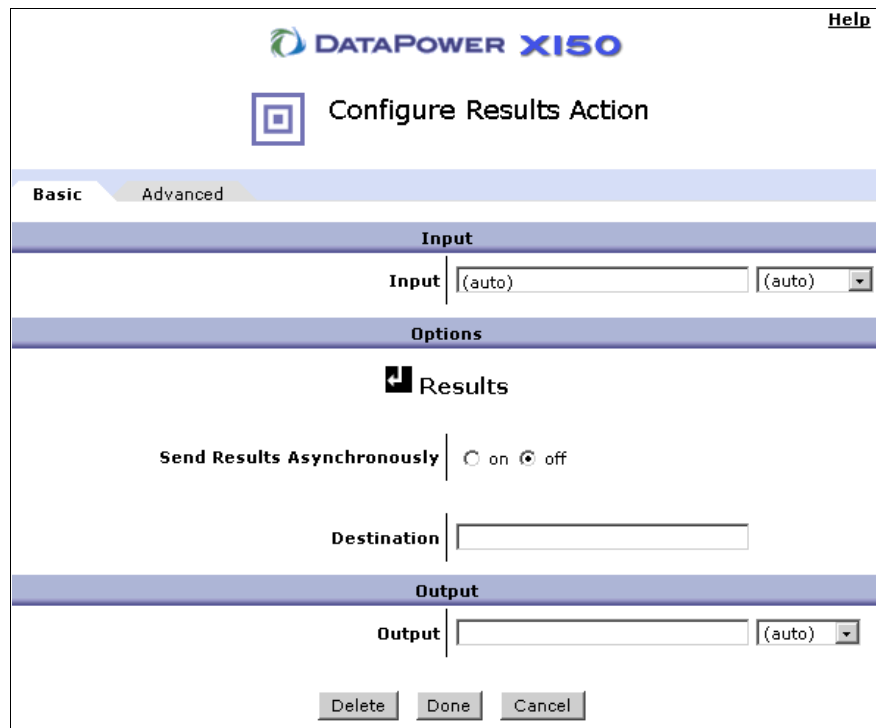


Figure 4-7 Configuring the Results action

Figure 4-8 shows an example of how the processing rule for the getPrime operation should look.

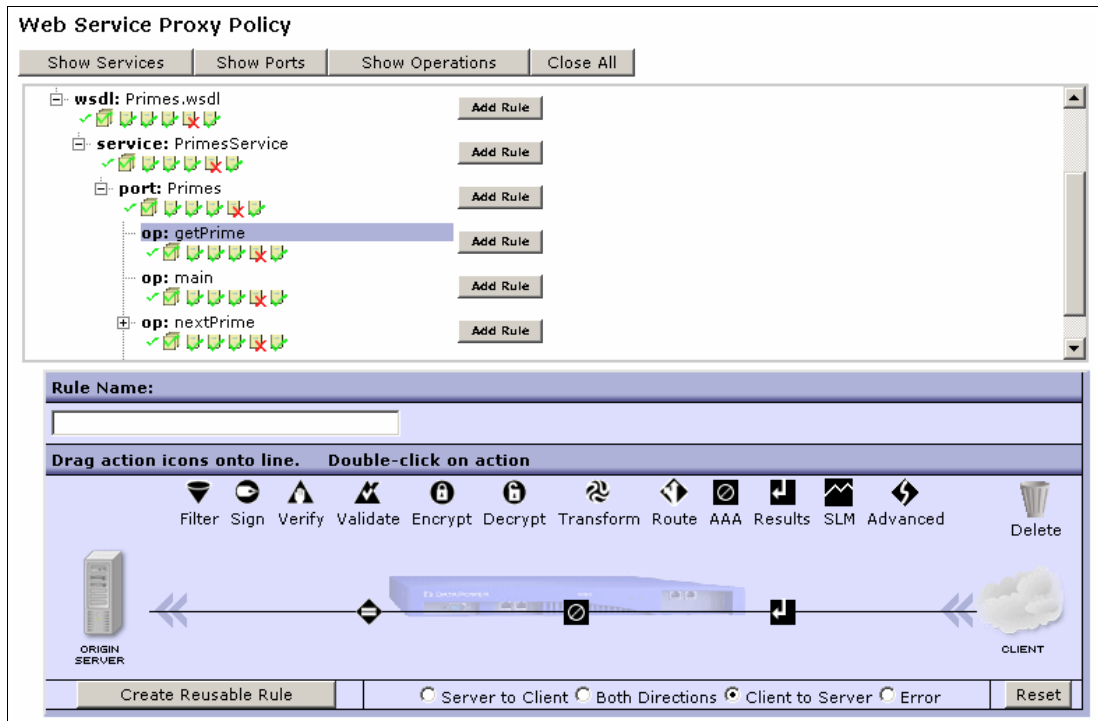


Figure 4-8 Processing rule

7. Click **Apply** to save the processing rule for the getPrime operation.
8. Click **Save Config** to make the changes permanent.
9. Repeat steps 3 on page 72 through 8 for the nextPrime and random operations of the ITSO_PRIMES_TDS Web service proxy.

4.4 Creating Tivoli Directory Server AAA policies

After the security policy of ITSOCorp has been applied to Tivoli Directory Server, you must create two AAA policies so that the Web service proxy ITSO_PRIMES_TDS can enforce that policy. In 3.2, “Creating the Tivoli Access Manager AAA policy” on page 35, we used the AAA policy wizard to create the AAA policy for Tivoli Access Manager. In this scenario, we use an alternative method to create the AAA policies.

4.4.1 Creating the Primes division AAA policy object

To create the Tivoli Directory Server AAA policy for the Primes division:

1. Navigate to **Objects** → **XML Processing** → **AAA Policy** in the DataPower WebGUI.
2. On the Configure AAA Policy page (Figure 4-9), click **Add**.

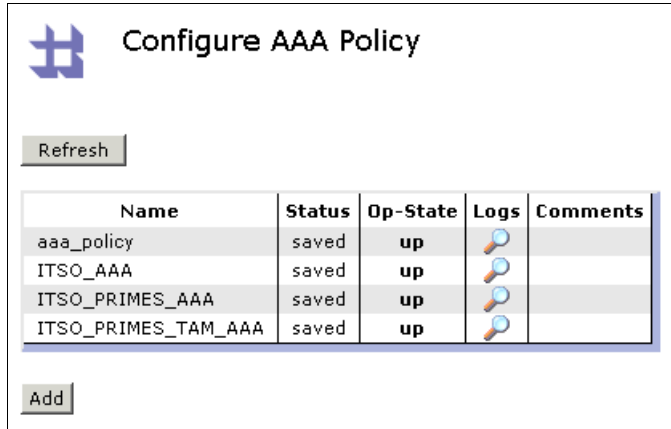


Figure 4-9 Creating the Tivoli Directory Server AAA policy

3. On the Main page (Figure 4-10), for Name of the AAA policy, enter ITSO_PRIMES_TDS_AAA.

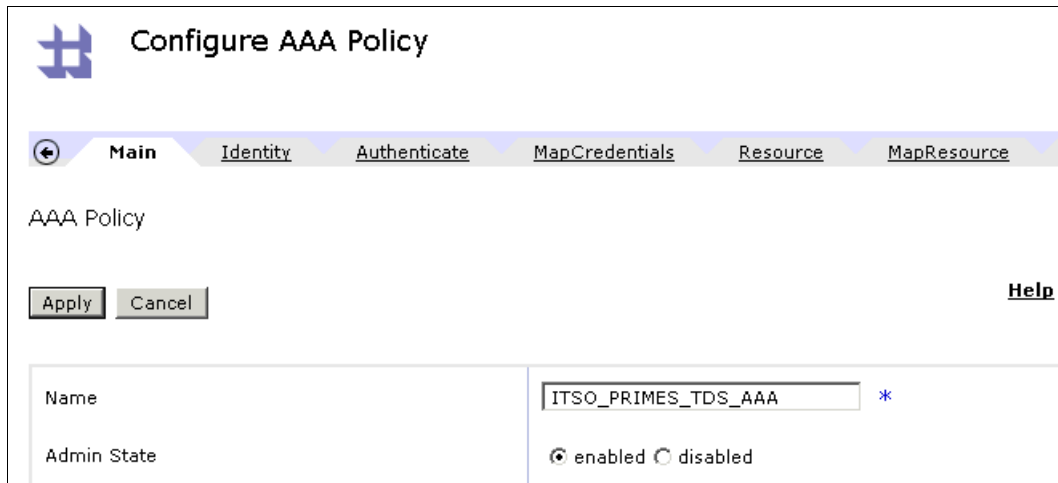
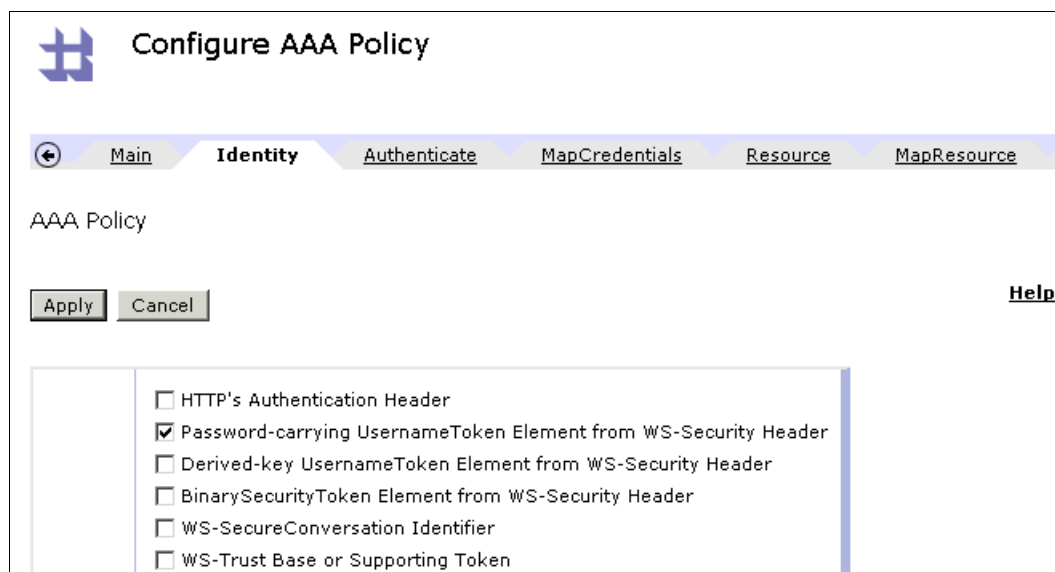


Figure 4-10 Entering the name of the AAA policy

Extracting the identity

The first stage of the AAA policy is the extraction of the user's identity. On the Configure AAA Policy page, click the **Identity** tab. On the Identity page (Figure 4-11), select the **Password-carrying UsernameToken element from WS-Security Header** option.



The screenshot displays the 'Configure AAA Policy' window. At the top, there is a navigation bar with tabs: 'Main', 'Identity' (which is active), 'Authenticate', 'MapCredentials', 'Resource', and 'MapResource'. Below the tabs, the text 'AAA Policy' is visible. There are 'Apply' and 'Cancel' buttons on the left, and a 'Help' link on the right. A list of identity extraction methods is shown with checkboxes:

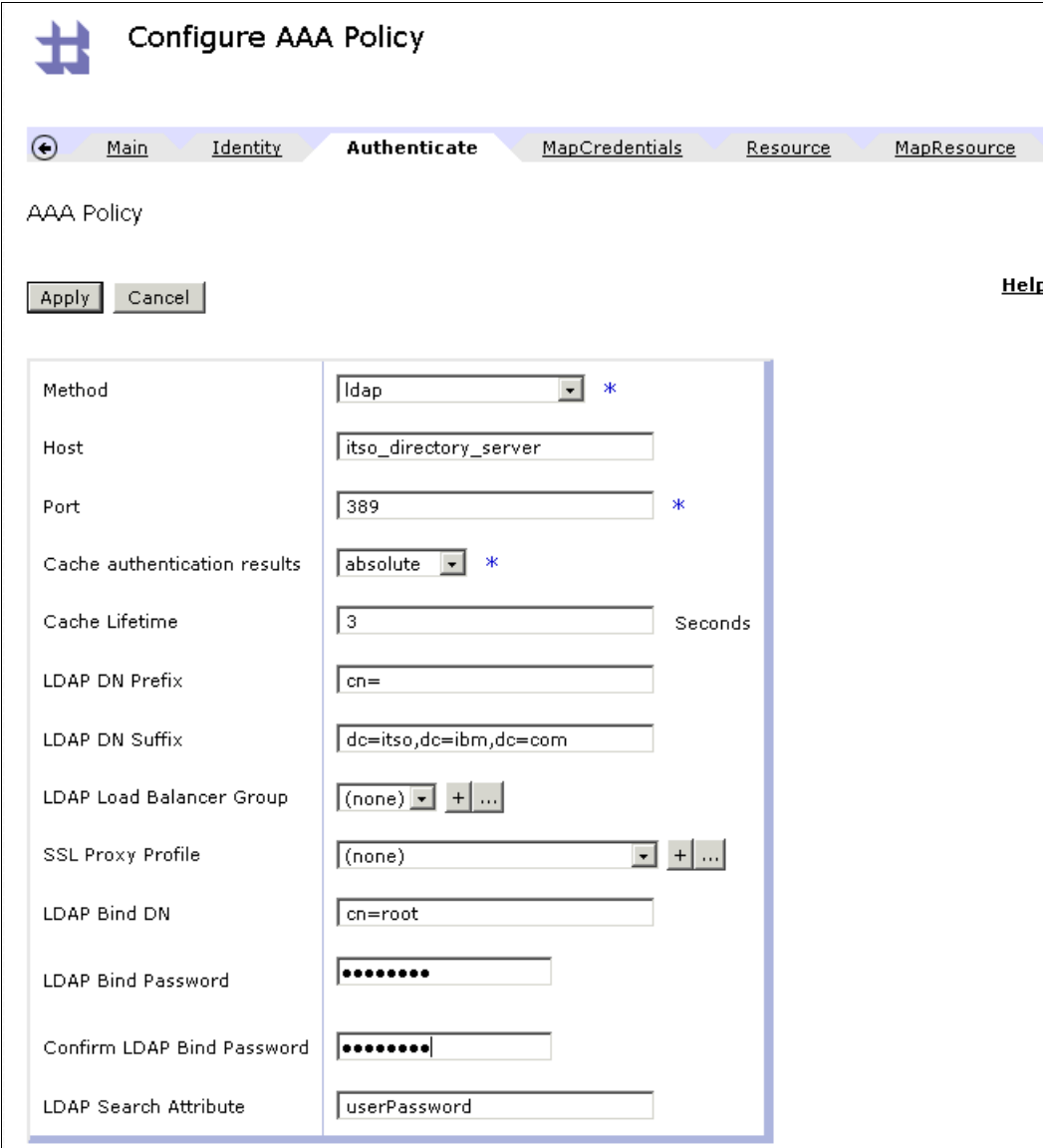
- ☐ HTTP's Authentication Header
- ☒ Password-carrying UsernameToken Element from WS-Security Header
- ☐ Derived-key UsernameToken Element from WS-Security Header
- ☐ BinarySecurityToken Element from WS-Security Header
- ☐ WS-SecureConversation Identifier
- ☐ WS-Trust Base or Supporting Token

Figure 4-11 Selecting the identity extraction method

Authenticating requests

The next logical stage of the AAA policy is the authenticate step. On the Configure AAA Policy page, click the **Authenticate** tab. On the Authenticate page, enter the appropriate information for the Tivoli Directory Server server location and bind the DN as shown in Figure 4-12.

Host alias: The host alias `itso_directory_server` is used to define the location of Tivoli Directory Server. This alias is configured in the DataPower WebGUI by navigating to **Network** → **Interface** → **Host Alias**.



Configure AAA Policy

← Main Identity **Authenticate** MapCredentials Resource MapResource

AAA Policy

Apply Cancel [Help](#)

Method	ldap *
Host	itso_directory_server
Port	389 *
Cache authentication results	absolute *
Cache Lifetime	3 Seconds
LDAP DN Prefix	cn=
LDAP DN Suffix	dc=itso,dc=ibm,dc=com
LDAP Load Balancer Group	(none) + ...
SSL Proxy Profile	(none) + ...
LDAP Bind DN	cn=root
LDAP Bind Password
Confirm LDAP Bind Password
LDAP Search Attribute	userPassword

Figure 4-12 Configuring the authentication mechanism for ITSO_PRIMES_TDS_AAA

Mapping the credentials

You do not need to perform any credential mapping in this scenario. On the Configure AAA Policy page, click the **Resource** tab to configure the resource extraction method.

Extracting the resources

You must define the resource that the request is trying to access. As described in Chapter 3, “AAA with Tivoli Access Manager” on page 29, several mechanisms are available in this step. Like the Tivoli Access Manager AAA policy, you select the **Local Name of Request Element** option as shown in Figure 4-13.

Configure AAA Policy

← Main Identity Authenticate MapCredentials **Resource** MapResource

AAA Policy

Apply Cancel [Help](#)

Resource Information

- ☐ URL Sent to Back End
- ☐ URL Sent by Client
- ☐ URI of Toplevel Element in the Message
- ☒ Local Name of Request Element
- ☐ HTTP Operation (GET/POST)
- ☐ XPath Expression
- ☐ Processing Metadata

*

Figure 4-13 Configuring the resource extraction mechanism

Mapping the resources

You do not need to perform any resource mapping in this scenario. On the Configure AAA Policy page, click the **Authorize** tab to configure the authorization mechanism.

Configuring the authorization

Next, you configure the authorization mechanism. For this AAA policy, you check the user's membership in the primes Tivoli Directory Server (LDAP) group. Configure the Authorize page as shown in Figure 4-14 and click **Apply**.

Configure AAA Policy

AAA Policy

[Main](#) [Identity](#) [Authenticate](#) [MapCredentials](#) [Resource](#) [MapResource](#)

[Apply](#) [Cancel](#) [Help](#)

Method	ldap *
Host	itso_directory_server
Port	389 *
Group DN	cn=itso-primes,dc=itso,dc=ibm,dc=com *
Cache authorization results	absolute *
Cache Lifetime	3 Seconds
LDAP Load Balancer Group	(none) + ...
Bind DN	cn=root
Bind Password
Confirm Bind Password
LDAP Group Attribute	member
SSL Proxy Profile	(none) + ...
LDAP Search Scope	subtree
LDAP Search Filter	

Figure 4-14 Configuring the authorization stage for the Primes AAA policy

Performing the post processing

No post processing is required by this scenario.

Applying the AAA policy

Click **Apply** and then click **Save Config** to save the AAA policy.

4.4.2 Creating the Random division AAA policy object

The Random division AAA policy is created in exactly the same way as the Primes AAA policy that we describe in the previous sections. The only divergence between the two policies is at the authorize stage of processing, where the group membership check is different.

Remember that the Primes and Random division users are members of the primes and random Tivoli Directory Server groups respectively. Therefore, the AAA policy for the random operation of the Web service must ensure that only Random division users (that is, members of the random Tivoli Directory Server group) are requesting access to the random operation. To achieve this, enter a different group DN, as shown in Figure 4-15, in the authorize step of AAA.

The screenshot shows the 'Configure AAA Policy' window with the 'Authenticate' tab selected. The 'Group DN' field is highlighted with a blue box. The 'Group DN' field contains the text 'cn=itso-random,dc=itso,dc=ibm,c'. Other fields include 'Method' (ldap), 'Host' (itso_directory_server), 'Port' (389), 'Cache authorization results' (absolute), 'Cache Lifetime' (3), 'LDAP Load Balancer Group' ((none)), 'Bind DN' (cn=root), 'Bind Password' (masked), 'Confirm Bind Password' (masked), 'LDAP Group Attribute' (member), 'SSL Proxy Profile' ((none)), 'LDAP Search Scope' (subtree), and 'LDAP Search Filter' ((objectClass=*)).

Field	Value
Method	ldap *
Host	itso_directory_server
Port	389 *
Group DN	cn=itso-random,dc=itso,dc=ibm,c *
Cache authorization results	absolute *
Cache Lifetime	3 Seconds
LDAP Load Balancer Group	(none) + ...
Bind DN	cn=root
Bind Password
Confirm Bind Password
LDAP Group Attribute	member
SSL Proxy Profile	(none) + ...
LDAP Search Scope	subtree
LDAP Search Filter	(objectClass=*)

Figure 4-15 Configuring the authorization stage for the Random AAA policy

4.5 Applying the AAA policies to the Web service proxy

Now that we have created the Tivoli Directory Server AAA policies, we must associate them with the appropriate processing rules created in 4.3.1, “Creating the processing rules” on page 71.

4.5.1 Configuring the getPrime and nextPrime operations

The AAA action of the getPrime operation’s processing rule was created but not configured to use a AAA policy. To configure the rule to use the AAA policy that corresponds to the Primes division users:

1. Click **Control Panel** → **Services** → **Web Service Proxy** → **ITSO_PRIMES_TDS** → **Policy** to navigate to the ITSO_PRIMES_TDS Web Service Proxy Policy tab.
2. On the Web Service Proxy Policy page (Figure 4-16), click the **Show Operations** button and expand the **getPrime** operation level to select the processing rule. Double-click the **AAA action**.

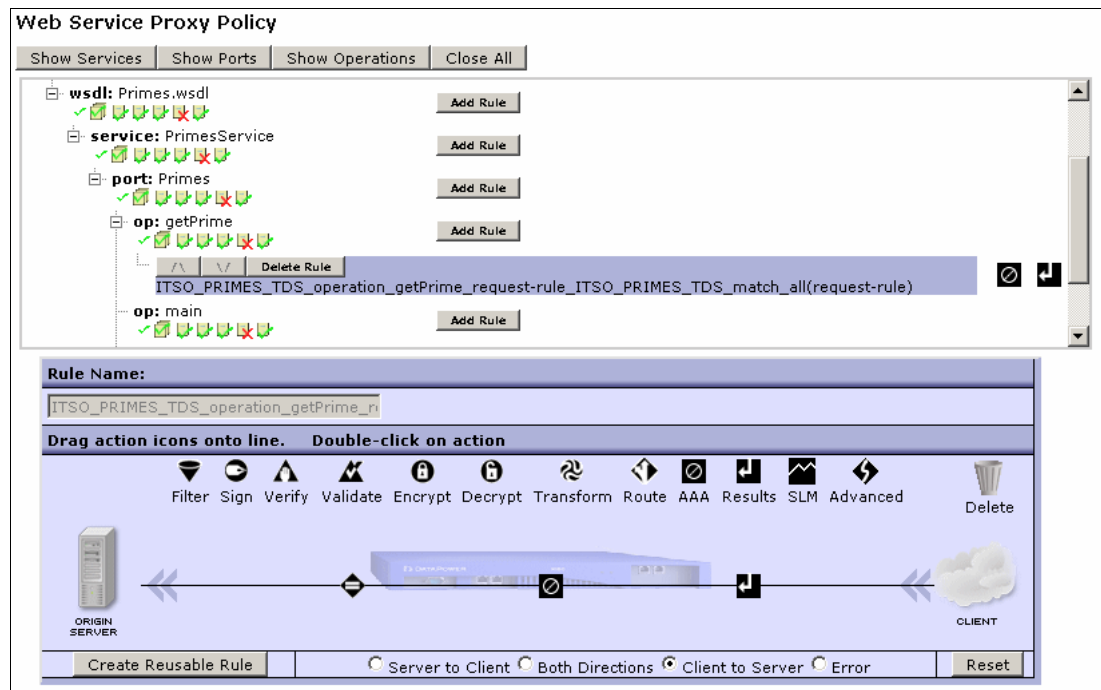


Figure 4-16 Editing the getPrime processing rule

3. On the Configure AAA Action page (Figure 4-17), for AAA Policy, select the **ITSO_PRIMES_TDS_AAA** policy. Click **Done**. You return the Policy tab of the ITSO_PRIMES_TDS Web service proxy.

Figure 4-17 Setting the ITSO_PRIMES_TDS_AAA policy of the getPrime AAA action

4. Repeat step 1 on page 82 through step 3 for the nextPrime operation of the ITSO_PRIMES_TDS Web service proxy.
5. Click **Apply** and then click **Save Config** to commit the changes.

4.5.2 Configuring the random operation

The random operation of the Primes Web service must allow access by Random division users of ITSO Corp. By using the same method as in the previous section, we configure the AAA action of the random operation to use a AAA policy to allow access by the ITSO Corp Random division users.

To configure the rule to use the AAA policy that corresponds to the Random division users:

1. Click **Control Panel** → **Services** → **Web Service Proxy** → **ITSO_PRIMES_TDS** → **Policy** to navigate to the ITSO_PRIMES_TDS Web Service Proxy Policy tab.
2. On the Web Service Proxy Policy page (Figure 4-18), click the **Show Operations** button and expand the **random** operation level to select the processing rule. Double-click the **AAA action**.

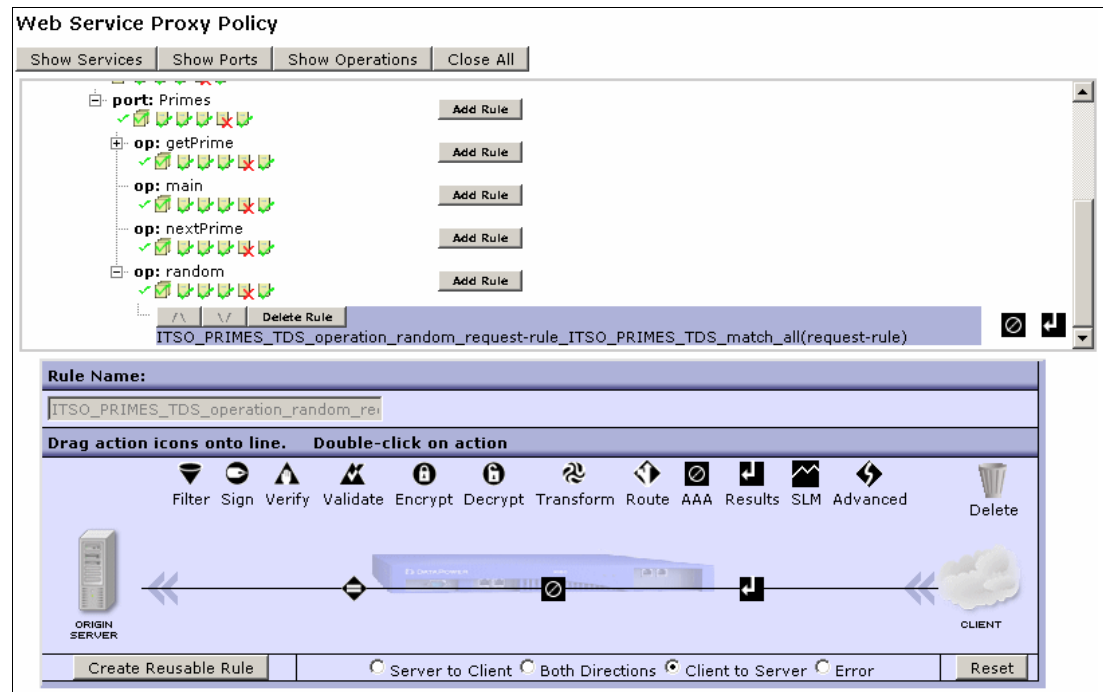


Figure 4-18 Editing the random processing rule

3. On the Configure AAA Action page (Figure 4-19), for the AAA Policy field, select the **ITSO_RANDOM_TDS_AAA** policy. Then click **Done**.

DATAPOWER XI50 [Help](#)

Configure AAA Action

Input

Input INPUT INPUT

Options

☒ AAA

AAA Policy ITSO_RANDOM_TDS_AAA + ... [up]

Output

Output tempvar1 tempvar1

Delete Done Cancel

Figure 4-19 Setting the *ITSO_Random_TDS_AAA* policy of the random AAA action

4. Click **Apply** and then click **Save Config** to commit the changes.

4.6 Testing

To test the functionality of the Tivoli Directory Server AAA policies, we can use the same tests here that were used in 3.4, “Testing” on page 54.

4.7 Summary

In this chapter, we constructed a AAA policy by using IBM Tivoli Directory Server. Tivoli Directory Server is the cornerstone of any enterprise identity management infrastructure.

We demonstrated the use of two AAA policies by using Tivoli Directory Server to enforce authentication and authorization. In addition, we showed an alternative approach to building a AAA policy.



A

External keys

The certificates and keys that are used to configure the DataPower appliance to accept HTTPS connections in Chapter 1, “Authentication and authorization” on page 1, were created on the appliance itself. In this appendix, we explain how to create keys by using the Global Secure ToolKit (GSKit) tool **ikeyman** to create keystores and certificates and upload them to the DataPower appliance.

GSKit ikeyman

As a prerequisite for this section, you must have a functioning GSKit installation. If you have a Tivoli Access Manager or WebSphere installation, GSKit will be installed on the system.

Creating a keystore

To create a keystore:

1. Launch the **ikeyman** utility. Ensure that you have a JAVA_HOME environment variable set in your path that references a valid Java™ run time.
2. Navigate to the bin directory of the GSKit installation, and run the **gsk7ikm** file.
3. In the IBM Key Management window (Figure A-1), click the **Create keystore** button.

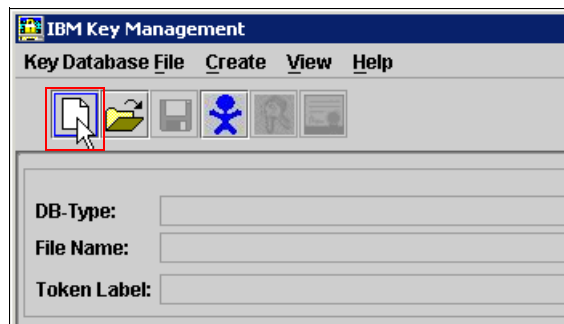


Figure A-1 Key database file creation

4. In the New window (Figure A-2), specify the database type, and enter the file name and location. Click **OK**.

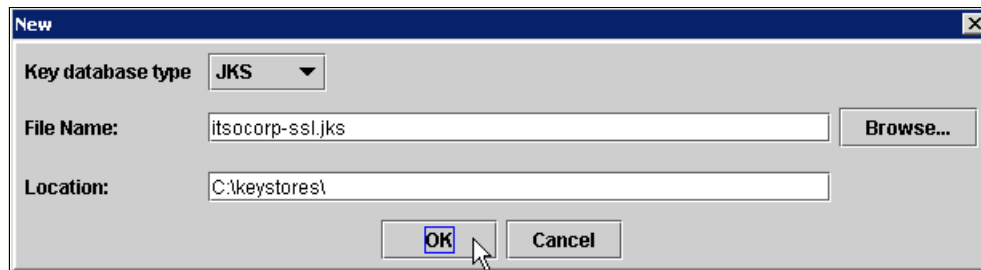


Figure A-2 Entering the keystore details

5. In the Password Prompt window (Figure A-3), enter a password for the keystore and then confirm the password. Click **OK**. The keystore is created.

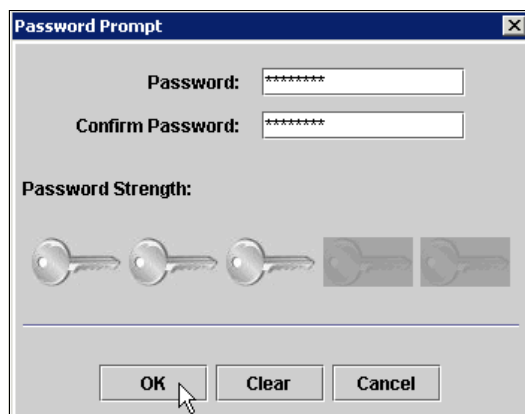


Figure A-3 Entering the password for the keystore

Figure A-4 shows the signer certificates for the keystore in the ikeyman window.

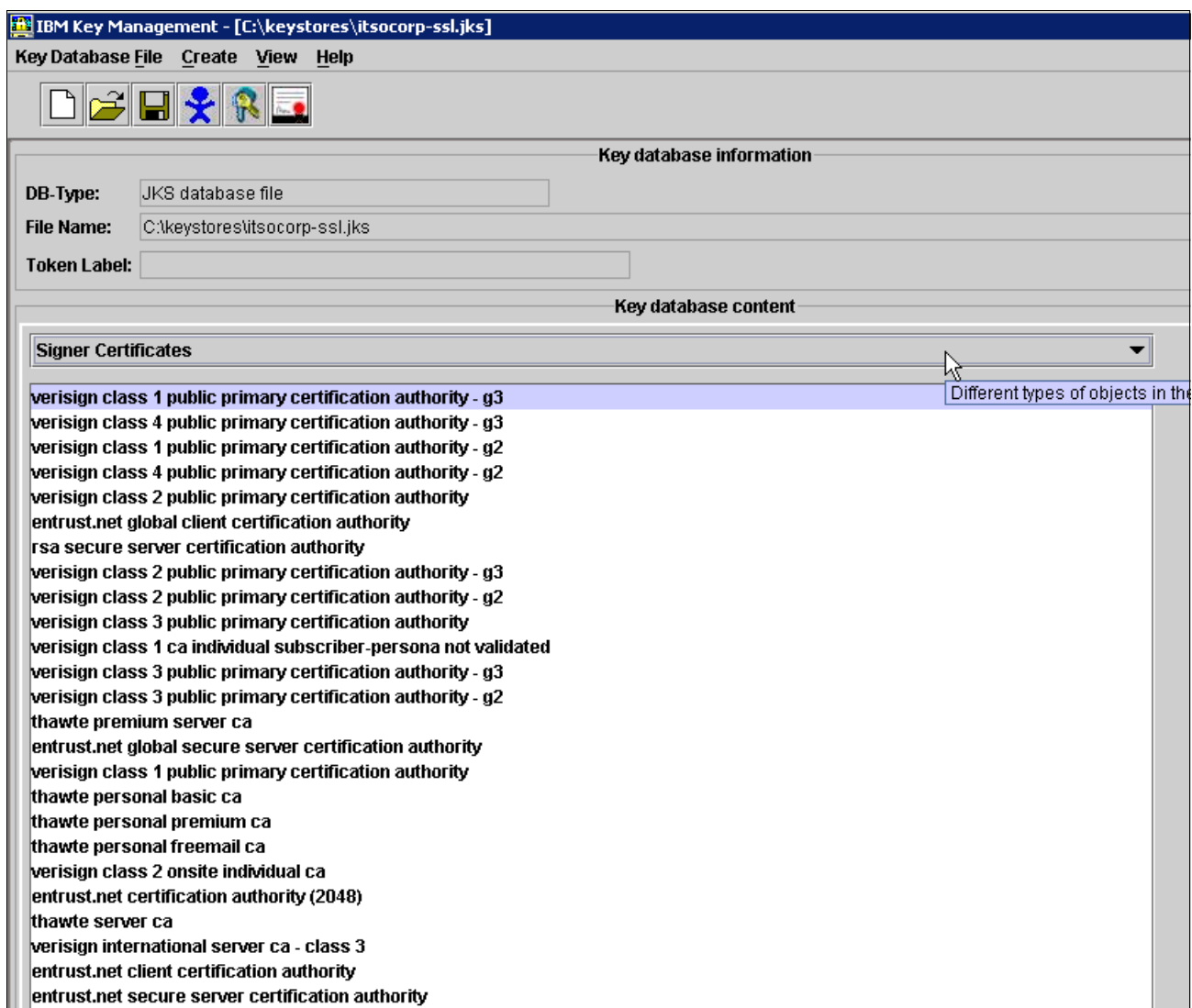


Figure A-4 Signer certificates for the itsocorp-ssl.jks keystore

Generating a certificate

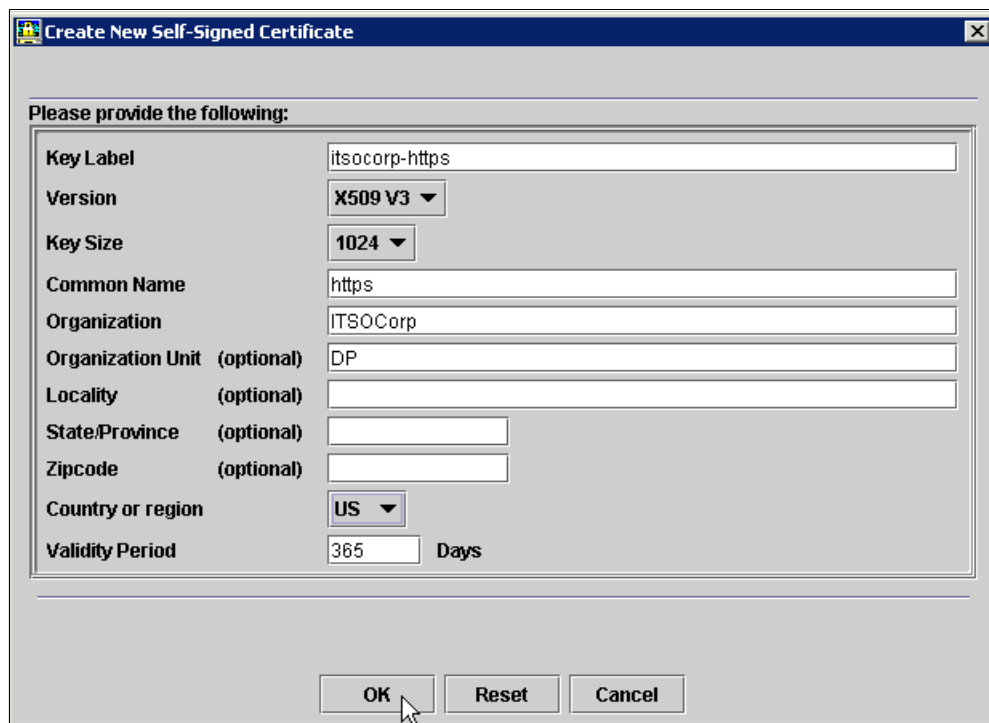
To create a self-signed certificate and private key in the keystore:

1. Click the **New Self-Signed** button shown in Figure A-5.



Figure A-5 Creating a self signed certificate

2. In the Create New Self-Signed Certificate window (Figure A-6), enter the appropriate details for the certificate. Click **OK** to generate the certificate.



Create New Self-Signed Certificate

Please provide the following:

Key Label	itsocorp-https	
Version	X509 V3 ▼	
Key Size	1024 ▼	
Common Name	https	
Organization	ITSOCorp	
Organization Unit (optional)	DP	
Locality (optional)		
State/Province (optional)		
Zipcode (optional)		
Country or region	US ▼	
Validity Period	365	Days

OK Reset Cancel

Figure A-6 Entering the details for the certificate

The ikeyman window should now resemble the window shown in Figure A-7.

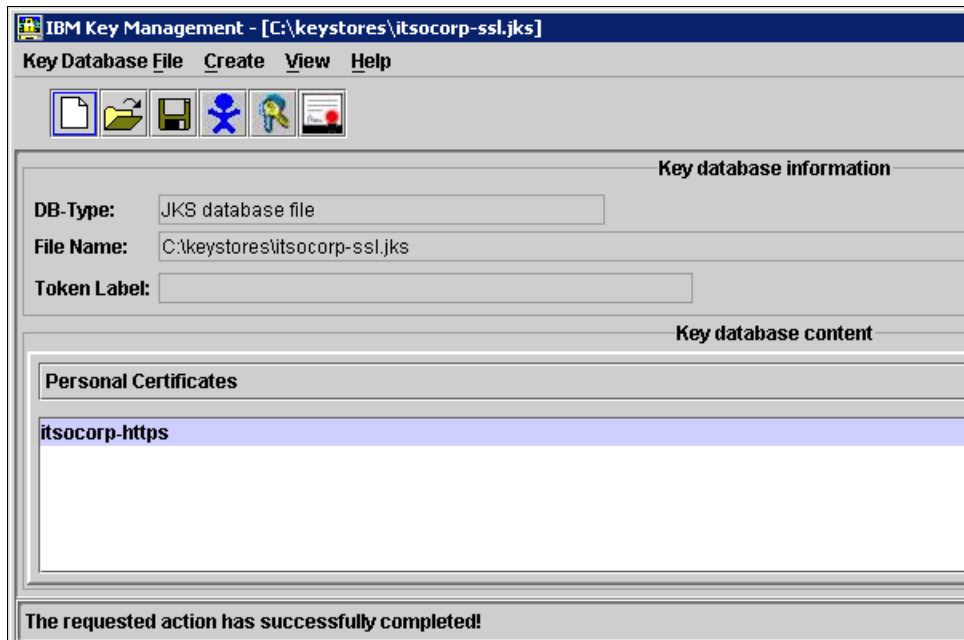


Figure A-7 Certificate created

3. The certificate creation is now complete. Shut down ikeyman.

Uploading the certificate and key to the DataPower appliance

Now we must upload the certificate and key from the keystore into the DataPower appliance as explained in the following sections.

Uploading the certificate

To upload the certificate to the appliance:

1. Navigate to **Administration** → **Crypto Tools**. Click the **Import Crypto Object** tab.
2. On the Import Crypto Object page (Figure A-8), click **Upload** to upload the certificate to the appliance.

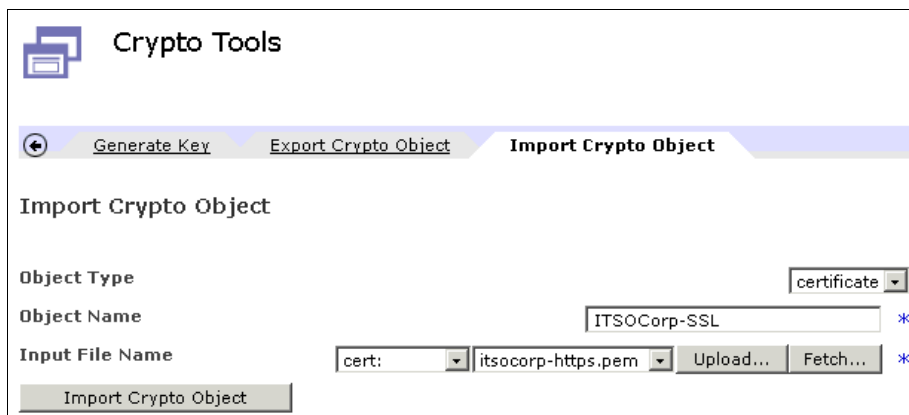


Figure A-8 Importing the certificate from the JKS keystore

3. In the File Management window (Figure A-9), complete the following tasks:
 - a. Select the **Java Key Store** radio button.
 - b. Click **Browse** to locate the keystore.
 - c. In the Key store password field, enter the password for the keystore.
 - d. For Key to upload, select the self-signed certificate, which is **itsocorp-http (certificate 0)** in this example.
 - e. Click **Upload** to copy the file to the device.

Figure A-9 Uploading the certificate

4. The WebGUI returns control to the Import Crypto Object page (Figure A-8 on page 91). On this page, ensure that the following fields are set:
 - For Object Type, select **certificate**.
 - For Object Name, type **ITS0Corp-SSL**.
 - For Input File Name, select **cert:** and **itsocorp-https-cert-0.pem**.

Uploading the key

To upload the certificate to the DataPower appliance:

1. Navigate to **Administration** → **Crypto Tools**. Click the **Import Crypto Object** tab.
2. On the Import Crypto Object page (Figure A-10), click **Upload**.

Crypto Tools

Generate Key Export Crypto Object **Import Crypto Object**

Import Crypto Object

Object Type certificate

Object Name ITSOCorp-SSL *

Input File Name cert: itsocorp-https.pem Upload... Fetch... *

Import Crypto Object

Figure A-10 Importing the certificate from the JKS keystore

3. In the File Management window (Figure A-11), complete the following tasks:
 - a. Select the **Java Key Store** radio button.
 - b. Click **Browse** to locate the keystore.
 - c. In the Key store password field, enter the password for the keystore.
 - d. For Key to upload, select the key, which is **itsocorp-http (key)** in this example.
 - e. Click **Upload** to copy the file to the device.

Figure A-11 Uploading the key

4. The WebGUI returns control to the Import Crypto Object page (Figure A-10 on page 93). On this page, ensure that the following fields are set:
 - For Object Type, select **certificate**.
 - For Object Name, type ITS0Corp-SSL.
 - For Input File Name, select **cert:** and **itsocorp-https.pem**.

Creating the DataPower Crypto object

Now create the DataPower Crypto Identification and Validation Credentials as explained in the following sections.

Crypto Identification Credentials

You must associate the certificate and key together that were imported in the previous section in Crypto Identification Credential object:

1. Navigate to **Objects** → **Crypto** → **Crypto Identification Credentials** by using the DataPower WebGUI.
2. On the Configure Crypto Identification Credentials page (Figure A-12), click the + button next to Crypto Key.

Configure Crypto Identification Credentials

Main

Crypto Identification Credentials

Apply Cancel

Name itsocorp-ssl *

Admin State ☒ enabled ☐ disabled

Crypto Key (none) + ... *

Certificate (none) + ... *

Intermediate CA Certificate

Delete Add + ...

Figure A-12 Creating a Crypto Identification Credential

3. In the Configure Crypto Key window, complete the fields as shown in Figure A-13 and click **Apply**.

https://9.42.170.230:9090 - DataPower XI50 - Configure: Crypto Key - Microsoft Internet ...

Troubleshooting Enabled (The performance of the device may be impacted!)

Configure Crypto Key

Main

Crypto Key

Apply Cancel

Name	itsocorp-ssl-key *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
File Name	cert: itsocorp-https.pem Upload... Fetch... *
Password
Confirm Password
Password Alias	<input type="radio"/> on <input checked="" type="radio"/> off

Internet

Figure A-13 Configuring the Crypto Key object

4. The WebGUI returns control to the Configure Crypto Identification Credential page (Figure A-14). Notice that the Crypto Key field has the newly created Crypto Key object. Click the + button next to Certificate.

Figure A-14 Crypto Identification Credential with new Crypto Key

5. In the Crypto Certificate window (Figure A-15), click **Apply**.

https://9.42.170.230:9090 - DataPower XI50 - Configure: Crypto Certificate - Microsoft Int...

Main

Crypto Certificate

Apply Cancel

Name itsocorp-ssl-cert *

Admin State ☒ enabled ☐ disabled

File Name cert: itsocorp-https-cert-0.pem Details... Upload...

Password

Confirm Password

Password Alias ☐ on ☒ off

Ignore Expiration Dates ☒ on ☐ off

Done Internet

Figure A-15 Configuring the Crypto Certificate object

6. The WebGUI returns control to the Configure Crypto Identification Credential page, which now shows the newly created Crypto Certificate object (Figure A-16). Click **Apply** and then click **Save Config** to save the changes.

Configure Crypto Identification Credentials

Main

Crypto Identification Credentials

Apply **Cancel**

Name	itsocorp-ssl *
Admin State	<input checked="" type="radio"/> enabled <input type="radio"/> disabled
Crypto Key	itsocorp-ssl-key + ... *
Certificate	itsocorp-ssl-cert + ... *
Intermediate CA Certificate	<div><div></div><div>Delete <div></div> Add + ...</div></div>

Figure A-16 Configured Crypto Identification Credential

Crypto Validation Credentials

Now you must add the certificate that was imported in the previous section to a Crypto Validation Credential object:

1. Navigate to **Objects** → **Crypto** → **Crypto Validation Credentials** by using the DataPower WebGUI.
2. On the Configure Crypto Validation Credentials page (Figure A-17), for Certificates, from the drop-down list (next to the Delete button), select the Crypto Certificate object and click **Add**. Click **Apply** to return to the Crypto Validation Credentials page.

Figure A-17 Configuring the Crypto Validation Credential object

3. Click **Save Config** to save the configuration.



Additional material

This paper refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this paper is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser at:

<ftp://www.redbooks.ibm.com/redbooks/REDP4364>

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the IBM Redpaper form number, REDP4364.

Using the Web material

The additional Web material that accompanies this paper includes the following files:

<i>File name</i>	<i>Description</i>
redp4364.zip	Compressed code samples

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	40 MB minimum
Operating System:	Microsoft® Windows®
Processor:	2 GB or higher
Memory:	2 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and extract the contents of the Web material compressed file into this folder.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 104. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *Enabling SOA Using WebSphere Messaging*, SG24-7163
- ▶ *Enterprise Security Architecture Using IBM Tivoli Security Solutions*, SG24-6014
- ▶ *IBM WebSphere DataPower SOA Appliances Part I: Overview and Getting Started*, REDP-4327
- ▶ *IBM WebSphere DataPower SOA Appliances Part III: XML Security Guide*, REDP-4365
- ▶ *IBM WebSphere DataPower SOA Appliances Part IV: Management and Governance*, REDP-4366
- ▶ *Patterns: SOA Design Using WebSphere Message Broker and WebSphere ESB*, SG24-7369
- ▶ *Understanding LDAP - Design and Implementation*, SG24-4986
- ▶ *WebSphere Message Broker Basics*, SG24-7137

Other publications

The following publications are also relevant as further information sources. They are either available as part of the product or orderable for a fee. The common documentation is available on the Web at the following address:

<http://www-1.ibm.com/support/docview.wss?rs=2362&uid=swg24014405>

- ▶ *IBM WebSphere DataPower Example Configurations Guide*
- ▶ *IBM WebSphere DataPower Common Installation Guide*
- ▶ *IBM WebSphere DataPower Integration Appliance XI50 Reference Kit*, part number 42C4212
- ▶ *IBM WebSphere DataPower WebGUI User's Guide*
- ▶ *IBM WebSphere DataPower XML Integration Appliance XI50 CLI Reference Guide Release 3.6.0*
- ▶ *IBM WebSphere DataPower XML Accelerator XA35 Reference Kit*, part number 42C4210
- ▶ *IBM WebSphere DataPower XML Security Gateway XS40 Reference Kit*, part number 42C4211

Online resources

These Web sites are also relevant as further information sources:

- ▶ Integrating WebSphere DataPower SOA Appliances with WebSphere MQ
http://www.ibm.com/developerworks/websphere/library/techarticles/0703_crocker/0703_crocker.html
- ▶ Integrating WebSphere DataPower XML Security Gateway XS40 with WebSphere Message Broker
http://www.ibm.com/developerworks/websphere/library/techarticles/0710_crocker/0710_crocker.html
- ▶ Integrating DataPower with WebSphere Message Broker using the Broker Explorer
http://www.ibm.com/developerworks/websphere/library/techarticles/0707_storey/0707_storey.html

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



IBM WebSphere DataPower SOA Appliances

Part II: Authentication and Authorization



**Integrate IBM Tivoli
Access Manager with
your DataPower
appliance**

**Implement enterprise
security and identity
management**

**Configure
authentication and
authorization using
LDAP**

IBM WebSphere DataPower SOA Appliances represent an important element in the holistic approach of IBM to service-oriented architecture (SOA). IBM SOA appliances are purpose-built, easy-to-deploy network devices that simplify, secure, and accelerate XML and Web services deployments while extending the SOA infrastructure. These appliances offer an innovative, pragmatic approach to harness the power of SOA. By using them, you can simultaneously use the value of existing application, security, and networking infrastructure investments.

This series of IBM Redpaper publications is written for architects and administrators who need to understand the implemented architecture in WebSphere DataPower appliances to successfully deploy it as a secure and efficient enterprise service bus (ESB) product. These papers give a broad understanding of the new architecture and traditional deployment scenarios. They cover details about the implementation to help identify the circumstances under which to deploy DataPower appliances. They include a sample implementation and architectural best practices for an SOA message-oriented architecture in an existing production ESB environment.

This part of the series explains ways to integrate the DataPower appliance with other products such as IBM Tivoli Access Manager and Tivoli Directory Server. The entire series includes the following papers:

- ▶ *IBM WebSphere DataPower SOA Appliances Part I: Overview and Getting Started*, REDP-4327
- ▶ *IBM WebSphere DataPower SOA Appliances Part II: Authentication and Authorization*, REDP-4364
- ▶ *IBM WebSphere DataPower SOA Appliances Part III: XML Security Guide*, REDP-4365
- ▶ *IBM WebSphere DataPower SOA Appliances Part IV: Management and Governance*, REDP-4366

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks