



# **FORUM SENTRY™ VERSION 9**

## **OAuth IdP SERVICE POLICY GUIDE**

**Legal Marks**

No portion of this document may be reproduced or copied in any form, or by any means – graphic, electronic, or mechanical, including photocopying, taping, recording, or information retrieval system – without expressed permission from Forum Systems, Inc.

FORUMOS™ Firmware, Forum Systems XMLSec™ WebAdmin, Forum Systems XML Security Appliance™, Forum Sentry™, Forum Presidio™, Forum XWall™, Forum Sentry™ Web Services Gateway, Forum Presidio™ OpenPGP Gateway, Forum FIA Gateway™, Forum XWall Type-PCI™, Forum XWall® Web Services Firewall and Forum XRay™ are trademarks and registered trademarks of Forum Systems, Inc.

All other products are trademarks or registered trademarks of their respective companies.

Copyright © 2002-2024 Forum Systems, Inc. – All Rights Reserved.

Forum Sentry™ Version 9 OAuth IdP Service Policy Guide, published May 2024.

D-ASF-SE-048469

## Contents

INTRODUCTION TO THE OAUTH IDP SERVER POLICY GUIDE .....	4
<i>Audience for the OAuth IdP Service Policy Guide .....</i>	<i>4</i>
<i>Conventions Used .....</i>	<i>4</i>
<i>Assumptions .....</i>	<i>4</i>
OAUTH IDP SERVER POLICIES.....	5
<i>Client Types .....</i>	<i>5</i>
Confidential.....	5
Public.....	6
<i>Grant Types .....</i>	<i>6</i>
Authorization code.....	6
Password (Resource Owner Password Credentials) .....	6
Implicit.....	6
Client Credentials .....	7
<i>OAuth Token Types .....</i>	<i>7</i>
Access Tokens .....	7
Refresh Tokens .....	7
<i>Policy Attributes to Dynamically Specify Timeouts and Redirect URI .....</i>	<i>7</i>
Authenticating Using a Database .....	8
ACL for Client Authentication .....	8
<i>OAuth IdP Service Policy Detail Terms .....</i>	<i>10</i>
OAuth Tab .....	10
Virtual Directory Tab.....	12
Settings Tab .....	12
IDP Rules Tab .....	13
Logging Tab.....	14
DATABASE .....	15
<i>Token Persistence .....</i>	<i>15</i>
<i>Authentication via a Database using Custom Auth Policy.....</i>	<i>15</i>
<i>Mapping To and From OAuth Attributes .....</i>	<i>16</i>
STEP-BY-STEP POLICY CONFIGURATION.....	17
<i>Example - Resource Owner Password Credentials Grant .....</i>	<i>17</i>
<i>Example – Authorization Code Grant Type.....</i>	<i>27</i>

# INTRODUCTION TO THE OAUTH IDP SERVER POLICY GUIDE

## Audience for the OAuth IdP Service Policy Guide

The *Forum Systems Sentry™ Version 9 OAuth IdP Server Policy Guide* is for scenarios where Forum Sentry will act as the Identity Provider (IdP) providing an OAuth Authorization Server. The Forum Sentry OAuth 2.0 authorization framework enables a third-party application to obtain access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the API service, or by allowing the third-party application to obtain access on its own behalf.

## Conventions Used

A red asterisk ( \* ) aligned with a field term means that this field is required. In this and other documentation, the Web Administration Interface is referred to as the WebAdmin and the Forum XML Security Appliance™ is referred to as the 'device', 'product' or 'system'.

In this document, all data or commands that must be entered or selected are displayed in boldface. Example:

User name:     **johnsmith**  
Password:     **\*\*\*\*\***

UI screens which display a STATUS column represent the following states:

- Green status light = enabled policy.
- Yellow status light = a required functional element of this policy is disabled.
- Red status light = disabled policy.

## Assumptions

This document assumes that the reader will review the appropriate chapter before performing the operations listed in this document. This document also assumes that the reader is familiar with OAuth. If you are not familiar, we recommend you read The OAuth 2.0 Authorization Framework <http://tools.ietf.org/html/rfc6749>.

## OAUTH IdP SERVER POLICIES

Forum Sentry provides an integrated OAuth authorization server which provides a comprehensive set of OAuth identity provider capabilities for grant types authorization code, implicit, resource owner password credentials, and client credentials to enable integration with any OAuth client or API infrastructure.

FORUMSENTRY > API SECURITY GATEWAY FORUMSYSTEMS

GENERAL  
Forum Systems  
Getting Started  
Help

DIAGNOSTICS

GATEWAY  
Network Policies  
Network Policies  
Proxy Policies  
Cloud Policies  
Cache Policies  
WSDL Policies  
WSDL Libraries  
WSDL Policies  
Content Policies  
XML Policies  
REST Policies  
JSON Policies  
HTML Policies  
STS Policies  
**OAuth Policies**  
Tests  
Task Policies  
Task List Groups  
Task Lists  
Redirect Policies  
Redirect Policies  
Request Filters  
Request Filters

RESOURCES

OAUTH POLICIES > NEW OAUTH POLICY

NEW OAUTH POLICY

Name\*: OAUTH\_Policy

Description:

Labels:

Client Authentication Mode:

☒ Use these client credentials

Client Id\*: client1

Client Secret\*: oKZ3x7hXTRJDqd9TDGxvw==

Redirect URI\*: http://www.forumsys.com/sentry

☐ Use this ACL to authenticate client

Client Authentication ACL: [Allow All]

Client Type:

☒ Confidential (Web Application) ☐ Public (Native or JavaScript)

Grant types\*: ☒ Authorization code ☐ Implicit ☐ Password ☐ Client credentials

Scope\*: default

Default Access Token Lifetime (secs)\*: 7200

☐ Default Refresh Token Lifetime (days): 60

☐ Reuse refresh token

☐ Enable persistent token storage

ACL Policy (password grant): [Allow All]

Request Task List Group: [None]

Response Task List Group: [None]

Next

### Client Types

Client types supported are

- Confidential
- Public

### Confidential

Clients capable of maintaining the confidentiality of their credentials (e.g., client implemented on a secure server with restricted access to the client credentials), or capable of secure client authentication using other means.

The Confidential option requires that both a Client Id and a Client Secret be provided when generating an access token.

## Public

Clients incapable of maintaining the confidentiality of their credentials (e.g., clients executing on the device used by the resource owner, such as an installed native application or a web browser-based application), and incapable of secure client authentication via any other means.

The Public option requires only the Client Id to be provided when generating an access token.

## Grant Types

Forum Sentry OAuth Policies support a comprehensive set of grant types per the OAuth specification. All OAuth grant types are fully supported as described below.

### Authorization code

The authorization code grant type is the most common OAuth2.0 flow, and is often referred to as 3-legged OAuth.

The authorization code is obtained by using an authorization server as an intermediary between the client and resource owner. Instead of requesting authorization directly from the resource owner, the client directs the resource owner to an authorization server which in turn directs the resource owner back to the client with the authorization code.

Before directing the resource owner back to the client with the authorization code, the authorization server authenticates the resource owner and obtains authorization. Because the resource owner only authenticates with the authorization server, the resource owner's credentials are never shared with the client.

The authorization code provides a few important security benefits, such as the ability to authenticate the client, as well as the transmission of the access token directly to the client without passing it through the resource owner's user-agent and potentially exposing it to others, including the resource owner

### Password (Resource Owner Password Credentials)

The resource owner password credentials (i.e., username and password) can be used directly as an authorization grant to obtain an access token. The credentials should only be used when there is a high degree of trust between the resource owner and the client (e.g., the client is part of the device operating system or a highly privileged application), and when other authorization grant types are not available (such as an authorization code).

Even though this grant type requires direct client access to the resource owner credentials, the resource owner credentials are used for a single request and are exchanged for an access token. This grant type can eliminate the need for the client to store the resource owner credentials for future use, by exchanging the credentials with a long-lived access token or refresh token.

### Implicit

The implicit grant is a simplified authorization code flow optimized for clients implemented in a browser using a scripting language such as JavaScript. In the implicit flow, instead of issuing the client an authorization code, the client is issued an access token directly (as the result of the resource owner authorization). The grant type is implicit, as no intermediate credentials (such as an authorization code) are issued (and later used to obtain an access token).

When issuing an access token during the implicit grant flow, the authorization server does not authenticate the client. In some cases, the client identity can be verified via the redirection URI used to

deliver the access token to the client. The access token may be exposed to the resource owner or other applications with access to the resource owner's user-agent.

## **Client Credentials**

The client credentials (or other forms of client authentication) can be used as an authorization grant when the authorization scope is limited to the protected resources under the control of the client, or to protected resources previously arranged with the authorization server. Client credentials are used as an authorization grant typically when the client is acting on its own behalf (the client is also the resource owner) or is requesting access to protected resources based on an authorization previously arranged with the authorization server.

## **OAuth Token Types**

### **Access Tokens**

Access tokens are credentials used to access protected resources. An access token is a string representing an authorization issued to the client. The string is usually opaque to the client. Tokens represent specific scopes and durations of access, granted by the resource owner, and enforced by the resource server and authorization server.

The token may denote an identifier used to retrieve the authorization information or may self-contain the authorization information in a verifiable manner (i.e., a token string consisting of some data and a signature).

The access token provides an abstraction layer, replacing different authorization constructs (e.g., username and password) with a single token understood by the resource server. This abstraction enables issuing access tokens more restrictive than the authorization grant used to obtain them, as well as removing the resource server's need to understand a wide range of authentication methods.

### **Refresh Tokens**

Refresh tokens are credentials used to obtain access tokens. Refresh tokens are issued to the client by the authorization server and are used to obtain a new access token when the current access token becomes invalid or expires, or to obtain additional access tokens with identical or narrower scope (access tokens may have a shorter lifetime and fewer permissions than authorized by the resource owner). Issuing a refresh token is optional at the discretion of the authorization server. If the authorization server issues a refresh token, it is included when issuing an access token.

A refresh token is a string representing the authorization granted to the client by the resource owner. The string is usually opaque to the client. The token denotes an identifier used to retrieve the authorization information. Unlike access tokens, refresh tokens are intended for use only with authorization servers and are never sent to resource servers.

## **Policy Attributes to Dynamically Specify Timeouts and Redirect URI**

to dynamically set the redirect-uri, access-token-seconds, and the refresh-token-hours based on mapping to those values via a mapping task with the Target type of "User Attribute".

## Authenticating Using a Database

The ACL on the OAuth Policy enables the ability to define how to authenticate the inbound ClientID and ClientSecret. To authenticate via a database, use the Forum Sentry Custom Authenticate Adapter found under Access->Custom area. Setting up this identity adapter to an ACL that is associated with the OAuth policy is how to tie the authentication event of the OAuth policy with custom logic to determine how and where to authenticate the credentials against, such as a database. The database authentication requires creation of db schema that holds the client id and the client secret. App-based timeouts can also be specified by having fields handle storing the access token timeout, and the refresh token timeout. Using the Query Database task, the attribute information from the database can be mapped to the dynamic OAuth specially named attributes to override the OAuth Policy default values.

The screenshot displays two web-based configuration screens. The top screen, titled 'CUSTOM AUTHENTICATION > CUSTOM AUTH POLICY CONFIGURATION', shows the 'CUSTOM AUTH POLICY' settings. A red oval highlights the 'Request Task List Group' field, which is set to 'Task List Groups'. A red arrow points from this field to the bottom screen. The bottom screen, titled 'TASK LISTS > TASK LIST: NEW TASK LIST2 > TASK: QUERY DATABASE', shows the 'QUERY DATABASE' task configuration. The 'Task Name' is 'Query Database', and the 'Data Source' is 'Database\_Policy'. The 'SQL' field is empty. The 'Output' is set to 'XML'. The 'Result Set Attribute Key Prefixes' field is empty. The 'Legacy Output Mode' is unchecked. The 'SQL Values (Click To Remove)' section shows a table with columns: SQL TYPE, SOURCE TYPE, SOURCE NAME, and OUTPUT NAME. The table is currently empty.

**CUSTOM AUTHENTICATION > CUSTOM AUTH POLICY CONFIGURATION**

[Always Show Advanced](#)

**CUSTOM AUTH POLICY**

Policy Name\*:

Enable privileged access: ☐ Yes ☒ No

Restrict Menus: ☐

Role policy:  [Edit](#)

Request Task List Group:

Cache timeout (in minutes):

Cookie Name:

[Apply](#) [Save](#)

**TASK LISTS > TASK LIST: NEW TASK LIST2 > TASK: QUERY DATABASE**

**QUERY DATABASE**

Task Type:

Task Name\*:

On Error: ☒ Log & Halt Processing ☐ Log & Continue

SQL:

Data Source:  [Edit](#)

Output:

Result Set Attribute Key Prefixes:

Legacy Output Mode: ☐

SQL Values (Click To Remove)

SQL TYPE	SOURCE TYPE	SOURCE NAME	OUTPUT NAME
----------	-------------	-------------	-------------

[Apply](#) [Save](#)

Note that when mapping the preset attributes, these have to be mapped as Identity Attributes.

## ACL for Client Authentication

When using an ACL to authenticate, the target authentication method can be a REST or Custom Sentry policy which provides Task List Group association for customized processing of the authentication. Common usage of this would be to have a database schema which has the ClientID, ClientSecret, and the Access and Token refresh times specified. A Sentry Custom Authentication adapter would then use a Task List Group that maps the inbound ClientID and ClientSecret to a database table and authenticate the user while in turn using the additional user settings that can be specified as Identity Attributes.



## OAuth IdP Server Policy Screen Terms

The OAuth screen includes the following terms and definitions:

TERM	DEFINITION
Name	The name of the OAuth Policy
Virtual Directory	One of the 3 directories that can be called, depending on the grant type <ul style="list-style-type: none"><li>• <b>authorize</b> – the OAuth authorization endpoint used to authenticate the resource owner and issue a code for authorization code grant or an access token for implicit grant.</li><li>• <b>token</b> – the OAuth token endpoint used to issue access tokens for grant types other than implicit.</li><li>• <b>attributes</b> – used to validate access tokens and obtain user attributes and information associated with an issued token.</li></ul>
Status	The Status column represent the following states: <ul style="list-style-type: none"><li>• Green status light = enabled policy.</li><li>• Yellow status light = a required functional element of this policy is disabled.</li><li>• Red status light = disabled policy.</li></ul>
URI	The full URI of the 3 virtual directories exposed for communication, based on the associated Listener Policy.
Listener ACL	The access control list, if used, to control access to the policy.
Directory ACL	The access control list to access the virtual directory. Leave setting as “[Allow All]” to not enforce an ACL.
IDP Group	The Intrusion Detection and Prevention policy to be active for the OAuth Policy

## OAuth IdP Service Policy Detail Terms

On the details screen is where the configuration aspects of the OAuth IdP Service policy across the listed Tabs. Click on the “Always Show Advanced” text link at the top of the page to show the OAuth special attribute names on the bottom of the policy inputs.

### OAuth Tab

OAUTH POLICIES > OAUTH POLICY

OAUTH POLICY

Policy Name: OAUTH\_Policy

OAuth Virtual Directories Settings IDP Rules Logging

OAUTH POLICY

TERM	DEFINITION
Client Type	<p>Confidential (Web Application)</p> <ul style="list-style-type: none"><li>• Clients capable of maintaining the confidentiality of their credentials (e.g., client implemented on a secure server with restricted access to the client credentials), or capable of secure client authentication using other means.</li></ul> <p>Public (Native or Javascript)</p> <ul style="list-style-type: none"><li>• Clients incapable of maintaining the confidentiality of their credentials (e.g., clients executing on the device used by the resource owner, such as an installed native application or a web browser-based application), and incapable of secure client authentication via any other means</li></ul>
Client Id	<p>The client identifier.</p> <ul style="list-style-type: none"><li>• The authorization server issues the registered client a client identifier, a unique string representing the registration information provided by the client. The client identifier is not a secret; it is exposed to the resource owner and per specification should not be used alone for client authentication. The client identifier is unique to the authorization server.</li></ul>
Client Secret	<p>The client secret.</p>
Grant types	<p>Authorization code</p> <ul style="list-style-type: none"><li>• The Authorization Code grant type is the most common OAuth2.0 flow. It implements 3-legged OAuth and involves the resource owner</li></ul>

granting the client an authorization code that can be exchanged for an access token.

#### Password

- A resource owner's username and password are submitted as part of the request, and an access token is issued upon successful authentication.

#### Implicit

- This is similar to the authorization code grant type, but rather than an authorization code being returned from the authorization request, a token is returned to the client. This is most common for client-side devices (i.e. mobile) where the client credentials cannot be stored securely.

#### Client Credentials

- The client uses its credentials to retrieve an access token directly, which allows the client to control access to resources.

Redirect URI	For grant types that include a redirect, this setting determines where to issue the HTTP redirect upon successful authorization.
Scope	Optional usage of an opaque string defining the scope parameter of the access request. The value of the scope parameter is expressed as a single string, or a list of space-delimited, case-sensitive strings.
Access Token Lifetime (secs)	The duration validity of an issued access token.
Refresh Token Lifetime (days)	Optional. If using refresh tokens, the validity time of the issued refresh token.  Reuse refresh token - when checked, the same refresh token can be reused repeatedly to acquire new access tokens. If not checked, a single-use refresh token is issued and a new single-use refresh token is issued each time a refresh token is used.
Enable persistent token storage	Allows storage of tokens in a database table that can be used for persistence and sharing of session and token information across multiple Sentry instances.
ACL Policy	Used with password grant to authorize the resource owner. Leverages an access control list to build any credential paradigm to any supported Forum Sentry identity adapter, including LDAP, AD, SOAP API, REST API, and Database.
Request Task List Group	Applies a task workflow to the inbound OAuth requests to the policy.
Response Task List Group	Applies a task workflow to response messages prior returning the response.
Redirect URI Attribute	redirect_uri – the name of the Identity attribute to override via Task mapping to map the redirect URI. This value would be set via the mapping Tasks during the processing of the ACL set on the OAuth policy
Access Token Lifetime Attribute	access_token_seconds – the name of the Identity attribute to override the Access Token duration. This value would be set via the mapping Tasks during the processing of the ACL set on the OAuth policy
Refresh Token Lifetime Attribute	refresh_token_hours – the name of the Identity attribute to override the Refresh Token duration. This value would be set via the mapping Tasks during the processing of the ACL set on the OAuth policy

## Virtual Directory Tab

**OAUTH POLICIES > OAUTH POLICY**

---

**OAUTH POLICY**

Policy Name: OAUTH\_Policy

---

**OAuth** Virtual Directories Settings IDP Rules Logging

---

Virtual Directories > Virtual Directory: authorize

---

**VIRTUAL DIRECTORY**

---

TERM	DEFINITION
Name	Name of the virtual directory.
Description	Description of the virtual directory to be used to add comments.
Listener Policy	Association with the policy to be used for the protocol communication to the OAuth IdP policy.
Virtual Host	Enables the use of the Host header to designate the policy beyond IP and virtual directory.
Virtual Path	The path segment of the URI.
Filter Expression	Regular expression to determine what can follow the defined virtual path.
IP ACL Policy	Association with an IP access control list for TCP Layer 3 filtering.
ACL Policy	Ability to associate an access control list for the specific virtual directory.
Password Authentication	Enables the ability to override the listener policy settings and define authentication requirements for the inbound HTTP(S) request.
Redirect Policy	Used to determine where to redirect in the case of one of these events: (Authentication Success, Authentication Failure, No Credentials, On Error)
Error Template	Can specify and override the error template affiliated with the listener policy to specify a custom format and structure for error responses.

## Settings Tab

**OAUTH POLICIES > OAUTH POLICY**

---

**OAUTH POLICY**

Policy Name: OAUTH\_Policy

---

**OAuth** Virtual Directories **Settings** IDP Rules Logging

---

TERM	DEFINITION
Policy Name	Name of the policy
Policy Description	Description of the policy to be used to add comments.
Enable Session Cookies	<p>This is used to enable cookie sessions managed by Sentry. Checking this setting will enable Sentry to issue Set-Cookie back to the client when an authentication event is successful. The cookie would then be subsequently enforced via another Sentry policy that extracts the session cookie, and then ensures it is still valid.</p> <p>Cookie Settings include:</p> <p>Cookie Name: Name of the cookie.</p> <p>Cookie Path: URI Path for the session cookie. Use / for root, or leave empty to use the inbound URI.</p> <p>Cookie Domain: (optional) The domain to issue the session cookie.</p> <p>Session Timeout (mins): The overall validity period of the issued cookie.</p> <p>Session Idle Timeout (mins): The idle timeout window that represents the period of time within which the cookie needs to be used before it's considered stale and thus invalid.</p>
Enable persistent sessions	If cookie sessions are enabled, this setting allows sessions to be persistent in a database, and can also be shared across multiple deployments of Sentry.
Default Character Encoding	The default encoding to use if the inbound request doesn't specify one

## IDP Rules Tab

The screenshot shows the Sentry web interface for configuring an OAuth Policy. The breadcrumb navigation at the top reads "OAUTH POLICIES > OAUTH POLICY". Below this, the "OAUTH POLICY" header is followed by a form where the "Policy Name" is set to "OAUTH\_Policy". A horizontal tab bar contains five tabs: "OAuth", "Virtual Directories", "Settings", "IDP Rules" (which is the active tab), and "Logging". Under the "IDP Rules" tab, there is a section titled "IDP GROUP". This section contains a label "IDP Group:" followed by a dropdown menu currently showing "Default HTML Policy Group" and an "Edit" link.

TERM	DEFINITION
IDP Group	The Intrusion Detection and Prevention policy set to associate with this policy

## Logging Tab

**OAUTH POLICIES > OAUTH POLICY**

---

**OAUTH POLICY**

Policy Name: OAUTH\_Policy

---

OAuth Virtual Directories Settings IDP Rules **Logging**

---

**LOGGING SETTINGS**

---

TERM	DEFINITION
Enable policy level logging settings	Allows overriding the system level logging settings.
Override log level for the following codes	This allows forced logging for the specific logging code, or logging code groups.
Filter codes	Enables defining logging codes, or code groups to not log.
Pattern Match Policy	Allows association with a pattern match policy to enable regular expression filtering or replacement of logged message content.

## DATABASE

Forum Sentry OAuth IdP features enable persisting the state of the issued tokens into a database. A single instance of Forum Sentry will store token information in local memory cache for subsequent access control checks and mappings. Commonly, Forum Sentry is deployed in high availability pairs, which requires the Sentry instances to share state with each other as pertains to issued tokens and authenticated sessions. This allows all Sentry instances to share information as well as allow token information to persist through restarts of the Sentry instances themselves.

### Token Persistence

To persist OAuth tokens into a database table, enable one of the supported database policy variants in Sentry (MySQL, SQL Server, Oracle, DB2) and apply the database schema to create the required tables. The database schema is accessible from the Logging->Data Sources screen by clicking the hyperlink on the applicable database type.

**DATA SOURCES > DATA SOURCE**

**CONFIGURATION**

Click on hyper link for data source schema

Type: ☒ [Oracle](#) ☐ [MySQL](#) ☐ [DB2](#) ☐ [SQL Server](#) ☐ [Oracle RAC](#)

Name\*:

Enable SSL: ☐

SSL initiation policy:

Server\*:

Port\*:

Database\*:

Schema:

User\*:

Password\*:

Connect Descriptor:

Max Connections\*:

Synchronous: ☒

[Test](#) [Apply](#) [Save](#)

To enable the use and storage of information in the database, go to the OAuth Policy and enable the “Enable persistent token storage” setting by selecting the applicable data source policy.

### Authentication via a Database using Custom Auth Policy

You can choose to use a database where the Client ID, Client Secret and timeout values can be stored and retrieved, thus allowing multiple ClientID and Client Secret values be associated with an OAuth Policy. This feature is enabled via the selection of the ACL pulldown

To use a database table, you will need to create a Custom Auth policy and associate a Task List Group. In the Task List Group, you will create a task list that uses the Query Database task to search for, or update values in the database table that you have created for handling these values. Use of a database table and schema are at the discussion of the end-customer, and is not a schema in the Forum Sentry provided tables. In the schema, we recommend at the least to include:

- UniqueID (index)
- ClientID
- ClientSecret
- AccessTokenTime

Note that these names are also arbitrary, the important aspect is to map these values from the database to the applicable Identity attribute values specified on the OAuth policy (redirect\_usi, access\_token\_seconds, refresh\_token\_hours). This will force the OAuth policy to use the Identity attribute settings rather than the default policy settings.

## Mapping To and From Oauth Attributes

Authorization code is the recommended grant type, in which the client\_id would be passed as a query/form parameter to the authorize endpoint. Client credentials are not used for the authorize endpoint, so the use of a query/form parameter for the client\_id is acceptable in the case of authorization code grant (and implicit code grant). The Sentry map task source is Query Parameter where you would obtain the inbound ClientID in order to look up into a database table.

When using Custom Auth, the client\_id is also available in an attribute as noted on the OAuth Policy for which the Sentry map task source is User Attribute.

For password grant, the use of the http basic authorization header is recommended instead of query/form parameters.

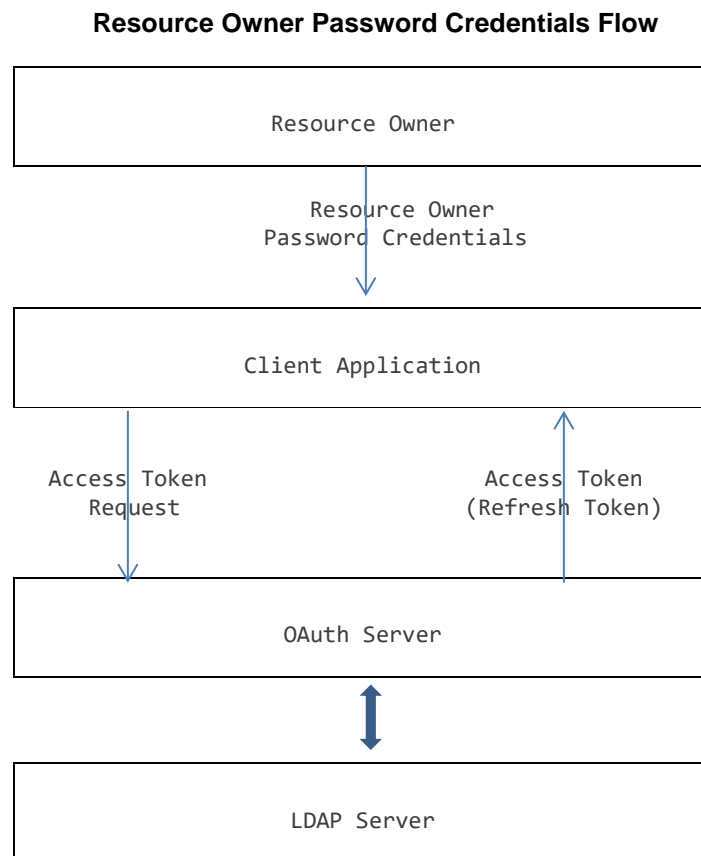


## STEP-BY-STEP POLICY CONFIGURATION

The following sections provide step by step screenshots and instructions for setting up some use-cases as to how to use the Forum Sentry OAuth IdP Service capabilities.

### Example - Resource Owner Password Credentials Grant

Mostly referred to as Password Grant Type, it is often used when there is a trust relationship between the Resource Owner and the Client Application. In this case the user does not have to be redirected to the authorization server; rather it supplies the credentials (username/password) to the client application. The client application uses the credentials to build the request to the OAuth server. The request is made up of the grant type, username, password and client ID. In cases where the OAuth server requires confidential client types, the request will have to also include the client secret. The client ID and client secret are supplied to the client application upon registry of the client. Upon authentication the OAuth server responds with an access token and optionally a refresh token.



#### The steps below explain the flow:

1. The resource owner provides the client with its username and password.
2. The client requests an access token from the authorization server's token endpoint by including the credentials received from the resource owner.

3. The OAuth server validates the client ID and client secret then authenticates the client against the LDAP Server. Upon success it responds with an access token.

If the access token validation fails the OAuth server returns an error response.

**The set up will use:**

- A. A public LDAP test server provided by Forum Sentry: ([Online LDAP Test Server](#))
- B. Forum Sentry as the OAuth Server
- C. [SOAPSonar](#) as the Client Application
- D. An FSG file containing the setup as well a [SOAPSonar](#) Project file for users familiar with Forum Sentry and [SOAPSonar](#)

**A. Setting up an Identity and Access Management (IAM) :**

*Note: For our example below we will setup LDAP (You can also use local accounts created in Sentry).*

1. Log in to the web admin and go to **ACCESS→User Policies→LDAP** and click **New**
2. LDAP setup (You can use the [Online LDAP Test Server](#) provided by Forum Systems):

The screenshot shows the 'LDAP POLICIES > LDAP POLICY' configuration page. At the top, it says 'Successfully authenticated with LDAP server' and 'Always Show Advanced'. The 'LDAP SERVER' section contains the following fields: Policy Name\* (LDAP\_FS\_ADMIN), Enable privileged access (radio buttons for Yes and No, with No selected), Restrict Menus (checkbox), Role policy (dropdown), Server\* (ldap.forumsys.com), Port\* (389), Use SSL to connect (checkbox), Authentication type (Simple dropdown), User\* (cn=read-only-admin,dc=example,dc=com), Password\* (masked with dots), Cache timeout (in minutes) (30), and Read Timeout (in minutes) (2). The 'MAPPINGS' section contains: Toggle Mappings (LDAP v3 dropdown), Root DN\* (ou=mathematicians,dc=example,dc=com), User/group context (radio buttons for List of users and Group containing users, with Group containing users selected), and Optimized group search (checkbox). At the bottom right, there are buttons for Test, Apply, and Save.

3. Add a User ACL and associate the LDAP Policy to it:

USER ACL MANAGEMENT > USER ACL DETAILS

---

**USER ACL DETAILS**

ACL Name: LDAP\_FS\_ACL

---

#	USER GROUP	EXECUTE
1	LDAP-LDAP_FS_ADMIN	<input checked="" type="checkbox"/>
2	LDAP_FS_Group	<input type="checkbox"/>

**Save**

## B. Forum Sentry as the OAuth Server:

### 1. Add a Listener Policy:

- Click on **GATEWAY**→**Network Policies**→**Network Policies**.
- Click **New** to a Listener policy
- Click next to step through filling in the required information (see image below for an example)

*Note: You might want to avoid using port 80 if you have other listeners on this port. Here we will use port 8080*

NETWORK POLICIES > HTTP LISTENER POLICY

---

**POLICY NAME**

Policy Name\*:

**Next**

---

**POLICY SELECTIONS**

<u>Policy Name:</u>	OAuthListenerPolicy
<u>IP ACL Policy:</u>	Unrestricted
<u>Inbound Protocol:</u>	HTTP (chunked)
<u>Listener:</u>	0.0.0.0:8080
<u>Password Authentication:</u>	
<u>Error Handling:</u>	Default Template

### 2. Create the OAuth Server Policy:

- Go to **GATEWAY**→**Content Policies**→**OAuth Policies**
- Click **New**
- Add an appropriate Name for you policy (see image below):

## OAuth Policies > NEW OAUTH POLICY

**NEW OAUTH POLICY**

Name\*:

Description:

Labels:

Client Authentication Mode: ☒ Use these client credentials

Client Id\*:

Client Secret\*:

Redirect URI\*:

☐ Use this ACL to authenticate client

Client Authentication ACL:

Client Type: ☒ Confidential (Web Application) ☐ Public (Native or JavaScript)

Grant types\*: ☒ Authorization code ☐ Implicit ☒ Password ☐ Client credentials

Scope\*:

Default Access Token Lifetime (secs)\*:

☐ Default Refresh Token Lifetime (days):

☐ Reuse refresh token

☐ Enable persistent token storage

Database\_Policy  [Edit](#)

ACL Policy (password grant):

Request Task List Group:

Response Task List Group:

[Next](#)

Note the following:

- Save the Client Id and Client Secret for use later by the client application (SOAPSonar)
- Client Type is set to **Public** (you can change this to **Confidential** which requires the use of the client secret as well as client id)
- Reuse refresh token is checked (This is to be able to request a new access token based on one received earlier)

- Click **Next** to go through the Virtual Directories' setup. This sets up the /authorize, /token and /attributes Virtual Directories used by the OAuth Server.
- Click **Save** and then go back to **OAuth Policies** to see the new policy:

**OAuth Policies**

Search Usage: type any text Filter Usage: type or select the label

No Labels

NAME	VIRTUAL DIRECTORY	STATUS	URI	LISTENER ACL	DIRECTORY ACL	IDP GROUP
<input type="checkbox"/> Grant_Type_Password_OAuth_Policy	authorize	<span style="color: green;">●</span>	<a href="http://127.0.0.1:8080/authorize">http://127.0.0.1:8080/authorize</a>	[Allow All]	[Allow All]	<a href="#">Default HTML Policy Group (7)</a>
	token	<span style="color: green;">●</span>	<a href="http://127.0.0.1:8080/token">http://127.0.0.1:8080/token</a>	[Allow All]	[Allow All]	<a href="#">Default HTML Policy Group (7)</a>
	attributes	<span style="color: green;">●</span>	<a href="http://127.0.0.1:8080/attributes">http://127.0.0.1:8080/attributes</a>	[Allow All]	[Allow All]	<a href="#">Default HTML Policy Group (7)</a>

[GDM Transfer](#) [GDM Export](#) [Delete](#) [New](#)

3. Add a Web Service to test using the Access Token:

- We have already created a listener policy above we can use
- Similarly create a Remote Policy to point to a web server (in this example we are using [www.forumsys.com](http://www.forumsys.com)):
  - Go to **GATEWAY→Network Policies→Network Policies**.
  - Click **New** to add a Remote Policy

NETWORK POLICIES > HTTP REMOTE POLICY

---

**POLICY NAME**

Policy Name\*:

---

**Next**

---

**POLICY SELECTIONS**

<u>Policy Name:</u>	OAuthRemotePolicy
<u>Outbound Protocol:</u>	HTTP
<u>Remote Server:</u>	www.forumsys.com:80
<u>TCP Timeouts:</u>	Connect: 10 Read: 600 Connection Limit: Unlimited
<u>Process Response:</u>	Off

- Add a **Content Policy**:
  - Go to **GATEWAY→Content Policies→HTML Policy**
  - Click **New**
  - Select the Listener and Remote Policies created above and set the Virtual Directory Path to /webservice

## HTML POLICIES > NEW HTML POLICY

### SET LISTENER POLICY

Please specify a listener policy for virtual directory: New Virtual Directory

- ☒ Select from existing listener policies

OAuthListenerPolicy (0.0.0.0:8080) [Edit](#)

- ☐ Create a new HTTP listener policy

Listener Policy Name\*: OAuthHTMLPolicy-Listener

Use Device IP: ☐

Listener IP\*: 169.254.84.66

Listener Port\*: 80

### SET VIRTUAL DIRECTORY PATH

Virtual Directory Path: /webservice

### SET REMOTE POLICY

Please specify a remote network policy

- ☐ Do not send to remote server  
☒ Select from existing remote policies

OAuthRemotePolicy (www.forumsys.com:80) [Edit](#)

- ☐ Create a new HTTP remote policy for this remote server

Remote Policy Name\*: OAuthHTMLPolicy-Remote

Remote Policy Host\*:

Remote Policy Port\*: 80

- ☐ Click **Finish**

#### 4. Create a Task to Validate the Access Token:

We will create a Task that will be set up on the /webservice service to first validate the Access Token before allowing access to the service.

- We first need to create a Remote Policy for use with the Task:
  - Remote server should point back to your server
  - Port should be the same one used above to point to /attributes
  - Check "Provide Basic Authentication Credentials"
  - Select "Propagate client's credentials"
  - Continue with the defaults

## NETWORK POLICIES > HTTP REMOTE POLICY

### POLICY NAME

Policy Name\*:

[Next](#)

### POLICY SELECTIONS

<u>Policy Name:</u>	HttpRemote_ForumOAuthServer
<u>Outbound Protocol:</u>	HTTP
<u>Remote Server:</u>	192.168.1.109:8080 (Basic Auth)
<u>Remote Authentication:</u>	Propagate client's credentials
<u>TCP Timeouts:</u>	Connect: 10 Read: 600 Connection Limit: Unlimited
<u>Process Response:</u>	Off

- Next go to **GATEWAY**→**Task Policies**→**Task Lists**
- Click **New** and name the Task (i.e. Validate OAuth Token) then click **Apply**
- Click **New** and select “User Identity & Access Control” then click **Next**
- Uncheck “Map identified user to a known user” then click **Next**
- Select “Validate OAuth token & establish identity” then click **Next**
- Select **Other** for the **Identity Provider**
- The image below shows the rest of the parameter (please match as many as possible)

## TASK LISTS > TASK LIST: VALIDATE OAUTH TOKEN > CONTROL

### TASK NAME

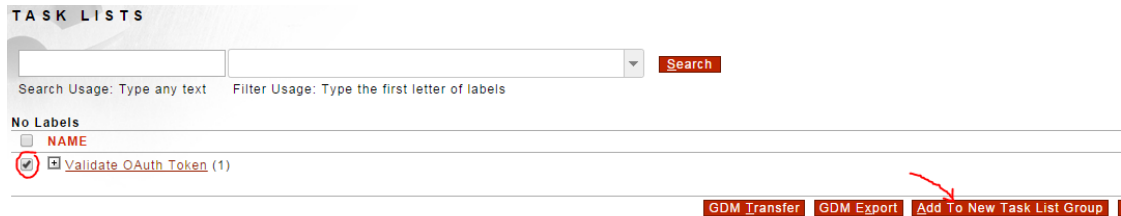
Task Name\*:

[Next](#)

### USER IDENTITY & ACCESS CONTROL

Task Type:	User Identity & Access Control
<u>Task Name:</u>	User Identity & Access Control
<u>ACL Policy:</u>	No user mapping
<u>User Identity Mechanism:</u>	Validate OAuth token & establish identity
<u>Identity Provider:</u>	Other
<u>Service Endpoint Policy:</u>	HttpRemote_ForumOAuthServer
<u>Service Endpoint Path:</u>	/attributes
<u>Request Task List Group Name:</u>	
<u>Response Task List Group Name:</u>	
<u>User Name Field:</u>	username
<u>Cache Timeout (in minutes):</u>	0
<u>Error Template:</u>	[From Policy]

- Create a new **Task List Group** and add the new “Validate OAuth Token” Task list to it or click on the Task List itself and then click “Add To New Task List Group” as seen below:

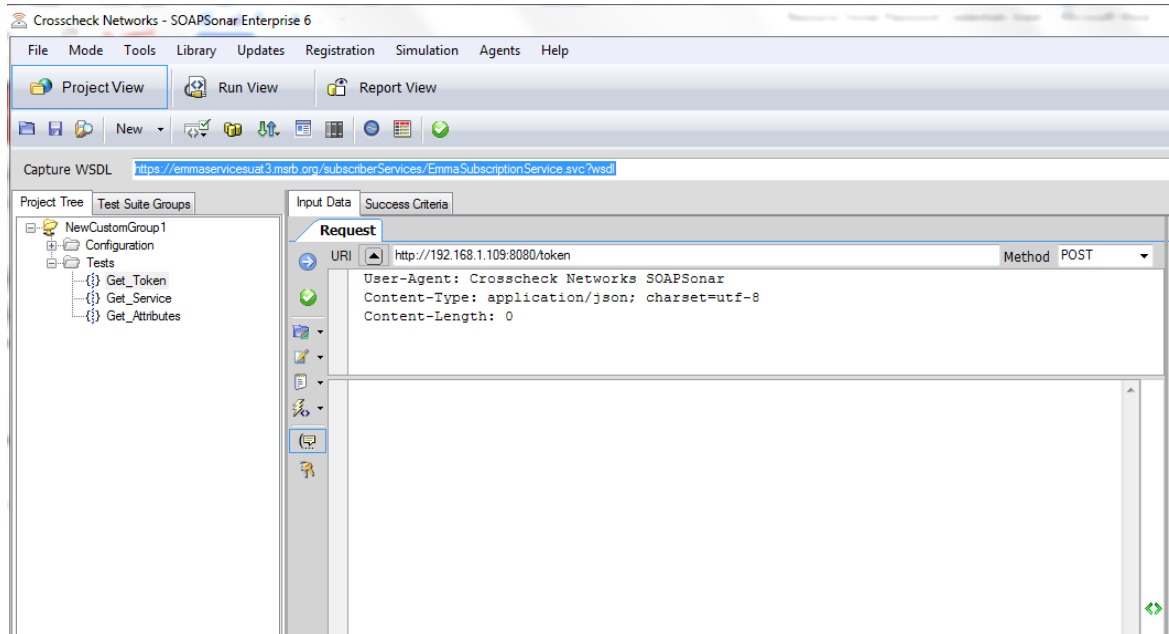


- Go back to you HTML Policy and click to access the Virtual Directory.
- Scroll to the bottom and select the “Validate OAuth Token” Task list Group for the “Request Task List Group”.

This concludes the Forum Sentry set up. Next we'll setup SOAPSonar to first request and receive a Access Token and then use this token to access our web service.

### C. Using SOAPSonar as the the Client Application:

- Launch [SOAPSonar](#), click on **Project View** and create a new **Test Group** by clicking on **File → New → Test Group**
- A test group is created. Right-click on **Tests** and select **New JSON Test**. Do this 3 time to create 3 new tests. Rename each so to have **Get\_Token**, **Get\_Service** and **Get\_Attributes** as show in the next image.



### Test 1:



- i. Test the Get\_Token to request a Access Token by following the steps below:
  - Click on Get\_Token
  - Set the URI to <http://oauth-server-ip-address:port/token>



- Set method to POST
- Click on the highlighted Headers icon, as show in the above image, to display the Headers section
- Change the Content-Type to: `application/x-www-form-urlencoded`
- Paste the following in the request body section:

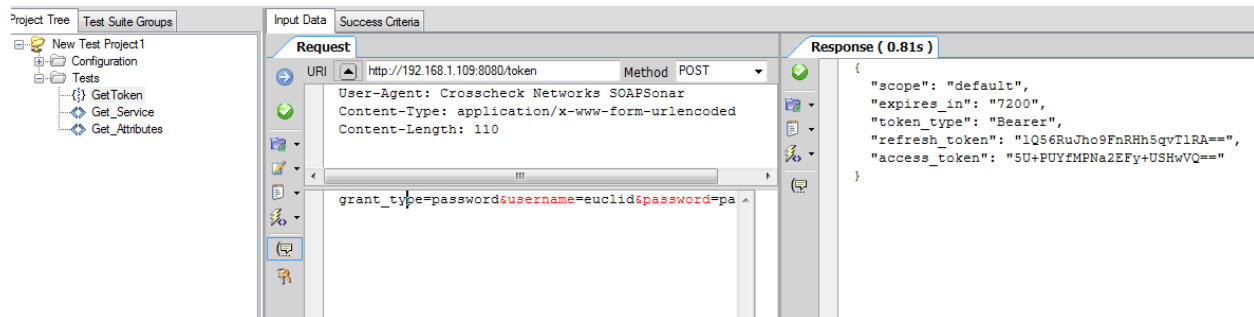
`grant_type=password&username=euclid&password=password&client_id=client1`

Note: Make sure that there are no space characters, stop the cut and paste at the end of "client1"

- Click the  to save then  to send the request.

### **Result:**

```
{
  "scope": "default",
  "expires_in": "7200",
  "token_type": "Bearer",
  "refresh_token": "lQ56RuJho9FnRHh5qvTlRA==",
  "access_token": "5U+PUYfMPNa2EFy+USHwVQ=="
}
```



### **Test 2:**

For this test we will use the access token obtained in above to get the attributes associated with the user "euclid":

- ii. Click on Get\_Attributes and use the following headers:  
*User-Agent: Crosscheck Networks SOAPSonar*  
*Content-Type: text/xml; charset=utf-8*  
*Content-Length: 0*  
*Authorization: Bearer 5U+PUYfMPNa2EFy+USHwVQ==*

Note: the Authorization header which must contain the worked Bearer and the token obtained as the result of the above test.

- Set the URI to <http://oauth-server-ip-address:port/attributes>
- Set method to POST
- Click the  to save then  to send the request.

### **Result:**

```
{
  "username": "euclid",
  "scope": "default",
```

```

"email": "euclid@ldap.forumsys.com",
"dn": "UID=euclid,DC=example,DC=com"
}

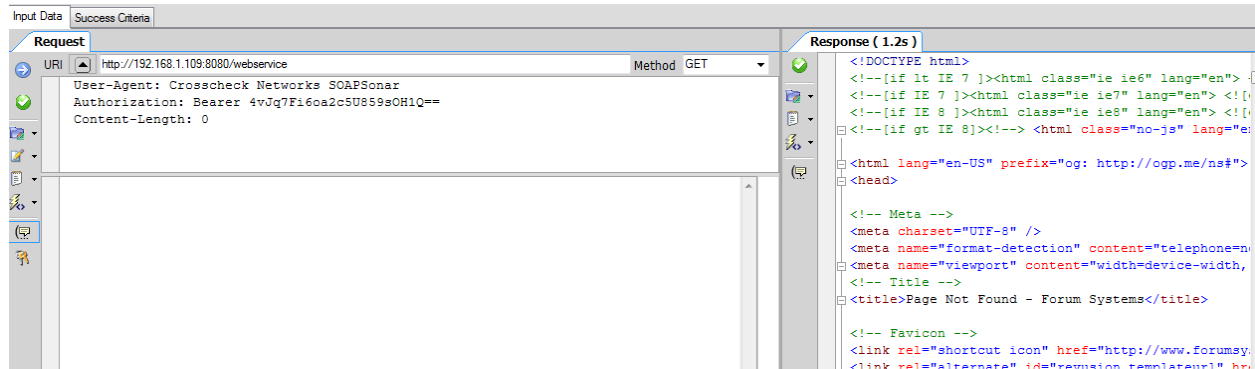
```

### **Test 3:**

This test will use the access token to connect to the [www.forumsys.com](http://www.forumsys.com) service via Forum Sentry. The access token will be validated before access to the site is allowed.

- Set the URI to <http://oauth-server-ip-address:port/webservice>
- Set method to POST

*Note: As you can see below, we have generated a new access token to use with the service call.*

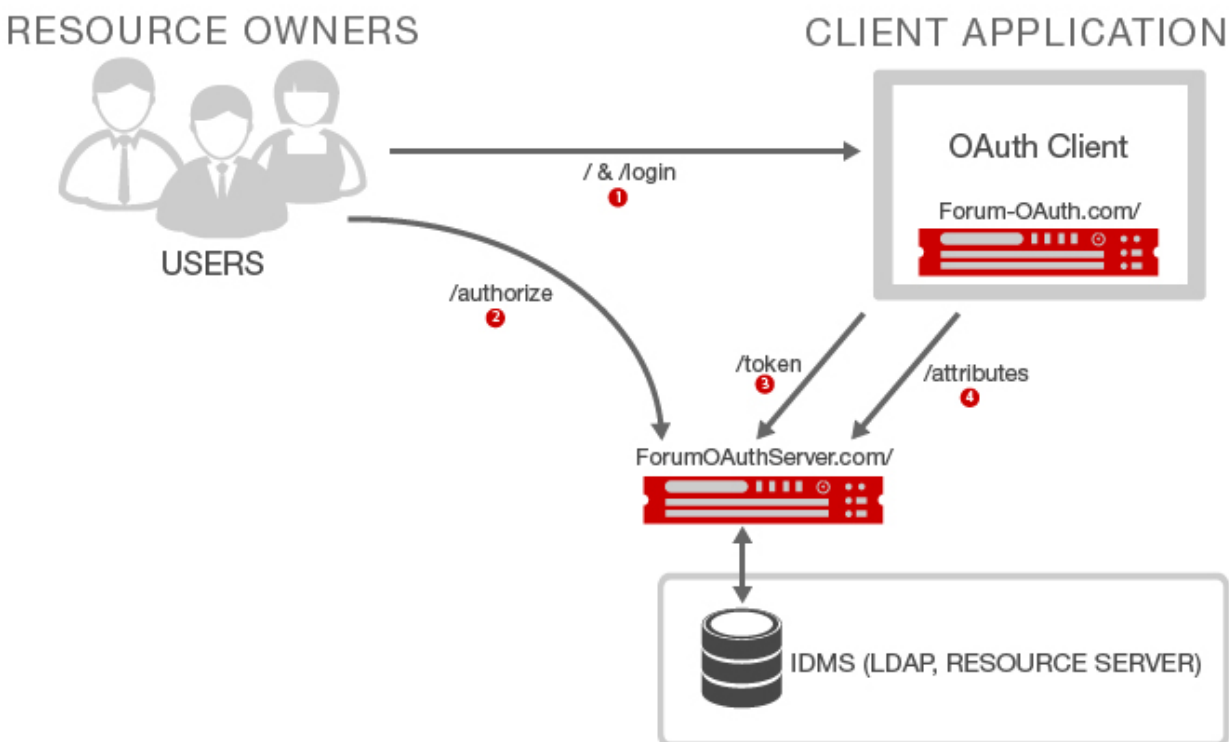


### **Additional Tests not covered above:**

- **Client Type Confidential:**
  - Go back to the OAuth policy and select Confidential. This will require the use of the client secret.
  - Send a Get Token request. This should now fail.
  - Add the client\_secret to the request in the Get\_Token test. Make sure you urlencode the secret since some characters when decoded by Sentry might end up translated to other characters  
Example: `($fn:UrlEncode(w8KVcmFvxkQg5zZdZsoC+w==))$`
- **Make use of the User ACL create earlier:**
  - Add the User ACL to the OAuth Policy
  - Send a Get Token request. This should now fail.
  - Add an allowed user (i.e. euclid/password) after selecting Basic Authentication in the Authentication tab on SAOPSonar.
  - Save and send a request

## Example – Authorization Code Grant Type

The authorization code grant type is the most common. It is used to obtain both access tokens and refresh tokens and is optimized for confidential clients. Confidential clients are ones capable of securing client secrets as well as user information. The end-user is redirected to the authorization server where it authenticates and is then supplied with a code that the client application uses to get an access token.



Note: The steps in the image above are further broken down into multiple steps and are explained in the flow section below.

- Forum API Gateway (OAuth Server) - Identified by a DNS name forumoauthserver.com
- Accessible LDAP server - ldap.forumsys.com
- A standard web browser as the user
- Accessible back-end website - www.forumsys.com
- A second Forum API Gateway (OAuth Client) - Identified by a DNS name forum-oauth.com

Note: All URLs will be HTTPS therefore please make sure you have setup SSL initiation and termination policies on the client application as well as the OAuth server. (Both are Sentry in this use case below)

### The steps below explain the flow:

#### 1. Initial contact with the application client (OAuth Client):

- User does a GET (<https://forum-oauth.com/>) to the OAuth client
- OAuth Client responds with a 303 and redirects user to /login
- User does a GET /login (<https://forum-oauth.com/login>)

- OAuth Client responds with a 303 redirect to the OAuth server (<https://forumoauthserver.com/authorize>)
2. OAuth server contact
- User does a GET (<https://forumoauthserver.com/authorize>)
  - OAuth server responds with 401 unauthorized and prompts user to authenticate
  - User does a GET (<https://forumoauthserver.com/authorize>) this time with a username/password
  - OAuth server authenticates against LDAP and returns a redirect back to <https://forum-oauth.com/login> with a code
3. OAuth client and server interaction:
- OAuth client uses the code to <https://forum-oauth.com/token>
  - OAuth server responds with a token, scope, etc...

i.e.

```
{
  "scope":"default",
  "expires_in":"7200",
  "token_type":"Bearer",
  "access_token":"6U920mmSbBbrUYN6ejFi5g=="
}
```

4. OAuth client does a POST to <https://forum-oauth.com/attributes>:
- OAuth client does a POST <https://forum-oauth.com/attributes> with the token
  - OAuth server responds with the user's attributes:

```
{
  "username":"euclid",
  "scope":"default",
  "dn":"UID=euclid,DC=example,DC=com"
}
```

## Forum API Gateway as the OAuth Server

### Setting up an Identity and Access Management (IAM) :

*Note: For our example below we will setup LDAP (You can also use local accounts created in Sentry).*

1. Log in to the web admin and go to **ACCESS→User Policies→LDAP** and click **New**
2. LDAP setup (You can use the [Online LDAP Test Server](#) provided by Forum Systems):

**LDAP POLICIES > LDAP POLICY**

Successfully authenticated with LDAP server

[Always Show Advanced](#)

---

**LDAP SERVER**

Policy Name\*:

Enable privileged access: ☐ Yes ☒ No

Restrict Menus: ☐

Role policy:

Server\*:

Port\*:

Use SSL to connect: ☐

Authentication type:

User\*:

Password\*:

Cache timeout (in minutes):

Read Timeout (in minutes):

---

**MAPPINGS**

Toggle Mappings:

Root DN\*:

User/group context: ☐ List of users ☒ Group containing users

Optimized group search: ☐

[Test](#) [Apply](#) [Save](#)

3. Add a User ACL and associate the LDAP Policy to it:

**USER ACL MANAGEMENT > USER ACL DETAILS**

**USER ACL DETAILS**

ACL Name: LDAP\_FS\_ACL

#	USER GROUP	EXECUTE
1 ↓	<a href="#">LDAP-LDAP FS ADMIN</a>	<input checked="" type="checkbox"/>
2 ↑	<a href="#">LDAP_FS_Group</a>	<input type="checkbox"/>

[Save](#)

4. Add a Listener Policy:
  - Click on **GATEWAY→Network Policies→Network Policies**.
  - Click **New** to a Listener policy
  - Click next to step through filling in the required information (see image below for an example)
  - *Note: Port 443 is the default HTTPS port, but this can be changed if you have any other listener or application on the system using this port.*

## NETWORK POLICIES > HTTP LISTENER POLICY

### POLICY NAME

Policy Name\*:

[Next](#)

### POLICY SELECTIONS

<u>Policy Name:</u>	ForumOAuthListener
<u>IP ACL Policy:</u>	Unrestricted
<u>Inbound Protocol:</u>	HTTPS
<u>Listener:</u>	0.0.0.0:443
<u>Password Authentication:</u>	
<u>SSL Termination Policy:</u>	SSL_Termination_Policy
<u>Error Handling:</u>	Default Template

### 5. Create the OAuth Server Policy:

- Go to **GATEWAY** → **Content Policies** → **OAuth Policies**
- Click **New**
- Add an appropriate Name for you policy (see image below):

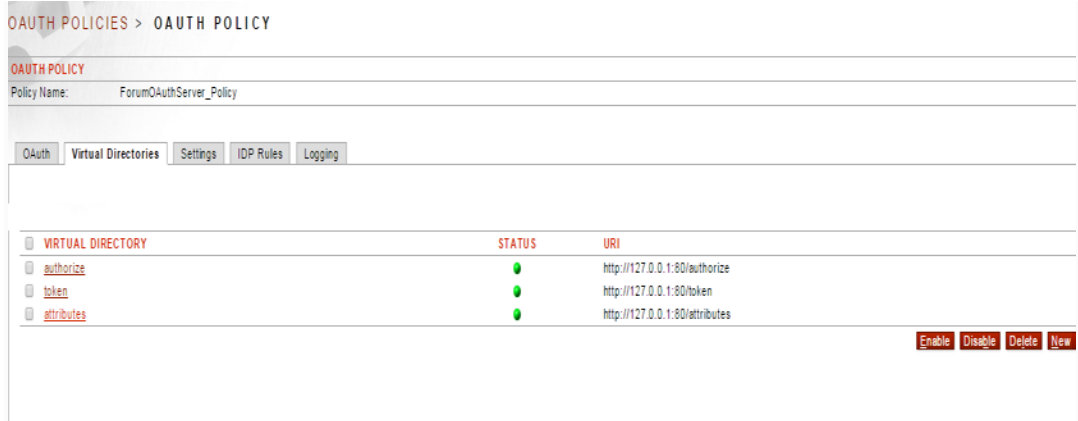
## OAuth POLICIES > NEW OAUTH POLICY

### NEW OAUTH POLICY

Name*:	<input type="text" value="ForumOAuthServer_Policy"/>
Description:	<div></div>
Labels:	<div></div>
Client Authentication Mode:	<input checked="" type="radio"/> Use these client credentials Client Id*: <input type="text" value="client1"/> Client Secret*: <input type="text" value="9xSsqT3rizT4LaMYAUB3RQ=="/> Redirect URI*: <input type="text" value="http://www.forumsys.com/sentry"/> <input type="radio"/> Use this ACL to authenticate client Client Authentication ACL: <input type="text" value="[Allow All]"/>
Client Type:	<input checked="" type="radio"/> Confidential (Web Application) <input type="radio"/> Public (Native or JavaScript)
Grant types*:	<input checked="" type="checkbox"/> Authorization code <input type="checkbox"/> Implicit <input type="checkbox"/> Password <input type="checkbox"/> Client credentials
Scope*:	<input type="text" value="default"/>
Default Access Token Lifetime (secs)*:	<input type="text" value="7200"/>
<input type="checkbox"/> Default Refresh Token Lifetime (days):	<input type="text" value="60"/>
<input type="checkbox"/> Reuse refresh token	
<input type="checkbox"/> Enable persistent token storage	<input type="text" value="Database_Policy"/> <a href="#">Edit</a>
ACL Policy (password grant):	<input type="text" value="[Allow All]"/>
Request Task List Group:	<input type="text" value="[None]"/>
Response Task List Group:	<input type="text" value="[None]"/>

[Next](#)

Note: Adding an OAuth Policy automatically adds 3 built in OAuth virtual directories as shown below:



The screenshot shows the 'OAuth POLICIES > OAUTH POLICY' configuration page. The 'Policy Name' is 'ForumOAuthServer\_Policy'. The 'Virtual Directories' tab is selected. A table lists three virtual directories: 'authorize', 'token', and 'attributes'. Each has a status of 'ON' (indicated by a green dot) and a specific URI. At the bottom right, there are buttons for 'Enable', 'Disable', 'Delete', and 'New'.

VIRTUAL DIRECTORY	STATUS	URI
<a href="#">authorize</a>	ON	http://127.0.0.1:80/authorize
<a href="#">token</a>	ON	http://127.0.0.1:80/token
<a href="#">attributes</a>	ON	http://127.0.0.1:80/attributes

Enable Disable Delete New

- Click on the /authorize virtual directory and make sure you have the following settings:
  - Password Authentication: Specify
  - Use basic authentication: [Checked]
  - Require password authentication: [Checked]

**OAUTH POLICIES > OAUTH POLICY**

---

**OAUTH POLICY**

Policy Name: ForumOAuthServer\_Policy

---

OAuth Virtual Directories Settings IDP Rules Logging

---

**Virtual Directories > Virtual Directory: authorize**

---

**VIRTUAL DIRECTORY**

Name\*: authorize

Description:

Listener Policy: ForumOAuthListener [Edit](#)

☐ Use virtual host as a regular expression

Virtual Host:

☐ Enable Virtual Path Case Insensitivity

Virtual Path: /authorize

Virtual URI: https://127.0.0.1:443/authorize(/.\*)"?(\\?.\*)"?

Filter Expression: (/.\*"?\\?.\*)"?

IP ACL Policy: Unrestricted [Edit](#)

ACL Policy: [Allow All]

Password Authentication: [Specify]

Use basic authentication: ☒

Use digest authentication: ☐

Use kerberos authentication: ☐

Use cookie authentication: ☐

Use form post authentication: ☐

Username Parameter:

Password Parameter:

Require password authentication (any type): ☒

Password Authentication Realm:

Redirect Policy: [None]

Error Template: [From Listener Policy]

## Forum API Gateway as the OAuth client

### A. Add Listener and Remote Network Policies

Note: When you get to Remote Server make that the target system you want to go to ([www.forumsys.com](http://www.forumsys.com))

- Go to **GATEWAY→Network Policies→Network Policies**.
- Click **New** to add a Listener Policy
- Go through the prompts by clicking **Next**. You listener should match the following (with the exception of the port, which can be specified any value, it does not have to be the default 443):



## NETWORK POLICIES > HTTP LISTENER POLICY

### POLICY NAME

Policy Name\*:

**Next**

### POLICY SELECTIONS

<u>Policy Name:</u>	HttpListenerPolicy
<u>IP ACL Policy:</u>	Unrestricted
<u>Inbound Protocol:</u>	HTTPS
<u>Listener:</u>	0.0.0.0:443
<u>Password Authentication:</u>	
<u>SSL Termination Policy:</u>	SSL_Termination_Policy
<u>Error Handling:</u>	Default Template

- Go to **GATEWAY→Network Policies→Network Policies**.
- Click **New** to add a Remote Policy
- Go through the prompt by clicking **Next**. You remote should match the Following:

## NETWORK POLICIES > HTTP REMOTE POLICY

### POLICY NAME

Policy Name\*:

**Next**

### POLICY SELECTIONS

<u>Policy Name:</u>	HttpsRemotePolicy
<u>Outbound Protocol:</u>	HTTPS
<u>Remote Server:</u>	www.forumsys.com:443
<u>SSL Initiation Policy:</u>	SSL_Initiation_Policy
<u>TCP Timeouts:</u>	Connect: 10 Read: 600 Connection Limit: Unlimited
<u>Process Response:</u>	Off

Note: We will need to add one more remote policy that will communicate with the OAuth Server and supply user credentials when prompted.

- Go to **GATEWAY→Network Policies→Network Policies**.
- Click **New** to add a Remote Policy
- Go through the prompt by clicking **Next**. You remote should match the Following:

## NETWORK POLICIES > HTTP REMOTE POLICY

### POLICY NAME

Policy Name\*:

**Next**

### POLICY SELECTIONS

<u>Policy Name:</u>	HttpRemote_ForumOAuthServer
<u>Outbound Protocol:</u>	HTTPS
<u>Remote Server:</u>	forumoauthserver.com:443 (Basic Auth)
<u>Remote Authentication:</u>	Propagate client's credentials
<u>SSL Initiation Policy:</u>	SSL_Initiation_Policy
<u>TCP Timeouts:</u>	Connect: 10 Read: 600 Connection Limit: Unlimited
<u>Process Response:</u>	Off

### B. Add a Task (User identity & Access Control):

1. Go to **GATEWAY**→**Task Policies**→**Task Lists**
2. Click **New** and name the Task (i.e. Task\_OAuth\_ForumServer\_GrantType\_Code) then click **Apply**
3. Click **New** and select "User Identity & Access Control" then click **Next**
4. Uncheck "Map identified user to a known user" then click **Next**
5. CHECK Validate OAuth SSO token & establish identity, **NEXT**
6. For Authorization URL enter "<https://forumoauthserver.com/authorize>"
7. For Scope enter "default"
8. For Token Endpoint Policy enter "HttpRemote\_ForumOAuthServer"
9. For Token Endpoint Path enter "/token"
10. SELECT Other to user Forum Sentry as the OAuth Service provider **NEXT**
11. Enter Client Id and Client Secret here these come from the OAuth server, **NEXT**
12. For Service Endpoint Policy enter "HttpRemote\_ForumOAuthServer"
13. For Service Endpoint Path enter "/attributes"
14. For User name Field enter "username"
15. ENTER "origUri" for Redirect Parameter, **FINISH**--->**SAVE**

TASK LISTS > TASK LIST: TASK\_OAUTH\_FORUMSERVER\_GRANTTYPE\_OAUTH >

---

**TASK NAME**

Task Name\*:

**Next**

---

**USER IDENTITY & ACCESS CONTROL**

Task Type:	User Identity & Access Control
<u>Task Name:</u>	User Identity & Access Control
<u>ACL Policy:</u>	No user mapping
<u>User Identity Mechanism:</u>	Validate OAuth SSO token & establish identity
<u>Identity Provider:</u>	Other
<u>Authorization URL:</u>	https://forumoauthserver.com/authorize
<u>Scope:</u>	default
<u>Token Endpoint Policy:</u>	HttpRemote_ForumOAuthServer
<u>Token Endpoint Path:</u>	/token
<u>Client Credentials:</u>	client1 (Form)
<u>Service Endpoint Policy:</u>	HttpRemote_ForumOAuthServer
<u>Service Endpoint Path:</u>	/attributes
<u>Request Task List Group Name:</u>	
<u>Response Task List Group Name:</u>	
<u>User Name Field:</u>	username
<u>Redirect Parameter:</u>	origUri
<u>Error Template:</u>	[From Policy]

**C. Create a Task List Group and add the Task List created above to it**

- Go to **GATEWAY** → **Task Policies** → **Task List Groups**
- Click **New**, name the policy and then click **Create**
- Select the task list from above and click **Add** then **save**

TASK LIST GROUP > TASK LIST GROUP DETAILS

---

**TASK LIST GROUP DETAILS**

Task List Group Name\*:

Description:

Process Each Task List Below in Sequence: ☐

Labels:

---

#	TASK LIST
1	<input checked="" type="checkbox"/> Task_OAuth_ForumServer_GrantType_Code (1)
	<input type="text" value="Task_OAuth_Google_GrantType_OAuth_Code"/> <b>Add</b>

**D. Create a Redirect Policy (i.e. NoAccess\_Redirect\_Policy would be indicative of what it does):**

- Check Authentication Fails and fill in the Redirect URI (explanation below)
- Check No Credentials and fill in with the Redirect URI
- Check "Include Origin URI" field with the URI Parameter Name "origUri" (Make sure the case matches the one in the task list above), **Save**

Explanation of a and b above:

What you are configuring is a sub-policy which will be eventually consumed by the HTML policy you will create later. This sub-policy tells Sentry that when the user first connects to <http://forum-oauth.com/> and the user carries no credentials then the user will be redirected to a new location such as <http://forum-oauth.com/login>

- Go to **GATEWAY** → **Redirect Policies** → **Redirect Policies**
- Click **New** and add similar to the example below:

REDIRECT POLICIES > REDIRECT POLICY

---

**REDIRECT POLICY**

Name\*:

Description:

Labels:

---

<input type="checkbox"/> EVENT	URL	USE HOST P
<input type="checkbox"/> Authentication Success	<input type="text"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/> Authentication Failure	<input type="text" value="https://forum-oauth.com/login"/>	<input type="checkbox"/>
<input checked="" type="checkbox"/> No Credentials	<input type="text" value="https://forum-oauth.com/login"/>	<input type="checkbox"/>
<input type="checkbox"/> On Error	<input type="text"/>	<input type="checkbox"/>

Include Original URI: ☒

URI Parameter Name\*:

**E. Create an HTML Policy (a good name is: Browser\_to\_Sentry\_OAuth\_Policy):**

- Select the Listener and Remote set up above. Click **Finish**
- Click on New Virtual Directory link:
  - Change the Name to a meaningful one (a good name is "Initial\_Contact")
  - Add Virtual Path / (user will be coming on initial URI, for example <https://forum-oauth.com/>)
  - Change Filter Expression to (.\*)
  - For Password Authentication Select *Specify*
  - Check Use Cookie Authentication (must be checked)
  - Check Require password authentication (any type) (must be checked)
  - Select NoAccess\_Redirect\_Policy for Redirect Policy
  - Click **Save**
- Click **New** to create another Virtual Directory for "/login":
  - Name: Login
  - Virtual Path: /login
  - Filter Expression: (.\*)
  - Make sure that Send to remote server option is unchecked

- . Select **Task\_Group\_OAuth\_ForumServer\_GrantType\_OAuth** for Request Task List Group
- . Click **Save**
- . Click Settings and check “Enable session cookies” and “Use secure cookies”, **Save**

### Testing and Logging:

The URL you will be testing with would be the one used in the steps above and must be resolvable (DNS entry). If you have used our example above then go to a browser after type `https://forum-oauth.com` (of course `forum-oauth.com` must be resolvable by all systems involved).

The system log will contain the detailed information showing all interactions and operation. You will need to visit the system log on both the server and the client. First, take a look at the access log on both client and server. You should see the following:

On the OAuth client:

INTERNAL LOGS > ACCESS LOG

OCT 28, 2014

Search:  Search

Refresh:  (0-30 secs)

Filter By Log Level:

4 items found, displaying all items.1

Time	Session	IP	Host Header	Type	URI	Code	Length
11:06:58.777	X0003D1	127.0.0.1	forum-oauth.com	GET	/	200	60360
11:06:57.923	X0003D0	127.0.0.1	forum-oauth.com	GET	/login	303	346
11:06:35.556	X0003CF	127.0.0.1	forum-oauth.com	GET	/login	303	346
11:06:35.534	X0003CE	127.0.0.1	forum-oauth.com	GET	/	303	309

On the OAuth server:

INTERNAL LOGS > ACCESS LOG

OCT 28, 2014

Search:  Search

Refresh:  (0-30 secs)

Filter By Log Level:

4 items found, displaying all items.1

Time	Session	IP	Host Header	Type	URI	Code	Length
06:43:39.121	X000647	10.5.1.166	forumoauthserver.com	GET	/attributes	200	115
06:43:39.112	X000646	10.5.1.166	forumoauthserver.com	POST	/token	200	108
06:43:39.092	X000645	10.5.1.166	forumoauthserver.com	GET	/authorize	303	358
06:43:19.237	X000644	10.5.1.166	forumoauthserver.com	GET	/authorize	401	331

There is much more detail in the system log files, but as you can see above the entries in the access log should match exactly with the flow described at the beginning above.