



FORUM SYSTEMS HANDS-ON TRAINING

LAB 5. PROTECTING A SOAP WEB SERVICE WITH FORUM SENTRY



FORUM SYSTEMS

A Crosscheck Networks Company

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 © 2002-2014 Forum Systems, Inc. – All Rights Reserved.

Published: September 2014

Forum Systems Hands-on Training – Lab 5. Deploying a SOAP Web Service through Forum Sentry
D-ASF-SE-010029

Contents

Introduction.....	4
<i>Skill Level</i>	4
<i>Prerequisites</i>	4
<i>Lab Overview</i>	4
Forum Sentry WSDL Policies.....	5
<i>Network Policies</i>	5
<i>Virtual Directories</i>	5
<i>Building a WSDL Policy</i>	5
<i>Testing a WSDL Policy</i>	8
Transactions in the Sentry Logs	10
Test Sentry Schema Validation.....	13
<i>BACK IT UP!</i>	16
<i>Additional Tests and Discussion Topics</i>	16
<i>Additional Information</i>	16
About Forum Systems.....	17

Introduction

Lab 5. Deploying a SOAP Web Service through Forum Sentry

Skill Level

This lab is beginner skill level. Little to no prior experience with Forum Sentry or SOAPSonar is required.

Prerequisites

This lab requires the Forum Sentry Training Image, with a licensed copy of SOAPSonar Enterprise Edition.

Refer to the “FS_Training_Labs_v8-1_Introduction” document for information on the Forum Sentry Training Image and licensing SOAPSonar Enterprise Edition.

The sample SOAP web service built in Lab 1 of this training series will be used with this lab.

General knowledge of testing a SOAP service using SOAPSonar is assumed. This is covered in Lab 2 of this training series.

Lab Overview

This lab provides instructions for deploying the Sample Web Service, built in Lab 1, through Forum Sentry.

Prior to this service being deployed through Sentry, clients access the service directly. A primary function of an API Gateway is to broker the traffic - behaving as a reverse proxy - so that the client cannot access the service directly.

Once the service is deployed through Sentry, the clients should never have any access to the actual service. As far as the client is concerned, the Sentry endpoint is the service.

This lab will provide instructions for deploying a SOAP service with Forum Sentry.

Topics include:

1. WSDL Policies
2. Network Policies
3. Schema Validation – Default IDP Rules
4. Reviewing transactions in the Sentry System and Access Logs

Forum Sentry WSDL Policies

A SOAP service is deployed in Sentry through a WSDL Policy. The key components to a WSDL policy are:

1. Network Policies
2. A Virtual Directory
3. A WSDL File

Network Policies

While building the WSDL Policy in Sentry, you will also build the Network Listener and Network Remote Policies. The network policies are essentially the “plumbing” or network framework that are used to get the runtime traffic into Sentry for processing, and then sent out to the remote server. We will be working with two types of network policies in this lab:

1. Listener Policies – Listen on IP/Port and is used to accept incoming traffic
2. Remote Policies – Define where to send the processed request

Virtual Directories

The virtual directories are built as the WSDL policy is built. The virtual directory ties together the network listener and network remote policies. This is also where many settings for the WSDL policy are modified including:

1. Virtual Path – The virtual URI for the service (the URL that the clients will use to access the service)
2. Remote Path – the URI for the remote service endpoint
3. Enable WSDL Access – To enable or disable dynamic WSDL retrieval from the WSDL policy
4. Many more – Policy level authentication, overriding the endpoint location in the exported WSDL, modify the HTTP content-types and methods allowed, etc...

Building a WSDL Policy

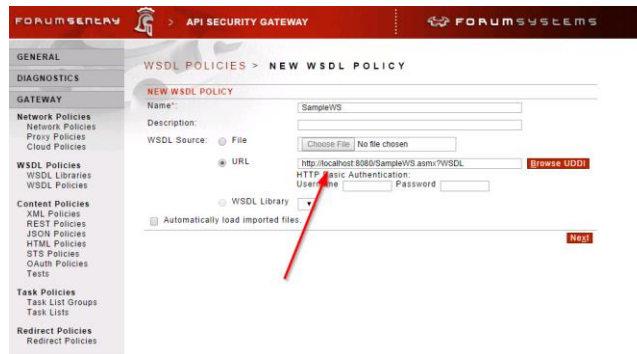
In this step we will import the WSDL from the Sample Web Service built in Lab 1 into Sentry. Importing the WSDL into Sentry tells the system:

1. The operations that are available for the service
2. Where the endpoints for the service reside
3. What the request/response SOAP messages should look like
4. What versions of SOAP the service supports.

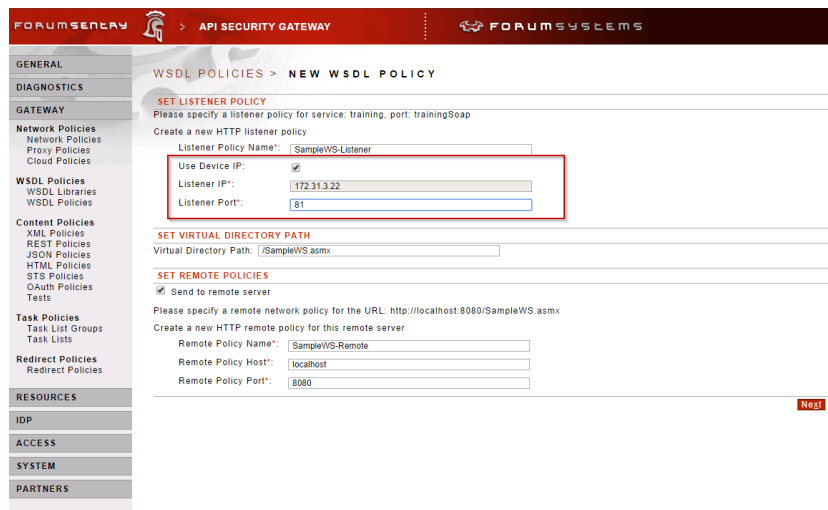
Note: The Sample Web Service WSDL supports both SOAP 1.1 and SOAP 1.2 so two separate endpoints will be created for each version of SOAP

Follow the steps below to build a WSDL Policy in Sentry.

1. Navigate to the Gateway→WSDL Policies→WSDL Policies page and click New
2. In the URL field, enter the URL to the WSDL for the Sample Web Service (built in Lab 1). This should be: <http://localhost:8080/SampleWS.asmx?WSDL>. The name field will be filled in automatically. Click NEXT.



3. This will import the WSDL into Sentry. On the next screen, the wizard will ask you to configure the Listener policy, the Virtual Directory Path, and the Remote Policy. Configure with the following criteria:
 - a. Check the “Use Device IP” box
 - b. Set the listener port to 81
 - c. Leave all other defaults (see screen shot below)
 - d. Click NEXT



4. After clicking Next, you'll be prompted again to set the Listener, Remote, and Virtual Path. This time, these settings are for the SOAP 1.2 port for this WSDL.
 - a. Change the Virtual Directory Path to: /SampleWS_SOAP12.asmx
 - b. Leave all other settings the same (use the listener and remote generated in step 3)
 - c. Click Next

FORUMSENTRY > API SECURITY GATEWAY **FORUMSYSTEMS**

WSDL POLICIES > NEW WSDL POLICY

SET LISTENER POLICY
Please specify a listener policy for service: training, port: trainingSoap12

Select from existing listener policies
[SampleWS-Listener (0.0.0.0:81)] [Edit](#)

Create a new HTTP listener policy
Listener Policy Name*: [SampleWS-Listener-2]
Use Device IP*: []
Listener IP*: [172.31.3.22]
Listener Port*: [80]

SET VIRTUAL DIRECTORY PATH
Virtual Directory Path: [/SampleWS_SOAP12.asmx]

SET REMOTE POLICIES
☒ Send to remote server
Please specify a remote network policy for the URL: http://localhost:8080/SampleWS.asmx

Select from existing remote policies
[SampleWS-Remote (localhost:8080)] [Edit](#)

Create a new HTTP remote policy for this remote server
Remote Policy Name*: [SampleWS-Remote-2]
Remote Policy Host*: [localhost]
Remote Policy Port*: [8080]

[Next](#)

- The WSDL Policy is created and you have successfully deployed the Sample SOAP Web Service through Sentry!

You'll notice there are two Virtual URIs and two Physical URIs. The first is for SOAP 1.1 and the second for SOAP 1.2. The Virtual URIs are the service endpoints that the clients will use to access this service. The Physical URIs are the back-end server URIs that Sentry will send the processed request to.

Sentry breaks out each operation of the service for both SOAP 1.1 and SOAP 1.2. This allows for setting access control, IDP rules, task lists, and schema tightening on a per operation basis.

FORUMSENTRY > API SECURITY GATEWAY **FORUMSYSTEMS**

WSDL POLICIES > WSDL POLICY

WSDL POLICY
Policy Name: SampleWS

[Upgrade](#) [Export WSDL](#) [Publish WSDL](#) [WSI Validation](#)

Services **Task Lists** **Settings** **IDP Rules** **Logging** **Documents**

SERVICE	PORT	STATUS	VIRTUAL URI	PHYSICAL URI
<input type="checkbox"/> training	trainingSoap	●	http://172.31.3.22:81/SampleWS.asmx	http://localhost:8080/SampleWS.asmx
<input type="checkbox"/> training	trainingSoap12	●	http://172.31.3.22:81/SampleWS_SOAP12.asmx	http://localhost:8080/SampleWS_SOAP12.asmx

[Enable](#) [Disable](#)

Service: training — Port: trainingSoap

OPERATION	STATUS	ACL	INPUT MESSAGE	OUTPUT MESSAGE	IDP GROUP
<input type="checkbox"/> Concat	●	[Allow All]	ConcatSoapIn	ConcatSoapOut	<input type="checkbox"/> Default Operation Group (0)
<input type="checkbox"/> Divide	●	[Allow All]	DivideSoapIn	DivideSoapOut	<input type="checkbox"/> Default Operation Group (0)
<input type="checkbox"/> Echo	●	[Allow All]	EchoSoapIn	EchoSoapOut	<input type="checkbox"/> Default Operation Group (0)
<input type="checkbox"/> Multiply	●	[Allow All]	MultiplySoapIn	MultiplySoapOut	<input type="checkbox"/> Default Operation Group (0)

[Enable](#) [Disable](#)

Service: training — Port: trainingSoap12

OPERATION	STATUS	ACL	INPUT MESSAGE	OUTPUT MESSAGE	IDP GROUP
<input type="checkbox"/> Concat	●	[Allow All]	ConcatSoapIn	ConcatSoapOut	<input type="checkbox"/> Default Operation Group (0)
<input type="checkbox"/> Divide	●	[Allow All]	DivideSoapIn	DivideSoapOut	<input type="checkbox"/> Default Operation Group (0)
<input type="checkbox"/> Echo	●	[Allow All]	EchoSoapIn	EchoSoapOut	<input type="checkbox"/> Default Operation Group (0)
<input type="checkbox"/> Multiply	●	[Allow All]	MultiplySoapIn	MultiplySoapOut	<input type="checkbox"/> Default Operation Group (0)

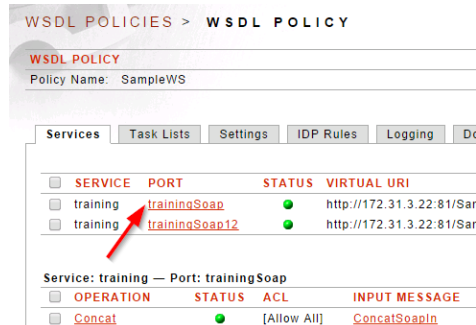
[Enable](#) [Disable](#)

Testing a WSDL Policy

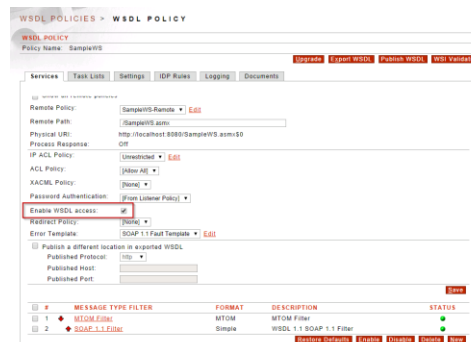
Now that the Sample SOAP service has been deployed through Sentry, the clients should only access this service through Sentry. Follow the steps below to test this Sentry secured service using SOAPSonar. For this lab we will only use SOAP 1.1. If you choose to you can disable the SOAP 1.2 virtual directory in Sentry to avoid confusion.

Follow these steps to test the Sentry WSDL Policy with SOAPSonar.

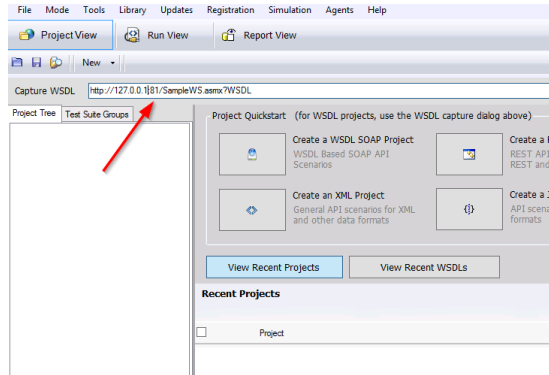
1. Enable dynamic WSDL Retrieval for this WSDL Policy
 - a. Open the WSDL Policy
 - b. Click the link under the Port column “trainingSOAP” to get to the Virtual directory page



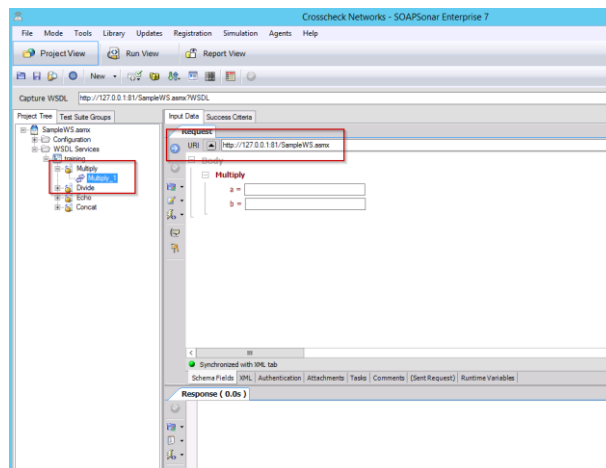
- c. On the Virtual Directory page, scroll down, find and select the “Enable WSDL Access” check box. This will allow the client to retrieve the Sentry WSDL for this service via URL. Click Save.





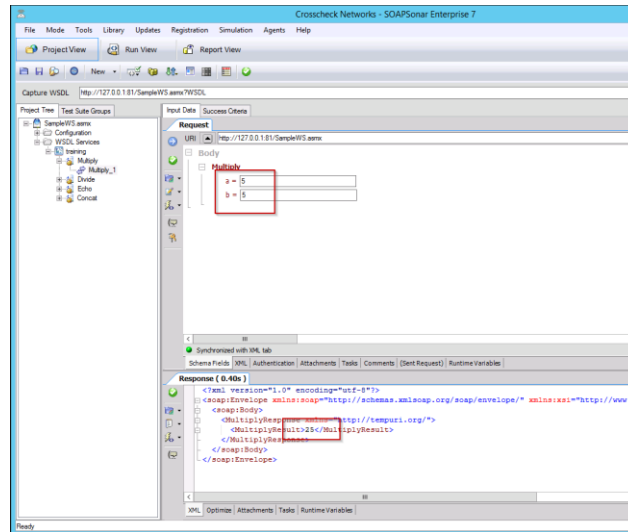
2. Load the Sentry WSDL into SOAPSonar.
 - a. Launch SOAPSonar
 - b. In the Capture WSDL field, enter the URL to the Sentry virtual directory and add ?WSDL to the end. This should work: <http://127.0.0.1:81/SampleWS.asmx?WSDL>



- c. Click Enter (or click the icon to the right of the bar) to retrieve and load the WSDL from Sentry into SOAPSonar.
- d. SOAPSonar will parse the WSDL and list out the 4 operations for this service (Multiply, Divide, Echo, Concat). SOAPSonar will also build sample test cases for each operation.
- e. Expand the 'Multiply' operation and click the test case named "Multiply_1". In the request pane, notice the URL is the Sentry Virtual Directory.



- f. Enter values in the Multiple 'a' and 'b' fields, then click  to commit the settings and  to send the request. The request will be processed by Sentry and then sent to the WebMatrix service. The response will be generated by WebMatrix and then returned to the client through Sentry.



g. You have successfully tested the WSDL policy in Sentry.

Transactions in the Sentry Logs

Now that you are able to send SOAP messages through Sentry, it is helpful to review a successful transaction at DEBUG level. The runtime traffic can be seen in both the Access and System log. Follow the steps below to enable DEBUG level logging and then review a successful transaction in the Sentry Access and System logs.

Follow the steps below to review the runtime transactions in the Sentry logs.

1. Navigate to the Diagnostics→Logging→Settings page
2. Set the log level for the Audit, System, and Access logs to DEBUG and click Save

3. Send another request from SOAPSonar through Sentry
4. Navigate to the Diagnostics→Logging→Internal Logs page
5. Open the Access Log for Today (hint, right click and open the log in another browser tab)

INTERNAL LOGS		
AUDIT LOGS		
Today	Download	(175B)
Sep 17, 2014	Download	(5.3KB)
Sep 4, 2014	Download	(1.6KB)
SYSTEM LOGS		
Today	Download	(1.1KB) X
Sep 17, 2014	Download	(6.5KB)
Sep 4, 2014	Download	(5.2KB)
ACCESS LOGS		
Today	Download	(318B) X
Sep 17, 2014	Download	(1.8KB)
Sep 4, 2014	Download	(1.6KB)

6. Notice that all of the previous runtime transactions are listed, one per line. The line shows the time of the request, the Session ID, the Client IP, the incoming host header, the HTTP method used, the URI (virtual directory triggered), the status code, and the request length for the transaction.

INTERNAL LOGS > ACCESS LOG

SEP 18, 2014

Search: Search

Refresh: (0-30 secs) Reset

Filter By Log Level: Debug

2 items found, displaying all items.1

Time	Session	IP	Host Header	Type	URI	Code	Length
00:13:25.267	X000004	127.0.0.1	127.0.0.1:81	POST	/SampleWS.asmx	200	346
00:01:38.573	X000003	127.0.0.1	127.0.0.1:81	POST	/SampleWS.asmx	500	814

- Click the most recent Session ID (at top by default) to jump to the System log.
- The System log will be sorted only showing log messages for this Session (transaction).

INTERNAL LOGS > SYSTEM LOG

SEP 18, 2014

Search: X000006 Search

Refresh: (0-30 secs) Reset

Filter By Log Level: Debug

Filter By Policy Name: All Policies

31 items found, displaying all items.1

ID	Time	Session	Code	Level	Message
0000F4	00:21:25.326	X000006	08402	D	Document left Communications Layer
0000F3	00:21:25.326	X000006	0840C	D	Sending client a raw response: Status Code: 200 Header Info: ...
0000F2	00:21:25.325	X000006	09334	D	Adding Via header to response

- The most recent log messages are listed at the top of the log by default. Scroll to the bottom to see the first log message for the transaction “Document entered communication layer” and read up from there to see all processing done in Sentry.

0000EB	00:21:25.324	X000006	09211	D	Received an HTTP response: Protocol: HTTP/1.1 Response Code: 200 ...
0000EA	00:21:25.320	X000006	0840B	D	Sending remote server a processed request: Method: POST Remot...
0000E9	00:21:25.319	X000006	09332	D	Adding Via header to request
0000E8	00:21:25.319	X000006	09330	D	Stored header suppressed from proxying - content-length: 377
0000E7	00:21:25.319	X000006	09330	D	Stored header suppressed from proxying - connection: keep-alive
0000E6	00:21:25.319	X000006	09330	D	Stored header suppressed from proxying - host: 127.0.0.1:81
0000E5	00:21:25.319	X000006	09003	D	Connecting to back end server at URL 'http://localhost:8080/SampleS.asmx'
0000E4	00:21:25.318	X000006	0E10B	D	No TaskListener configured, document will not be processed
0000E3	00:21:25.318	X000006	0E20A	D	Document left WSDL validation
0000E2	00:21:25.318	X000006	0E208	D	WSDL message: MultiplySoapIn
0000E1	00:21:25.318	X000006	0E221	D	ACL check skipped - no ACL associated with operation 'Multiply'.
0000E0	00:21:25.318	X000006	0E209	D	Document entered WSDL validation
0000DF	00:21:25.318	X000006	0E207	D	Matched WSDL operation 'Multiply(MultiplySoapIn)'
0000DE	00:21:25.317	X000006	08407	D	Request document: <?xml version="1.0" encoding="utf-8"?> <soap:Envelop...
0000DD	00:21:25.317	X000006	09604	D	Simple decode succeeded
0000DC	00:21:25.317	X000006	09607	D	Decoding a document of 377 bytes
0000DB	00:21:25.316	X000006	09410	D	Message type filter match succeeded - matched filter 'SOAP 1.1 Filter' of type Simple
0000DA	00:21:25.316	X000006	08016	D	ACL check skipped - no ACL associated with virtual directory '/SampleWS.asmx/'
0000D9	00:21:25.316	X000006	08016	D	ACL check skipped - no ACL associated with network policy 'SampleWS-Listener'
0000D8	00:21:25.316	X000006	0915C	D	Processing request for 'WSDL Policy: 'SampleWS''
0000D7	00:21:25.316	X000006	09140	D	Received an HTTP request: Protocol: HTTP/1.1 Scheme: http ...
0000D6	00:21:25.314	X000006	08401	D	Document entered Communications Layer

- Note that the “connecting to remote server” line is very important. This indicates that Sentry has successfully processed the request message and is now forwarding it to the remote server. If there are any errors returned for this transaction, the failure did not occur during request processing but rather either while connecting to the remote server, in processing on the remote server, or in the Sentry processing of the response document.

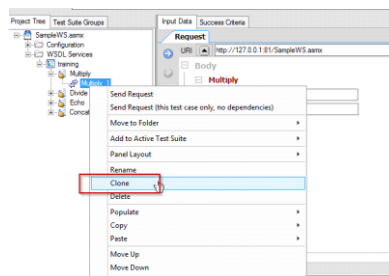
Test Sentry Schema Validation


Now that you are able to send valid SOAP messages through Sentry, it is helpful to send in an invalid request to ensure Sentry blocks the message – and see how it responds. One of the default “out of the box” IDP rules enabled in Sentry is the “Invalid WSDL Message” rule. This is triggered when a request or response document fails schema validation.

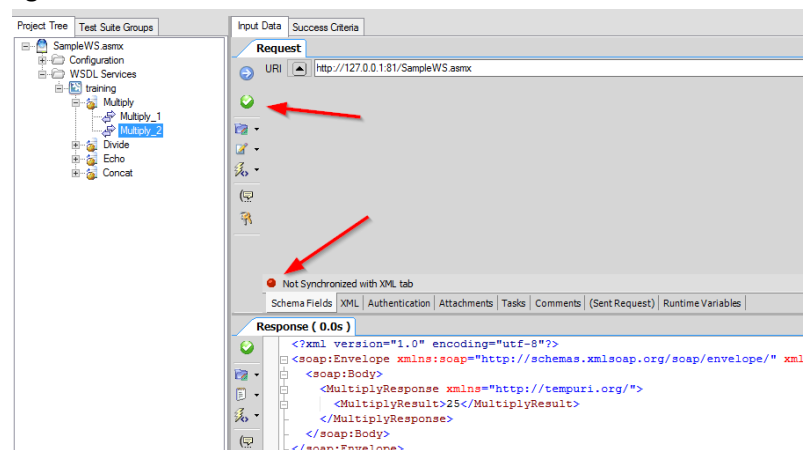
Because the WSDL for the service has been imported into Sentry, the product is capable of determining if the format of the SOAP message is valid. If it is not, the “Invalid WSDL Message” IDP rule is triggered and a SOAP fault is generated and returned to the client.

Follow the steps below to generate a request that should not pass any schema validation for the Sample Web Service.

1. In SOAPSonar, right click on the Multiply_1 test case and choose Clone. This will create a “cloned” copy of the first test case. We will modify this test case and test with it, so that the original working test case will continue to work.

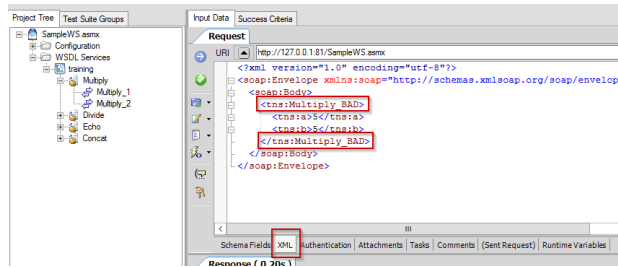




2. SOAPSonar has imported the WSDL from Sentry and so it knows how to build the SOAP requests correctly. The Schema Fields tab allows for easy input of request data, without having to edit XML directly. However, you can disable the Schema Fields tab and modify the XML data (and add custom headers) as necessary. To disable this, click the green “Synchronized with XML” indicator on the bottom of the pane (it should turn from green to red). Be sure to commit your settings with the . Click the XML tab to view the actual SOAP/XML data that SOAPSonar has generated.

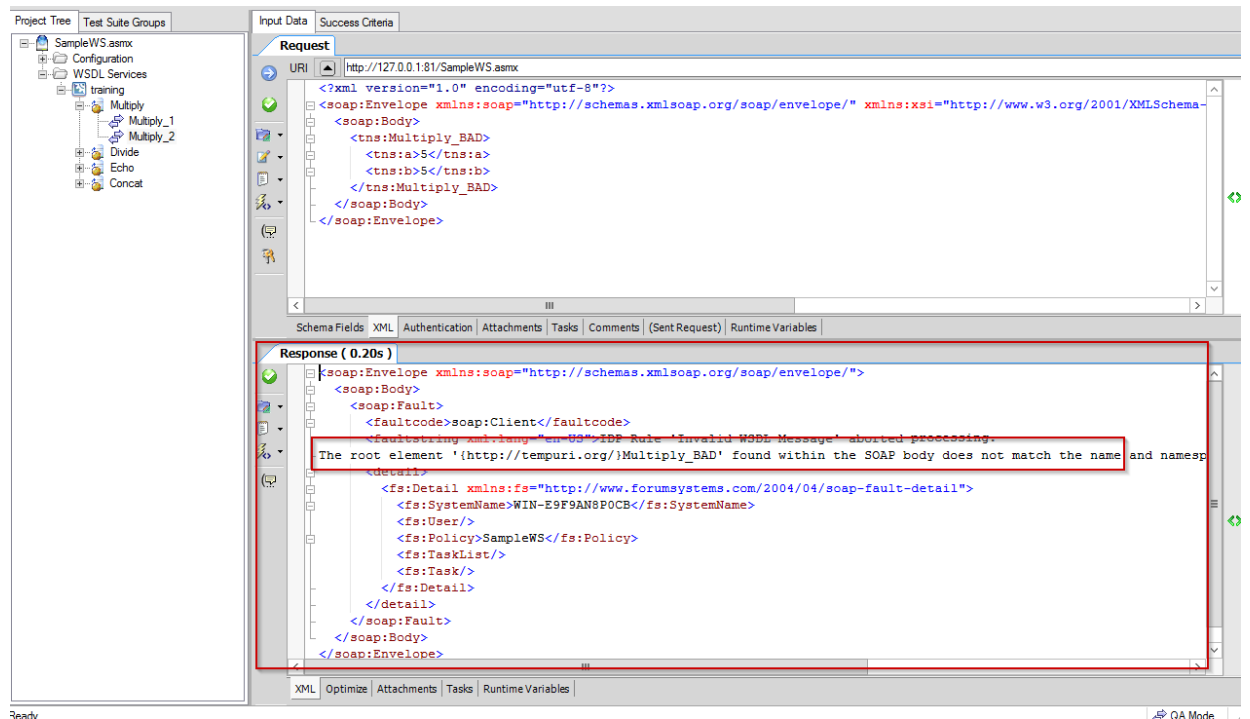


3. You can now view and edit the SOAP/XML data manually. Change the node “tns:Multiple” to “tns:Multiply_BAD” – note there are 2 edits required (open tag and close tag). While the new XML document is still valid XML/SOAP, it is not valid per the schema in the WSDL imported into

Sentry. Therefore Sentry should reject this request.



4. Click the  to commit the changes and send the request with . Notice the error response returned from Sentry. This request properly triggered the “Invalid WSDL Message” IDP rule and a SOAP fault was returned. Importantly, this request was never sent to the remote server. Sentry successfully blocked this malformed message thereby shielding the SOAP processor on the WebMatrix service.



- Now find this transaction in the Sentry log. You can find it through the Access Log and jump to the System log, or simply find it in the System log. The error will be easy to spot as the failure will be marked in yellow.

INTERNAL LOGS > SYSTEM LOG

SEP 18, 2014

Search:

Refresh: (0-30 secs)

Filter By Log Level:

Filter By Policy Name:

13 items found, displaying all items. 1

ID	Time	Session	Code	Level	Message
000106	00:29:04.208	X000007	08402	D	Document left Communications Layer
000105	00:29:04.208	X000007	0840E	D	Sending client an internally generated error response: Status Code: 500 ...
000104	00:29:04.208	X000007	0914D	D	Message type filter encode: document was generated locally; encoding with 'simple' format
000103	00:29:04.207	X000007	0600D	W	IDP Rule: 'Invalid WSDL Message', IDP Group 'Default WSDL Policy Group', Associated Policy: WSDL Policy: 'SampleWS', Triggered 1 time(s) on Request, Policy: SampleWS, Client IP: 127.0.0.1, User: -. The root element '{http://tempuri.org}Multiply_BAD' found within the SOAP body does not match the name and namespace of any message defined in the WSDL file.
000102	00:29:04.207	X000007	08407	D	Request document: <?xml version="1.0" encoding="utf-8"?> <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://tempuri.org/"> <soap:Body> <tns:Multiply_BAD> <tns:a>5</tns:a> <tns:b>5</tns:b> </tns:Multiply_BAD> </soap:Body> </soap:Envelope>
000101	00:29:04.207	X000007	09604	D	Simple decode succeeded

END

BACK IT UP!

It is recommended that you save your SOAPSonar project file AND export your full Sentry configuration after completing this lab.

To export your full Sentry configuration, navigate to the System→Configuration→Import/Export screen and use the Export option in the center of the page to export the full Sentry configuration file as a password encrypted FSX file. This can later be imported on the same screen.

We recommend including the lab number in the name of the export files.

Additional Tests and Discussion Topics

1. After deploying the SOAP service through Sentry and testing it successfully, try sending different malformed requests into Sentry to see how it responds.
2. Do some get through that should get blocked? If so the security parameters in Sentry likely need to be adjusted.
3. For testing, try changing the HTTP Method, the HTTP Content-Type header, the message body, etc... and see what happens when Sentry processes the request.
4. What happens if you send a request into the wrong URI?
5. What happens if you try to use a browser to access the service through Sentry?
6. Go to the Gateway→Network Policies→Network Policies page to review the new HTTP Listener and Remote Policies created when building the WSDL policy.
7. Understanding the Sentry logging is very important for both Sentry administrators and Sentry developers.
8. Understanding the basics of a WSDL file and SOAP may be beneficial to Sentry developers.

Additional Information

For more information, review the following Forum Sentry Admin Guides:

1. Network Policies Guide
2. WSDL Policies Guide
3. IDP Guide

About Forum Systems

Forum Systems is the global leader in API and Cloud Security technology with industry-certified, patented, and proven products deployed in the most rigorous and demanding customer environments worldwide. Forum Systems has been an industry leader for over 12 years and has built the core architecture of its technology on the foundation of FIPS 140-2 and NDPP. Forum Systems security-first mindset enables trusted, network edge deployments of its technology for protecting critical enterprise transactions.

Our product technology is purpose-built and designed for mission-critical, enterprise-class scalable solutions where business solutions require the modern day security and identity enforcement protection, while enabling a scalable architecture and low-latency, high-volume throughput.

Forum Systems supports global enterprise customers across industries in commercial, government, and military sectors. Forum Systems technology provides the leading-edge of modern-day cyber-security innovation with integrated identity and SSO features that enable out-of-the box business solutions with point-and-click technology.

Forum's patented; FIPS 140-2 and NDPP certified hardware and virtual products make modern-day business communications secure by actively protecting and accelerating data exchange and API service access across networks and business boundaries. For more information, please visit www.forumsys.com.