The ArcSight™ ESM Service Layer

Introduction

Beginning with ArcSight™ ESM v5.0, the ESM Service Layer is available and exposes ESM functionalities as Web Services. By consuming the exposed Web Services, you can integrate ESM functionality in your own applications. ESM Service Layer uses a service-oriented architecture (SOA) that supports multiple Web Service clients written in different languages.

You will have the ability to create programs that

- Run an ESM report and feed it back to your third-party home-grown system
- Execute full-text searches on ESM resources

The SOA approach enables ESM Service Layer to support multiple consumption options, for example:

- Java developers can take advantage of the ESM Service Layer SDK to create SOAP or Google Web Toolkit (GWT) clients.

  See "ESM Service Layer SDK" on page 2.

- Developers applying Representational State Transfer (REST) principles write scripts to consume the services. The developers can refer to the Web Services Description Language (WSDL) files for a description of each ESM service.

  For more information about REST, refer to this link:

  http://en.wikipedia.org/wiki/Representational_State_Transfer

- Developers who prefer to create their own stubs in a language other than Java (for example, in .NET or C++) can refer to the Web Services Description Language (WSDL) files for a description of each ESM service.

ESM Resources as Web Services

ESM Service Layer provides access to the ESM resources listed below.

ESM Resource Name	Service Name
Archive Report	ArchiveReportService
Dashboard	DashboardService
Data Monitor	DataMonitorService
File	FileResourceService
Report	ReportService
Resource	ResourceService

ArcSight ESM has more resources than listed. If you are retrieving information on an ESM resource that is not yet supported, you can use ResourceService to get the Base Resource attributes that are common to all resources (the ID, name, and description), but not the details unique to the unsupported resource.

ESM Service Modules

ESM Service Layer groups services in two service modules:

- **Core Service module**. This module provides login services (`loginService`) by returning the authentication token (`authToken`) needed to begin consuming a service. The services are designed to be stateless. You will therefore pass the authentication token every time you consume a service.
- **Manager Service module**. This module provides the ESM functionality, for example, ArchiveReportService.

ESM Service Layer SDK

The SDK provides a set of tools and libraries for Java applications that consume the services in ESM Service Layer.

- SOAP clients use Simple Object Access Protocol (SOAP) XML messages to send requests to and get responses from the ESM Service Layer web server over HTTP.
- The Google Web Toolkit (GWT) provides the capability to create user interfaces, use RPC to pass Java objects between the client and the server over HTTP, and more.

  For more information about GWT, start with this link:

  http://code.google.com/webtoolkit/doc/overview.html

SDK Installation files

The SDK is distributed as part of the ESM installation. Installation files are located at

```
$ARCSIGHT_HOME/utilities/sdk
```

The SDK libaries are located at

```
$ARCSIGHT_HOME/utilities/sdk/lib
```

Under `/lib`, you will also find the Javadoc containing the API descriptions. The Javadoc is distributed in jar format.

Setting Up Your Development Environment

Following are the requirements to set up your development environment:

- All exposed ESM services are TLS/SSL-secured, therefore, import the ArcSight ESM Manager's certificate into your development/runtime environment. The certificate option was chosen during ArcSight ESM installation. It could be a temporary certificate authority (CA), a self-signed certificate, or a signed certificate from a trusted CA. Ask your ArcSight administrator about which certificate option was chosen during installation and import that certificate into your development JRE's `jre/lib/security/cacerts`.
- The ESM Sevice Layer modules are `core-ws-client.jar` and `manager-ws-client.jar`, respectively. Include these jar files in your Java classpath.
- Install the Java API for XML Web Services (JAX-WS) libraries, for example, the toolkit for Apache Axis2, from your preferred software provider.

Obtaining a List of Available ESM Services

After setting up your development environment, you will next want to know the services available for consumption. You do this by displaying the listServices file provided by the Manager Service module.

To view the listServices file

1. Open your browser and enter the URL with the following format:

   https://*myhost*:8443/www/manager-service/services/listServices

2. Replace *myhost* as appropriate.

The browser displays the listServices page. Scroll down to view more services. The following example is the ArchiveReportService.

The listServices file provides the information about a service, including:

- The service's end point reference (EPR) URL
- A list of service names, for example, ArchiveReportService
- The methods associated with each service

The parameters associated with each method are available in the resource's WSDL file (described in "Finding Service Information in the WSDL" on page 5).

Obtaining the Authentication Token

An authentication token is the first requirement for accessing ESM Service Layer to obtain service information and then consume the services. Two examples on how to get this token are provided: REST and SOAP. As explained earlier, the Core Service module handles authentication token requests. You will then use the returned token to log in to the Manager Services module and consume the desired service.

REST Example

The following example shows how to enter the URL to the Core Service module and obtain the `authToken` string. The first part of the URL address, https://*host*:8443/www/, constitutes the base URL. You will always start your URLs with this base URL.

To log in and obtain an authentication token

1. In the browser, enter the URL in the following format:

   https://*myhost*:8443/www/core-service/rest/LoginService/login?login=*admin*&password=*password*

2. Replace *myhost*, *admin*, and *password* as appropriate.

The browser displays the response which is the authentication token string. You will pass that string every time you consume a service.

Java Example

The following example shows how to invoke the login service of the Core Service module and obtain the `authToken` string. You will pass this string every time you consume a service. You will also set the base URL in the format https://*host*:8443/www/.

```
//================================================
//  Invoke the Login Service
//================================================

//construct LoginServiceFactory (loginService is part of Core Service module)
LoginServiceClientFactory loginServiceClientFactory = new
LoginServiceClientFactory();

//set the service base url. ESM's service base URL is https://host:8443/www/
loginServiceClientFactory.setBaseURL("https://myhost:8443/www/");

//create service client instance from factory
LoginService loginService = loginServiceClientFactory.createClient();

//invoke login service and get authToken
String authToken = loginService.login(null, "admin", "password");
```

Finding Service Information in the WSDL

The ESM Service Layer's Web Service Description Language (WSDL) files are XML-formatted documents describing ESM services, one WSDL file for each service. WSDLs are used to generate clients automatically. Programmers who are writing their own stubs instead of using the SDK can refer to the WSDLs to get information about ESM services.

This topic takes you through different parts of the WSDL file using the ArchiveReportService's `findByUUID` method as an example. The purpose of the `findByUUID` method is to find a resource by its ID. Based on this ID, you will be able to obtain additional details about the resource.

Using fragments taken from the WSDL, the example walks you through the following process:

- Obtain the method parameters
- Obtain the response

To display the WSDL for a specific service

1. On your browser, enter the URL using the following format:

   https://*myhost*:8443/www/manager-service/services/*servicename*?wsdl

2. Replace *myhost* with the actual server and *servicename* with the service you want to consume, for example, **ArchiveReportService**. See "Obtaining a List of Available ESM Services" on page 3 for information about supported ESM services.

The browser displays the WSDL file for the specified service.

Finding the Service's URL

The following WSDL fragment contains the URL to the service.

```
<wsdl:service name="ArchiveReportService">
       <wsdl:port name="ArchiveReportServiceHttpport"
                    binding="ns0:ArchiveReportServiceHttpBinding">
              <http:address location="http://localhost:9090/manager-
                    service/services/ArchiveReportService"/>
       </wsdl:port>
</wsdl:service>
```

Finding the Method

The following WSDL fragment provides the URL to the method of interest, `findByUUID`.

```
<wsdl:operation name="findByUUID">
       <http:operation location="ArchiveReportService/findByUUID"/>
       <wsdl:input>
              <mime:content type="text/xml" part="findByUUID"/>
       </wsdl:input>
       <wsdl:output>
              <mime:content type="text/xml" part="findByUUID"/>
       </wsdl:output>
</wsdl:operation>
```

Finding the Parameters

The following WSDL fragment contains the XML Schema description of `findByUUID`'s parameters.

Method	Description
authToken	The parameter that takes the authentication token string
id	The ID of the resource to find

```
<xs:element name="findByUUID">
       <xs:complexType>
              <xs:sequence>
                     <xs:element minOccurs="0"
                            name="authToken"
```

```
                                    nillable="true"
                                    type="xs:string"/>
                             <xs:element minOccurs="0"
                                    name="id"
                                    nillable="true"
                                    type="xs:string"/>
                      </xs:sequence>
               </xs:complexType>
         </xs:element>
```

Obtaining the Output Description

The following WSDL fragment describes the response to the `findByUUID` request. The response indicates ArchiveReport to be a complex type.

```
<xs:element name="findByUUIDResponse">
       <xs:complexType>
              <xs:sequence>
                     <xs:element minOccurs="0" name="return"
                            nillable="true" type="ns1:ArchiveReport"/>
              </xs:sequence>
       </xs:complexType>
</xs:element>
```

Because ArchiveReport is a complex type, this means you will find additional details about the output, as shown in the following WSDL fragment:

```
<xs:complexType name="ArchiveReport">
       <xs:complexContent>
              <xs:extension base="ax23:Resource">
                     <xs:sequence>
                            <xs:element minOccurs="0" name="archiveType"
                                   nillable="true" type="xs:string"/>
                            <xs:element minOccurs="0" name="expireDate"
                                   nillable="true" type="xs:long"/>
                            <xs:element minOccurs="0" name="owner"
                                   nillable="true" type="xs:string"/>
                            <xs:element minOccurs="0" name="reportDefName"
                                   nillable="true" type="xs:string"/>
                            <xs:element minOccurs="0" name="reportFileName"
                                   nillable="true" type="xs:string"/>
                            <xs:element minOccurs="0" name="uploaded"
                                   nillable="true" type="xs:string"/>
                            <xs:element minOccurs="0" name="valid"
                                   type="xs:boolean"/>
                     </xs:sequence>
              </xs:extension>
       </xs:complexContent>
</xs:complexType>
```

Consuming ESM Services

This topic provides two examples:

- In REST, how to perform a text search on a resource
- For SOAP clients, how to download a report given report file's ID

Performing a Text Search on a Resource (REST Example)

The search is similar to the full text search performed on the ESM Console. It is assumed that the `authToken` string is available. In the example, a full text search is performed on the DataMonitor resource. The search results include the ID. Based on the returned ID, the ResourceService is then used to retrieve DataMonitor details.

To perform a text search:

1. In the browser, enter your query string to obtain the corresponding UUID. Enter the URL in the following format:

   https://*myhost*:8443/www/manager-service/rest/ManagerSearchService/search1?authToken=*authtokenstring*&queryStr=datamonitor *querystring*&pageSize=50

2. Replace *myhost* as appropriate, *authtokenstring* with the actual string you obtained in "Obtaining the Authentication Token" on page 4, and *querystring* with your actual string. For example, you are searching for DataMonitor with the name **event throughput**.

   The browser displays the UUID string corresponding to the resource that matches *querystring*. For example:

```
-<uri>
       /All_Data_Monitors/ArcSight_Administration/ESM/System_Health/Events
       /Event_Throughput/Event_Throughput
</uri>
<uuid>someUUIDstring</uuid>
```

   Take note of the returned UUID string. You will use the `findByUUID` method in ResourceService and pass the UUID string to get the data details about the DataMonitor resource with that UUID.

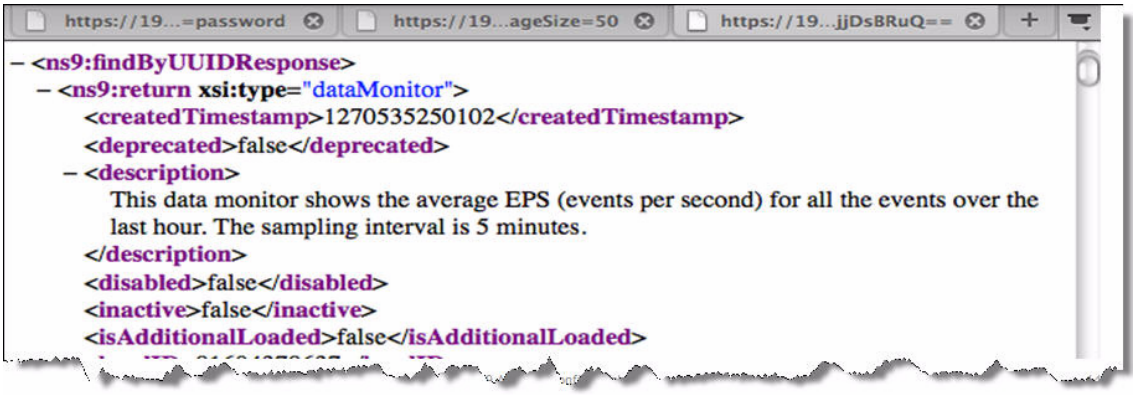
3. In the browser, enter the URL in the following format:

   https://*myhost*:8443/www/manager-service/rest/ResourceService/findByUUID?authToken=*authtokenstring*&id=*UUIDstring*

   The first part of the URL that starts with https://*myhost*:8443/www/ is called the base URL.

4. Replace *myhost* as appropriate, *authtokenstring* with the actual string you obtained in "Obtaining the Authentication Token" on page 4, and *UUIDstring* with the actual UUID string returned by your query.

   The browser displays the resource information. A partial example is shown below:



Downloading an Archived Report (SOAP Example)

The following Java example for SOAP clients shows how to invoke ArchiveReportService, set the base URL in the format https://*host*:*port*/www/, and obtain the report's file ID. You will then pass this ID to download the archived report. The example assumes you have invoked the login service and passed the `authToken` string prior to invoking the ArchiveReportService. See "Obtaining the Authentication Token" on page 4.

```
//  Invoke Login Service here and pass the authToken

//================================================
//  Invoke Archive Report Service
//================================================

//Construct ArchiveReportServiceFactory (ArchiveReportService is part of
//the Manager Service module)
ArchiveReportServiceFactory archiveReportServiceClientFactory = new
ArchiveReportServiceFactory ();

//Set the service base URL. ESM's service base URL is https://host:port/www/
archiveReportServiceClientFactory.setBaseURL("https://myhost:8443/www/");

//Create service client instance from factory
archiveReportService archiveReportService =
archiveReportServiceClientFactory.createClient();

//Invoke report service to create archiveReport by its ID. This returns
//the archived report's file ID. Use that ID to download report.

String fileId =
archiveReportService.initDefaultArchiveReportDownload(authToken,"
authtokenstring", "Manual");

 //Download report using the obtained fileId

       /**
        * Here is the example of using the fileId to download the report:

        * https://myhost:8443/www/manager-service
        * /fileservlet?file.command=download&file.id
        * =2r2Yp5RYNQ2WSmVWa2V9_yAuNLSS4TdTQMV2T3upay4
        */
```

Revision History

Date	Product Version	Description
05/30/10	First version	Introductory content to the ArcSight ESM Service Layer.