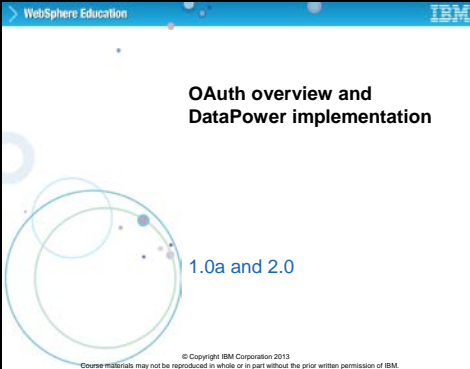


Slide 1



The slide features a blue header bar with the text "WebSphere Education" on the left and the IBM logo on the right. The main content area has a light blue background with a decorative graphic of overlapping circles and dots on the left side. The title "OAuth overview and DataPower implementation" is centered in the upper half. Below the title, the text "1.0a and 2.0" is displayed. At the bottom, there is a small copyright notice.

WebSphere Education

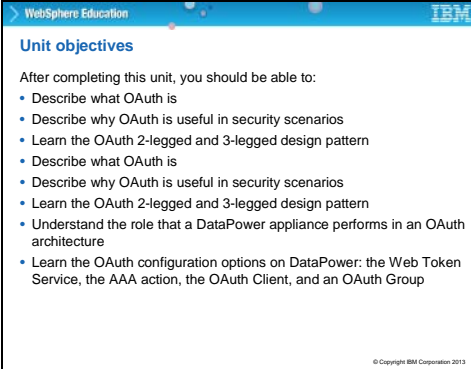
IBM

**OAuth overview and
DataPower implementation**

1.0a and 2.0

© Copyright IBM Corporation 2013
Course materials may not be reproduced in whole or in part without the prior written permission of IBM.

Slide 2



The slide is titled 'WebSphere Education' in the top left corner and features the IBM logo in the top right corner. The main heading is 'Unit objectives' in blue. Below this, it states 'After completing this unit, you should be able to:' followed by a bulleted list of objectives. The list includes: 'Describe what OAuth is', 'Describe why OAuth is useful in security scenarios', 'Learn the OAuth 2-legged and 3-legged design pattern', 'Understand the role that a DataPower appliance performs in an OAuth architecture', and 'Learn the OAuth configuration options on DataPower: the Web Token Service, the AAA action, the OAuth Client, and an OAuth Group'. A small copyright notice '© Copyright IBM Corporation 2013' is located at the bottom right of the slide.

WebSphere Education

Unit objectives

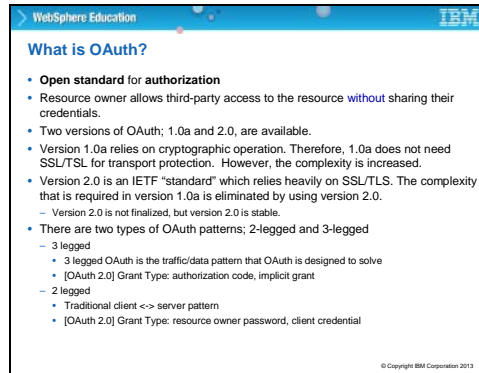
After completing this unit, you should be able to:

- Describe what OAuth is
- Describe why OAuth is useful in security scenarios
- Learn the OAuth 2-legged and 3-legged design pattern
- Describe what OAuth is
- Describe why OAuth is useful in security scenarios
- Learn the OAuth 2-legged and 3-legged design pattern
- Understand the role that a DataPower appliance performs in an OAuth architecture
- Learn the OAuth configuration options on DataPower: the Web Token Service, the AAA action, the OAuth Client, and an OAuth Group

© Copyright IBM Corporation 2013

This unit includes an introduction to the OAuth security pattern.

Slide 3



WebSphere Education IBM

What is OAuth?

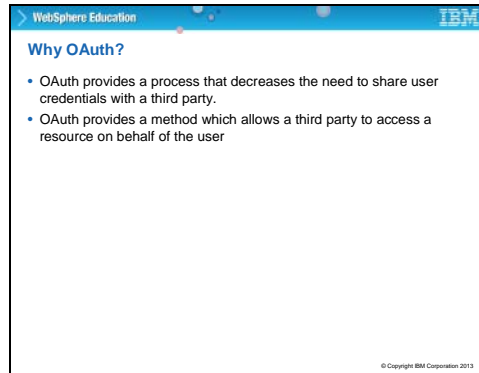
- **Open standard for authorization**
- Resource owner allows third-party access to the resource **without** sharing their credentials.
- Two versions of OAuth; 1.0a and 2.0, are available.
- Version 1.0a relies on cryptographic operation. Therefore, 1.0a does not need SSL/TLS for transport protection. However, the complexity is increased.
- Version 2.0 is an IETF "standard" which relies heavily on SSL/TLS. The complexity that is required in version 1.0a is eliminated by using version 2.0.
 - Version 2.0 is not finalized, but version 2.0 is stable.
- There are two types of OAuth patterns; 2-legged and 3-legged
 - 3-legged
 - 3-legged OAuth is the traffic/data pattern that OAuth is designed to solve
 - [OAuth 2.0] Grant Type: authorization code, implicit grant
 - 2-legged
 - Traditional client <-> server pattern
 - [OAuth 2.0] Grant Type: resource owner password, client credential

© Copyright IBM Corporation 2013

What is OAuth?

OAuth allows users to share their private resources (for example, photos, videos, contact lists) stored on one site with another site without having to hand out their credentials, typically supplying username and password tokens instead. Each token grants access to a specific site (for example, a video editing site) for specific resources (for example, just videos from a specific album) and for a defined duration (for example, the next 2 hours). The process allows a user to grant a third-party site access to their information stored with another service provider, without sharing their access permissions or the full extent of their data.

Slide 4



The slide is titled "WebSphere Education" and "IBM". The main heading is "Why OAuth?". It contains two bullet points:

- OAuth provides a process that decreases the need to share user credentials with a third party.
- OAuth provides a method which allows a third party to access a resource on behalf of the user

© Copyright IBM Corporation 2013

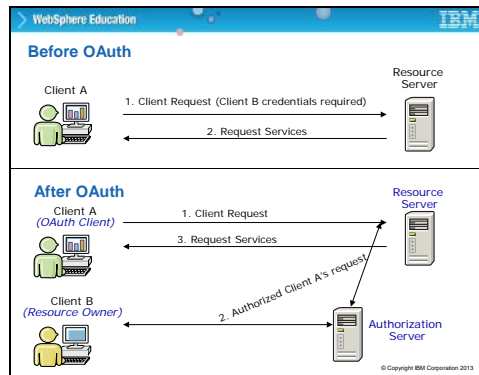
Why OAuth?

Many proprietary web authorization protocols emerged over the years: AuthSub (Google), OpenAuth (AOL), BBAuth (Yahoo), Upcoming API, Amazon Web Services API, to name a few. OAuth integrates the commonalities and adopts the good practices of these other web authorization protocols into a single open standard.

Other reasons for using OAuth authorization:

- Compatible with existing authorization methods
- Flexibility to adjust to security needs of different sites
- Extensible through different signing algorithms
- Designed to work with mobile devices and desktop applications

Slide 5



Before OAuth:

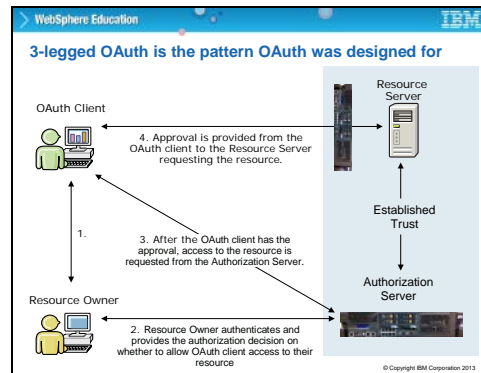
Before OAuth, if Client A requested information from a server as Client B, Client A would require Client B credentials. The process requires the sharing of credentials.

After OAuth:

When using OAuth, Client B's credentials are housed within the Authorization Server. Therefore, Client A needs permission only to use Client B's credentials. Client A is granted permission without ever receiving the actual credentials.

The four actors in an OAuth security scenario include;

- OAuth Client
- Resource Owner
- Authorization Server
- Resource Server



3-legged OAuth:

3-legged OAuth describes the scenario for which OAuth was originally developed: a resource owner wants to give a client access to a server without sharing the resource owner credentials (that is, username/password). A typical example is a user (resource owner) who wants to give a third-party application (client) access to the resource owner's Twitter account (server).

On a conceptual level, it works in the following way:

Client signed up to the server and got the client credentials (also known as "consumer key and secret") ahead of time

User wants to give the client access to the user protected resources on the server

Client retrieves the temporary credentials (also known as "request token") from the server

Client redirects the resource owner to the server

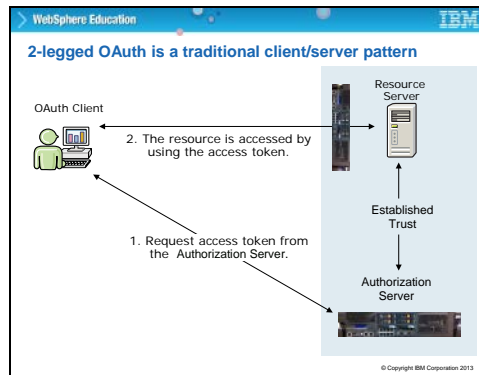
Resource owner grants the client access to the resource owner protected resources on the server

Server redirects the user back to the client

Client uses the temporary credentials to retrieve the token credentials (also known as "access token") from the server

Client uses the token credentials to access the protected resources on the server

Slide 7



2-legged OAuth:

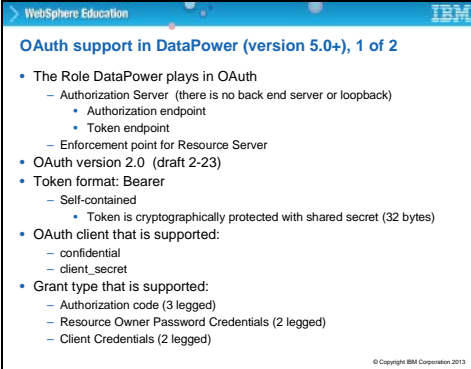
2-legged OAuth, describes a typical client/server scenario, without any user involvement. An example for such a scenario might be a local Twitter client application that accesses your Twitter account.


On a conceptual, level 2-legged OAuth consists of the first and last steps of 3-legged OAuth:

Client signed up to the server and got the client credentials (also known as "consumer key and secret")

Client uses the client credentials (and empty token credentials) to access the protected resources on the server

Slide 8



WebSphere Education 

OAuth support in DataPower (version 5.0+), 1 of 2

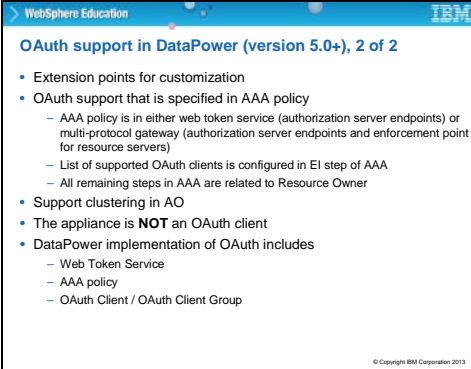
- The Role DataPower plays in OAuth
 - Authorization Server (there is no back end server or loopback)
 - Authorization endpoint
 - Token endpoint
 - Enforcement point for Resource Server
- OAuth version 2.0 (draft 2-23)
- Token format: Bearer
 - Self-contained
 - Token is cryptographically protected with shared secret (32 bytes)
- OAuth client that is supported:
 - confidential
 - client_secret
- Grant type that is supported:
 - Authorization code (3 legged)
 - Resource Owner Password Credentials (2 legged)
 - Client Credentials (2 legged)

© Copyright IBM Corporation 2013

OAuth support in DataPower

Starting with firmware version 5.0, DataPower participates in an OAuth configuration by being the Authorization endpoint and the Token endpoint. DataPower can also serve as the enforcement point for a resource server.

Slide 9



WebSphere Education

IBM

OAuth support in DataPower (version 5.0+), 2 of 2

- Extension points for customization
- OAuth support that is specified in AAA policy
 - AAA policy is in either web token service (authorization server endpoints) or multi-protocol gateway (authorization server endpoints and enforcement point for resource servers)
 - List of supported OAuth clients is configured in Ei step of AAA
 - All remaining steps in AAA are related to Resource Owner
- Support clustering in AO
- The appliance is **NOT** an OAuth client
- DataPower implementation of OAuth includes
 - Web Token Service
 - AAA policy
 - OAuth Client / OAuth Client Group

© Copyright IBM Corporation 2013

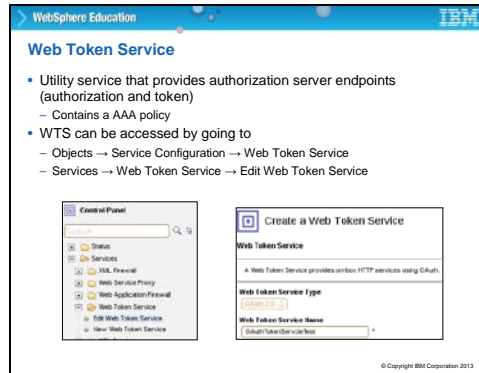
OAuth support that continued:

OAuth support for DataPower is tied to the AAA action. OAuth configuration occurs when a AAA is configured.

OAuth is supported in a clustered Application Optimization (AO) DataPower environment.

DataPower offers three components to implement OAuth. These three ways include:

- Web Token Service configuration
- AAA action configuration
- OAuth Client and OAuth Client Group configuration.



The slide is titled "WebSphere Education" and "Web Token Service". It contains a bulleted list of information about the service and two screenshots of the IBM WebSphere Administration Console.

- Utility service that provides authorization server endpoints (authorization and token)
 - Contains a AAA policy
- WTS can be accessed by going to
 - Objects → Service Configuration → Web Token Service
 - Services → Web Token Service → Edit Web Token Service

The first screenshot shows the "Control Panel" with a tree view on the left. The tree view is expanded to "Web Token Service". The second screenshot shows the "Create a Web Token Service" wizard. It has a "Web Token Service" section with a description: "A Web Token Service provides various HTTP services using OAuth." Below this is a "Web Token Service Type" section with a dropdown menu set to "OAuth 2.0". At the bottom, there is a text field for "Web Token Service Name" with the value "OAuthTokenServiceTest".

© Copyright IBM Corporation 2013

Web Token Services:

WTS configuration has three simple tabs – Main, Advanced, and Probe Settings.

Has support for HTTP and HTTPS front side handlers and allows configuration of a Processing Policy.

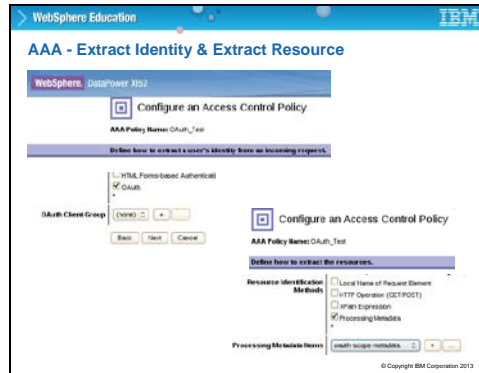
WTS wizard eases the configuration of the WTS to be an OAuth Authorization server.

Takes the inputs and then auto creates the processing policy at the time of commit.

The processing policy contains the required actions (such as http-convert) to configure an OAuth Token service.

Administrator can still modify the processing policy that is created by the wizard.

WTS wizard eases the configuration of Form-Based Login for ResourceOwner.



Using the Authentication, Authorization, and Audit (AAA) Action in DataPower:

To apply OAuth Protocol support, check the OAuth box. An OAuth Client Group contains a list of the OAuth clients that are supported by this AAA Policy. Use the processing metadata as the resource to authorize the access such as variables, and protocol headers. One can define only the metadata items with this object to return, If this object is not selected for the Extract Resource method, all the metadata items applicable for the current processing rule is returned.

The screenshot shows the 'Configure OAuth Client Group' page in the WebSphere Education environment. The page has a blue header with the IBM logo. On the left, there are two bullet points explaining the purpose of the configuration. The main area contains several configuration options: a 'Name' field, an 'Administrative State' section with 'enabled' and 'disabled' radio buttons, a 'Comments' text area, a 'Customized OAuth' checkbox, an 'OAuth Role' dropdown menu, and a 'Clients' list with 'Add' and 'Remove' buttons. The 'OAuth Role' is currently set to 'Authorization Server', and the 'Customized OAuth' checkbox is checked. The 'Clients' list is currently empty.

Configure a OAuth client group:

To support the OAuth 2.0 protocol, an AAA policy requires the configuration of an OAuth client group. An OAuth client group contains the configured OAuth clients that the DataPower appliance accepts requests from.

When creating an OAuth client group for an AAA policy, the OAuth client group consists of one or more OAuth clients with the **same** OAuth roles.

The customized OAuth indicates whether the configuration is for a customized OAuth client group. This property is mutually exclusive to the OAuth Role property.

The OAuth Role identifies the roles of the clients in the group. This property is mutually exclusive to the Customized OAuth property.

If the Authorization Server check box is selected, the DataPower appliance acts as an authorization server.

If the Resource server check box is selected, the DataPower appliance acts as a resource server.

The client manages the group of OAuth clients. Use the controls to add or remove clients from the group.

WebSphere Education

Configure OAuth Client, 1 of 2

- An OAuth client object:
 - represents the DataPower endpoint for an external OAuth client
 - is the basic building block for an OAuth client group.
- The following types of OAuth clients can be created:
 - Authorization server
 - Enforcement point for a resource server
 - Authorization server and an enforcement point for a resource server.

OAuth Client

Name:

Administrative State: ☒ enabled ☐ disabled

Comments:

Customized OAuth: ☐

OAuth Role: ☒ Authorization Server ☒ Resource Server

Client Type:

Supported Authorization Grant Type: ☒ Authorization Code ☐ Resource Owner Password Credentials ☐ Client Credentials

© Copyright IBM Corporation 2013

Configure a OAuth client group, 1 of 2

An OAuth client is the basic building block for an OAuth client group. When you create an OAuth client, you define its role.

The following type of OAuth clients can be created:

- * A client that acts as an authorization server.
- * A client that acts as an enforcement point for a resource server.
- * A client that acts as an authorization server and an enforcement point for a resource server.

When creating an OAuth client, style sheets for customization can be used.

A fully customized OAuth client can be created. A customized OAuth client allows one to customize clients as an authorization server and, optionally, as the enforcement point for a resource server. To create a customized OAuth client as only the enforcement point for a resource server, create the client as only a resource server.

Customization uses style sheets. These style sheets must be in the local: or store: directory. The style sheets must define the following behaviors that depend on:

- * How to verify requests. (This verification is required when the client acts an authorization server.)
- * For an authorization code grant type, how to issue and verify the authorization code. (This action is required when the client acts an authorization server.)
- * For an access token, how to issue the access token. (This action is required when the client acts an authorization server.)
- * For an access token, how to verify the access token. (This action is required when the client acts the enforcement point for a resource server.)

OAuth clients interact with protected resource in the following sequence:

1. The client requests access from the authorization server.
2. The authorization server grants access in the form of an access token
3. The client presents the access token to access protected resources on the resource server.
4. The resource server provides access to the protected resource.

WebSphere Education

Configure OAuth Client, 2 of 2

- The Redirect URI defines the set of redirection URIs to exchange tokens for an authorization code.
- By default, the resource owner is the user name from the extracted identity.
- The Access Token Lifetime sets the lifetime for the access token in seconds.

Generate Client Secret ☒

Redirect URI

Customized Scope Check ☐

Scope

Custom Resource Owner Handling ☐

Default Token State Lifetime

Authorization Grant Lifetime

Access Token Lifetime

Shared Secret

Authorization Process

Additional Client Process

Configure a OAuth client group, 2 of 2

The Generate Client Secret indicates whether to generate the client secret for the OAuth client. The specification refers to the client secret as client secret.

- * If you generate the passphrase, the passphrase becomes the client secret.
- * If you do not generate the passphrase, you must explicitly define the client secret.

The Redirect URI defines the set of redirection URIs to exchange tokens for an authorization code. Define as many redirection URIs as the authorization process uses.

Redirection URIs help to detect malicious clients and prevent phishing attacks. The authorization server must have the registered redirection URIs before the authorization server can validate the authorization request from the client.

The Customized Scope Check indicates how to check the scope for authorization grants and access tokens.

- * When checking the scope that use a custom style sheet, specify the location of the style sheet with the Scope Customized Process property. The style sheet must be in the local: or store: directory.
- * When checking the scope with a PCRE, specify the expression with the Scope property.

A custom scope check should be used in the following situations:

- * An authorization request where the OAuth client requests an authorization code.
- * An access request where the OAuth client can request an access token.
- * A resource request where the OAuth client requests a resource.

The Scope specifies the PCRE to check the scope. The minimum length of the expression is one character. The maximum length of the expression is 1023 characters.

The Scope Customized Process specifies the location of the style sheet for a custom scope check. The style sheet must be in the local: or store: directory. The style sheet validates and sets the scope to check.

The Custom Resource Owner Handling indicates whether to use a style sheet to extract information about the resource owner. When extracting that use a custom style sheet, use the Resource Owner Process property to specify the location of the style sheet. The style sheet must be in the local: or store: directory.

By default, the resource owner is the user name from the extracted identity. For custom handling, you must provide a style sheet that overrides information about the resource owner.

- * For AAA identity extraction, the extraction method can be basic authentication or forms-based login.

- * For custom handling, the style sheet overrides data about the resource owner with information from authentication.

The custom handling should be used in the following situations:

- * When presenting the authorization form to the resource owner
- * When issuing a code for an authorization code grant type
- * When issuing an access token

The Resource Owner Process specifies the location of the style sheet to extract information about the resource owner. The file must be in the local: or store: directory.

The DataPower State Lifetime sets the operational duration in seconds for the local authorization page. Enter a value in the range 1 - 6000. The default value is 1200.

If the user does not submit the request before the duration elapses, the authorization request from the OAuth client is rejected. The location of the style sheet that defines the local authorization page and the error handling is set with the Authorization Form property.

The Authorization Grant Lifetime sets the lifetime for an authorization code in seconds. Enter a value in the range 1 - 600. The default value is 300.

An authorization code is the intermediary result of a successful authorization. The client uses authorization codes to obtain the access token. Instead of sending tokens

to a client, clients receives authorization codes on their redirection URI. Each supported redirection URI for the client is defined with the Redirect URI property. The Access Token Lifetime sets the lifetime for the access token in seconds. Enter a value in the range 1 - 2678400. The default value is 3600.

The Shared Secret assigns the shared secret key to protect tokens that use the OAuth protocol.

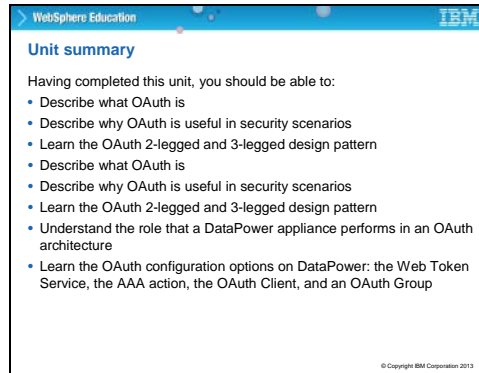
The Authorization Form specifies the location of the style sheet that submits the authorization request from the resource owner and handles errors. The file must be in the local: or store: directory. You can use the OAuth-Generate-HTML.xsl style sheet in the store: directory or copy this file to the local: directory and modify as needed.

The style sheet must be stored on the appliance in the local: or store: directory. The HTML authorization form remains operational for the duration that is defined with the DataPower State Lifetime property. If the user does not submit the request before the duration elapses, the authorization from the OAuth client is rejected.

The Additional OAuth Process specifies the location of the style sheet to process after generating a code, after generating an access token, or after generating an access token but before sending it to the resource server. The style sheet must be in the local: or store: directory.

Additional OAuth processing can be used in the following situations.

- * An authorization request after successfully generating a code for an authorization code grant with the `authorization_request` operation. Processing returns a node set. This information becomes part of the query string and is returned to the OAuth client.
- * An access request after successfully generating an access token with the `access_request` operation. Processing returns a node set. This information becomes part of the JSON object that contains the access token.
- * A resource request after successfully verifying an access token but before sending the request to the resource server with the `resource_request` operation.



WebSphere Education

Unit summary

Having completed this unit, you should be able to:

- Describe what OAuth is
- Describe why OAuth is useful in security scenarios
- Learn the OAuth 2-legged and 3-legged design pattern
- Describe what OAuth is
- Describe why OAuth is useful in security scenarios
- Learn the OAuth 2-legged and 3-legged design pattern
- Understand the role that a DataPower appliance performs in an OAuth architecture
- Learn the OAuth configuration options on DataPower: the Web Token Service, the AAA action, the OAuth Client, and an OAuth Group

© Copyright IBM Corporation 2013

In summary:

This unit described what OAuth is and why OAuth is useful in security scenarios. The unit covered the OAuth 2-legged and 3-legged security pattern. You can now understand the role that a DataPower appliance performs in an OAuth architecture. Finally, you learned the configurations on DataPower when using OAuth, which included the Web Token Service, the AAA action, the OAuth Client, and the OAuth Group.

Slide 16

WebSphere Education

IBM

Checkpoint

1. True or False. With OAuth, a resource owner allows third-party access to the resource **without** sharing their credentials.
2. True or False: 3 legged OAuth is the traffic/data pattern that OAuth is designed to solve.
3. DataPower implementation of OAuth includes (select 3):
 - a. Web Token Service
 - b. 1-legged authentication
 - c. AAA action
 - d. SSL
 - e. OAuth Client / OAuth Client Group
4. True or False: OAuth configuration on DataPower does not allow for the use of custom style sheets.

© Copyright IBM Corporation 2013

Slide 17

WebSphere Education

IBM

Checkpoint answers

1. **True.** With OAuth, a resource owner allows third-party access to the resource **without** sharing their credentials.
2. **True.** 3 legged OAuth is the traffic/data pattern that OAuth is designed to solve.
3. DataPower implementation of OAuth includes (select 3):
 - a. **Web Token Service**
 - b. 1-legged authentication
 - c. **AAA action**
 - d. SSL
 - e. **OAuth Client / OAuth Client Group**
4. **False:** OAuth configuration on DataPower **does** allow for the use of custom style sheets.

© Copyright IBM Corporation 2013