



# WSO2 API Manager 4.2.0 Advanced - Integration Profile

Troubleshooting



WSO2 Training

## Module Objective

At the end of this module, attendees will be able to:

- Understand the basics of logs in the Micro Integrator.
- Learn how to monitor messages.
- Work with monitoring tools in the Micro Integrator.
- Understand how to handle errors in the Micro Integrator.
- Learn how to troubleshoot timeout issues.
- Learn best practices associated with troubleshooting and monitoring.



## General Guidelines

- Investigate to identify the problem.
- Based on where the problem is, you can perform any of the following:
  - Debug configurations.
  - Debug network connectivity.
  - Debug the product instance.
  - Monitoring messages.



# Debugging Configurations

To debug configurations, you can use:

- Logs
  - ⦿ Log Mediator
  - ⦿ Log4j2 Logs
  - ⦿ Access Logs
- Mediation Debugger

## Debugging Configurations - Log Mediator

- **Log** mediator can be used to log specific messages during mediation flow.

```
<log [level="string"] [separator="string"]>  
  <property name="string" (value="literal" | expression="xpath")/>*</log>
```

- There are four levels of logging:
  - Simple
  - Headers
  - Full
  - Custom

## Debugging Configurations - Log4j2 Logs

- Logs are saved in the `<MI_HOME>/repository/logs` directory.
- Types of log files:
  - **HTTP Access logs:** records all http requests.
  - **Carbon logs:** application and runtime specific information.
  - **API logs:** information specific to deployed APIs.
  - **Service logs:** information specific to deployed services.
  - **Trace logs:** tracing information about services.
  - **Error logs:** records errors occurred in WSO2 Micro Integrator
  - **Audit logs:** records actions carried out on the server.
- Do not save sensitive information such as passwords, social security numbers, and credit card numbers as plain text in log files.



## Debugging Configurations - Logs

|              |  |
|--------------|--|
| <b>OFF</b>   | The highest possible log level. Intended for disabling logging.  |
| <b>FATAL</b> | Indicates server errors that cause premature termination.  |
| <b>ERROR</b> | Indicates other runtime errors or unexpected conditions.   |
| <b>WARN</b>  | Indicates the use of deprecated APIs, poor use of API, possible errors, and other runtime situations that are undesirable or unexpected but not necessarily wrong. |
| <b>INFO</b>  | Indicates important runtime events: server startup/shutdown.   |
| <b>DEBUG</b> | Detailed information on the flow through the system.   |
| <b>TRACE</b> | Additional details on the behavior of events and services.   |

## Debugging Configurations - Changing Log Level

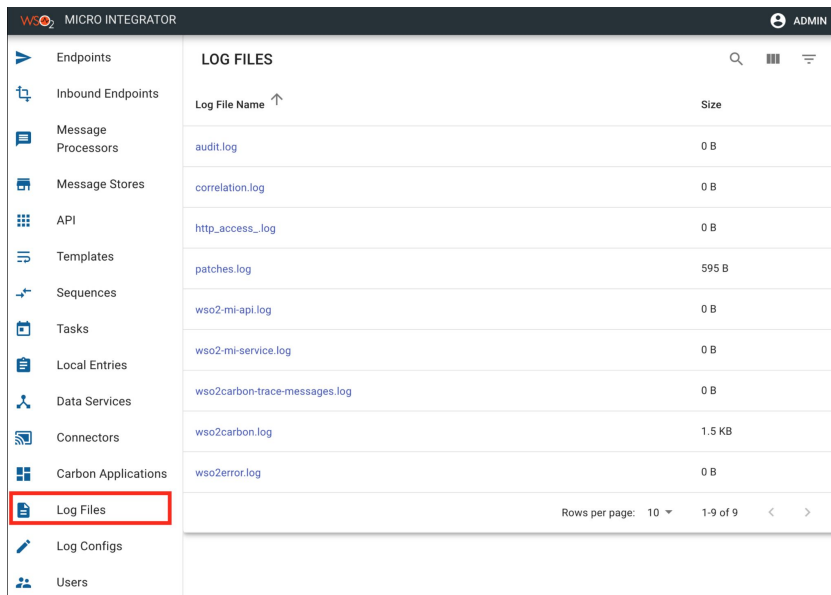
|                         |   |
|-------------------------|---|
| <b>INFO &gt; DEBUG</b>  | Changing the level from INFO to DEBUG can increase resource usage because all stack traces as well as information useful for debugging the application can appear in the log. |
| <b>WARN &gt; INFO</b>   | Changing from WARN to INFO can increase the log file size because WARN level is only displayed when a warning should be given.  |
| <b>ERROR &gt; WARN</b>  | Changing from ERROR to WARN can result in a slight increase in the log file size.   |
| <b>FATAL &gt; ERROR</b> | Changing from FATAL to ERROR can result in a slight increase in the log file size.  |





# Dynamically Changing Log Level

To dynamically change the log levels for various components, use the Micro Integrator **Dashboard** or the **CLI**.



The screenshot displays the WSO2 Micro Integrator Dashboard. On the left is a navigation sidebar with various menu items. The 'Log Files' item is highlighted with a red rectangular box. The main content area is titled 'LOG FILES' and contains a table with two columns: 'Log File Name' and 'Size'. The table lists several log files, including 'audit.log', 'correlation.log', 'http\_access\_log', 'patches.log', 'wso2-mi-api.log', 'wso2-mi-service.log', 'wso2carbon-trace-messages.log', 'wso2carbon.log', and 'wso2error.log'. At the bottom right of the table, there is a pagination control showing 'Rows per page: 10' and '1-9 of 9'.

| Log File Name                 | Size   |
|-------------------------------|--------|
| audit.log                     | 0 B    |
| correlation.log               | 0 B    |
| http_access_log               | 0 B    |
| patches.log                   | 595 B  |
| wso2-mi-api.log               | 0 B    |
| wso2-mi-service.log           | 0 B    |
| wso2carbon-trace-messages.log | 0 B    |
| wso2carbon.log                | 1.5 KB |
| wso2error.log                 | 0 B    |

## Debugging Configurations - Debug Logs

Prints useful information related to the mediation flow inside the Micro Integrator.

```
[2021-07-27 15:42:41,540] INFO {EventAdminConfigurationNotifier} - Logging configuration changed. (Event Admin service unavailable - no notification sent)
[2021-07-27 15:42:44,064] INFO {API} - {api:HelloWorld} Initializing API: HelloWorld
[2021-07-27 15:42:44,073] INFO {APIDeployer} - API named 'HelloWorld' has been deployed from file : /Applications/IntegrationStudio.app/Contents/Eclipse/
[2021-07-27 15:42:44,076] INFO {CappDeployer} - Successfully Deployed Carbon Application : helloworldCompositeExporter_1.0.0-SNAPSHOT{super-tenant}
[2021-07-27 15:42:44,079] INFO {AppDeployerServiceComponent} - Dashboard is configured. Initiating heartbeat component.
[2021-07-27 15:42:44,122] INFO {PassThroughListeningIOReactorManager} - Pass-through HTTP Listener started on 0.0.0.0:8290
[2021-07-27 15:42:44,125] INFO {PassThroughListeningIOReactorManager} - Pass-through HTTPS Listener started on 0.0.0.0:8253
[2021-07-27 15:42:44,126] INFO {StartupFinalizer} - WS02 Micro Integrator started in 4.11 seconds
[2021-07-27 15:42:44,233] INFO {PassThroughListeningIOReactorManager} - Pass-through EI_INTERNAL_HTTP_INBOUND_ENDPOINT Listener started on 0.0.0.0:9201
[2021-07-27 15:42:44,246] INFO {PassThroughListeningIOReactorManager} - Pass-through EI_INTERNAL_HTTPS_INBOUND_ENDPOINT Listener started on 0.0.0.0:9164
[2021-07-27 15:42:46,711] INFO {AuthenticationHandlerAdapter} - User admin logged in successfully
[2021-07-27 15:42:46,720] INFO {ServiceComponent} - Initializing Security parameters
```

## Debugging Configurations - Debug Logs

- To enable debug logs dynamically,
  - a. Append *org-apache* to the loggers section in the *log4j2.properties* file.
  - b. Use the **Micro Integrator Dashboard** or **CLI**.
- Alternatively:
  - a. Shut down the Micro Integrator.
  - b. Open the *log4j2.properties* file (located in <MI\_HOME>/conf)
    - i. Change the logger for logger.org-apache as follows:

```
logger.org-apache.level = DEBUG  
logger.org-apache.additivity = true
```
    - ii. Append *org-apache* to the loggers section.
  - c. Start the Micro Integrator.



## Debugging Configurations - Trace Logs

Trace logs trace the entire path when a message travels along a mediation sequence.

```
14:32:40,548 [-] [http-nio-9443-exec-39] INFO TRACE_LOGGER Building Axis service for Proxy service : SplitAggregateProxy
14:32:40,551 [-] [http-nio-9443-exec-39] INFO TRACE_LOGGER Loading the WSDL : <Inlined>
14:32:40,552 [-] [http-nio-9443-exec-39] INFO TRACE_LOGGER Did not find a WSDL. Assuming a POX or Legacy service
14:32:40,552 [-] [http-nio-9443-exec-39] INFO TRACE_LOGGER Exposing transports : [http, https]
14:32:40,552 [-] [http-nio-9443-exec-39] INFO TRACE_LOGGER Adding service SplitAggregateProxy to the Axis2 configuration
14:32:40,554 [-] [http-nio-9443-exec-39] INFO TRACE_LOGGER Successfully created the Axis2 service for Proxy service : SplitAggregateProxy
14:33:57,890 [-] [PassThroughMessageProcessor-8] INFO TRACE_LOGGER Proxy Service SplitAggregateProxy received a new message from :
127.0.0.1
14:33:57,891 [-] [PassThroughMessageProcessor-8] INFO TRACE_LOGGER Message To:
/services/SplitAggregateProxy.SplitAggregateProxyHttpSoap11Endpoint
14:33:57,891 [-] [PassThroughMessageProcessor-8] INFO TRACE_LOGGER SOAPAction: urn:mediate
14:33:57,891 [-] [PassThroughMessageProcessor-8] INFO TRACE_LOGGER WSA-Action: urn:mediate
14:33:57,892 [-] [PassThroughMessageProcessor-8] INFO TRACE_LOGGER Setting default fault-sequence for proxy
14:33:57,892 [-] [PassThroughMessageProcessor-8] INFO TRACE_LOGGER Using the anonymous in-sequence of the proxy service for mediation
14:33:57,927 [-] [PassThroughMessageProcessor-9] INFO TRACE_LOGGER Setting default fault-sequence for proxy
```

## Debugging Configurations - Trace Logs

### Enabling Trace logs:

- Tracing can be enabled for a proxy or for a sequence. This can be done by adding the following attribute for the proxy/sequence configuration:

```
trace="enable"
```

- When tracing is enabled, trace logs can be seen from the *wso2-ei-trace.log* file.
- This will provide more fine-grained information than the debug logs.

## Debugging Configurations - Wire Logs

- Monitor messages flowing through the Micro Integrator over the HTTP transport.
- Advantages over the TCPMon tool:
  - Monitor messages going through the HTTPS transport.
  - Do not need to change the synapse configuration.

## Debugging Configurations - Wire Logs

- To enable wire logs dynamically, use the **Micro Integrator Dashboard** or **CLI**.
- Alternatively:
  - a. Shut down the Micro Integrator.
  - b. Open the **log4j2.properties** file (located in `<API-M_HOME>/repository/conf`).
  - c. 

```
logger.synapse-transport-http-wire.name=org.apache.synapse.transport.http.wire  
logger.synapse-transport-http-wire.level=DEBUG
```
  - d. Start the Micro Integrator.



# Wirelogs

# Wirelogs

```

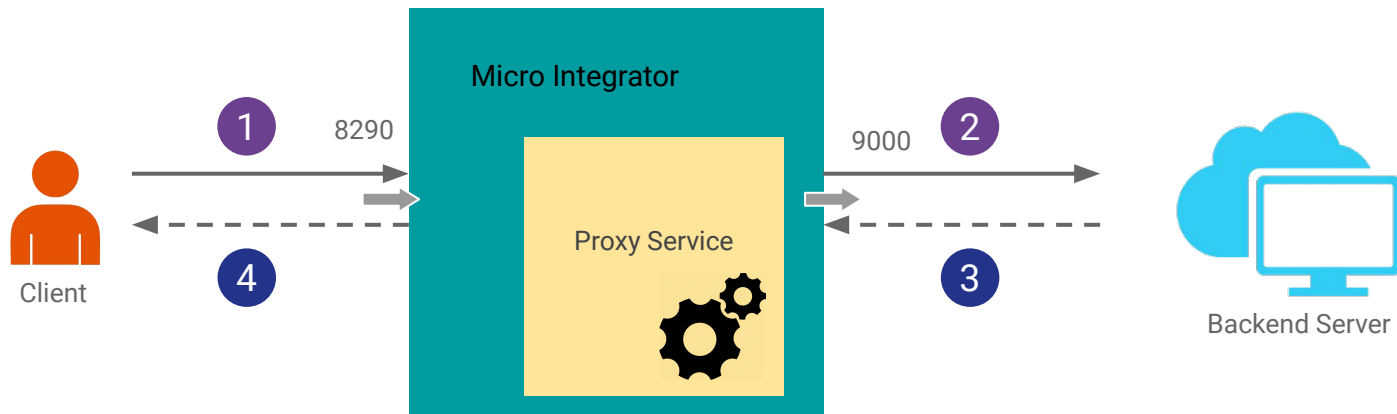
[2018-04-20 13:47:40,144] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "POST /services/StockQuoteProxy/StockQuoteProxyHttpSoap1.1Endpoint HTTP/1.1[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "Accept-Encoding: gzip,deflate[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "Content-Type: text/xml; charset=UTF-8[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "SOAPAction: \"urn:mediate\"[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "Content-Length: 280[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "Host: ws02123-ThinkPad-X1-Carbon-3rd8280[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "Connection: Keep-Alive[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "User-Agent: Apache/Http-Client/4.1.1 (Java/1.5)[rl]n"
[2018-04-20 13:47:40,145] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "<?xml version='1.0' encoding='UTF-8'?><soap:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "  <soapenv:Header><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "    <[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "      <m:getQuote xmlns:m='http://services.samples'><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "        <m:request><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "          <m:symbol>IBM</m:symbol><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "        </m:request><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "      </m:getQuote><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "    </soapenv:Body><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "  </soapenv:Envelope><[rl]n"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "</?>"
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "10 seconds, irrespective of the timeout action, after the specified or optional timeout
[2018-04-20 13:47:40,146] [E]-Core DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "IN PROGRESS: This engine will stop all calls after the GLOBAL TIMEOUT: 10 seconds, irrespective of the timeout action, after the specified or optional timeout
[2018-04-20 13:47:40,212] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "POST /services/StockQuoteProxy/StockQuoteProxyHttpSoap1.1[rl]n"
[2018-04-20 13:47:40,212] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "Accept-Encoding: gzip,deflate[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "Content-Type: text/xml; charset=UTF-8[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "SOAPAction: \"urn:mediate\"[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "Transfer-Encoding: chunked[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "Host: localhost:9000[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "Connection: Keep-Alive[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "User-Agent: Synapse-PT-HttpComponents-NT[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "[rl]n"
[2018-04-20 13:47:40,213] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "<?xml version='1.0' encoding='UTF-8'?><soap:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "  <soapenv:Header><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "    <soapenv:Body><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "      <m:getQuote xmlns:m='http://services.samples'><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "        <m:request><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "          <m:symbol>IBM</m:symbol><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "        </m:request><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "      </m:getQuote><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "    </soapenv:Body><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "  </soapenv:Envelope><[rl]n"
[2018-04-20 13:47:40,214] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "</?>"
[2018-04-20 13:47:40,215] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 << "[rl]n"
[2018-04-20 13:47:40,217] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "HTTP/1.1 200 OK[rl]n"
[2018-04-20 13:47:40,217] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "Content-Type: text/xml; charset=UTF-8[rl]n"
[2018-04-20 13:47:40,217] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "Date: Fri, 20 Apr 2018 08:17:40 GMT[rl]n"
[2018-04-20 13:47:40,218] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "Transfer-Encoding: chunked[rl]n"
[2018-04-20 13:47:40,218] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "Connection: Keep-Alive[rl]n"
[2018-04-20 13:47:40,218] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "[rl]n"
[2018-04-20 13:47:40,225] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "<?xml version='1.0' encoding='UTF-8'?><soap:Envelope xmlns:soapenv='http://schemas.xmlsoap.org/soap/envelope/'><soapenv:Body><ns:getQuoteResponse xmlns:ns='http://services.samples'><ns:return
xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance' xsi:type='xsd:string'><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:Symbol>IBM</ns:Symbol><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:Volume>7769</ns:Volume><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:LastTradeTimestamp>Fri Apr 20 13:47:40 IST<[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:MarketCap>1.4375117912060674E7</ns:MarketCap><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:Name>IBM<[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:PercentageChange>0.21</ns:PercentageChange><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:PreviousClose>186.18701603846165</ns:PreviousClose><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:Symbol>IBM</ns:Symbol><[rl]n"
[2018-04-20 13:47:40,226] [E]-Core DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "  <ns:Volume>7769</ns:Volume><[rl]n"
[2018-04-20 13:47:4
```



## Debugging Configurations - Wire Logs

To read a wire log, we first have to identify the message direction.

- **DEBUG - wire >>** - Represent the message coming into MI from the wire.
- **DEBUG - wire <<** - Represent the message going into the wire from the MI.



## Debugging Configurations - Wire Logs

```
[2018-04-20 13:47:40,144] [EI-Core] DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "POST  
/services/StockQuoteProxy.StockQuoteProxyHttpSoap11Endpoint HTTP/1.1[r][\n]"
```

```
[2018-04-20 13:47:40,145] [EI-Core] DEBUG - wire HTTP-Listener I/O dispatcher-1 >> "Content-Type: text/xml; charset=UTF-8[r][\n]"
```

```
[2018-04-20 13:47:40,212] [EI-Core] DEBUG - wire HTTP-Sender I/O dispatcher-1 << "POST /services/SimpleStockQuoteService HTTP/1.1[r][\n]"
```

```
[2018-04-20 13:47:40,213] [EI-Core] DEBUG - wire HTTP-Sender I/O dispatcher-1 << "Content-Type: text/xml; charset=UTF-8[r][\n]"
```

```
[2018-04-20 13:47:40,217] [EI-Core] DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "HTTP/1.1 200 OK[r][\n]"
```

```
[2018-04-20 13:47:40,217] [EI-Core] DEBUG - wire HTTP-Sender I/O dispatcher-1 >> "Content-Type: text/xml; charset=UTF-8[r][\n]"
```

```
[2018-04-20 13:47:40,235] [EI-Core] DEBUG - wire HTTP-Listener I/O dispatcher-1 << "HTTP/1.1 200 OK[r][\n]"
```

```
[2018-04-20 13:47:40,239] [EI-Core] DEBUG - wire HTTP-Listener I/O dispatcher-1 << "Content-Type: text/xml; charset=UTF-8[r][\n]"
```

# Debugging Configurations - Mediation Debugger

Debug message mediation flows.

The screenshot displays the Mediation Debugger interface with the following components:

- Top Left (Servers):** Shows the Micro Integrator Server (Generic Server) running at `/Applications/DeveloperStudio.app/Contents/Eclipse/jdk-home/Contents/Home/bin/java` (Apr 3, 2019, 3:35:11 PM).
- Top Right (Variables):** A table showing the current state of various scopes.

| Name                          | Value   |
|-------------------------------|---|
| Axis2 Scope Properties        | ("To":"/services/SimpleStockQuoteServiceProxy;SimpleStockQuoteServicePr...    |
| Axis2-Client Scope Properties | 0   |
| Transport Scope Properties    | ("Accept-Encoding":"gzip,deflate","Connection":"Keep-Alive","Content-Lengt... |
| Operation Scope Properties    | 0   |
| Synapse Scope Properties      | ("IsClientDoingSOAP11":true,"TRANSPORT_IN_NAME":"http","proxy.name":"...      |
- Center (Design View):** A mediation flow diagram showing the sequence of steps: `Property` (red box) → `log payload` → `Call` (blue box) → `Named` (purple box) → `log response` → `respond` (blue box).
- Bottom Left (Console):** Displays the XML message being processed.

```
<?xml version="1.0" encoding="UTF-16"?>
<soapenv:Envelope xmlns:sap="http://services.samples"
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://services.samples/xsd">
  <soapenv:Body>
    <ser:getQuote>
      <!--Optional-->
      <ser:request>
        <!--Optional-->
        <xsd:symbolWSO2<xsd:symbol>
      </ser:request>
    </ser:getQuote>
  </soapenv:Body>
</soapenv:Envelope>
```
- Bottom Right (Property Key):** A table showing the properties of the message.

| Property Key    | Property Value                       |
|-----------------|--------------------------------------|
| To              | /services/SimpleStockQuoteService... |
| From            |                                      |
| Content-Length  | 417                                  |
| Accept-Encoding | gzip,deflate                         |
| Connection      | Keep-Alive                           |
| Content-Type    | text/xml; charset=UTF-8              |
| Host            | Hasithas-MacBook-Pro.local:8290      |
| SOAPAction      | "urn:getQuote"                       |

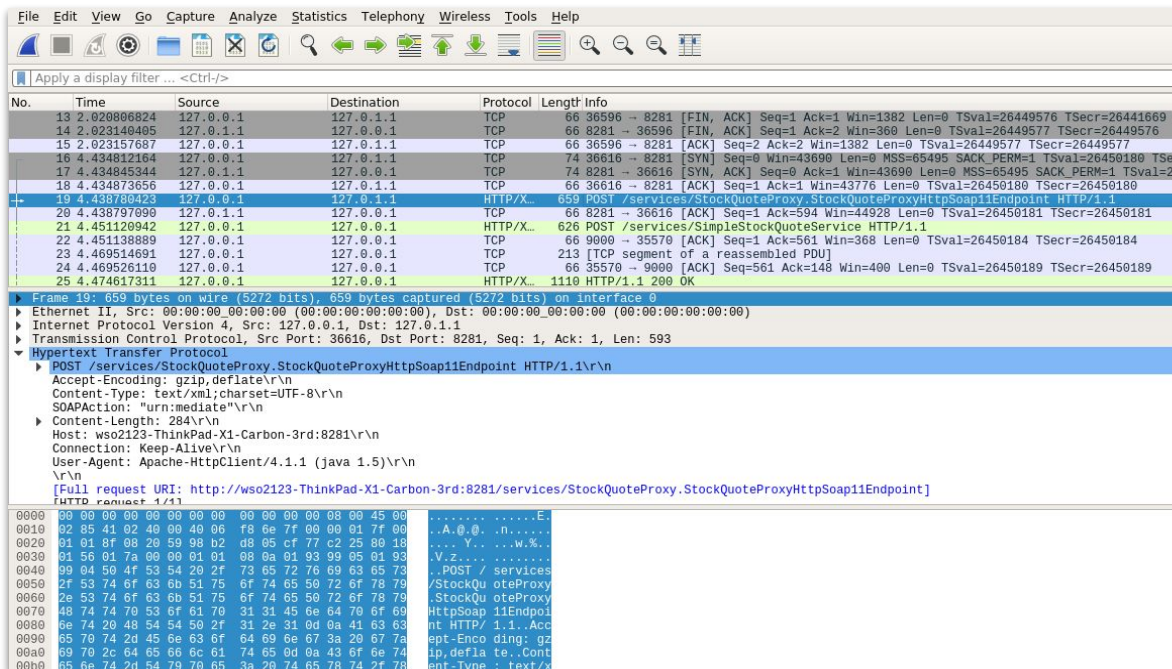


## Debug Network Connectivity - Wireshark

- Wireshark is a network protocol analyzer.
- Captures packets in real time and displays them in human-readable format.
- Packet capture can provide information about individual packets such as transmit time, source, destination, protocol type, and header data.
- Can monitor both HTTP and HTTPS requests.
- No configuration changes needed.

# Debug Network Connectivity - Wireshark

## Capture data from request.



The screenshot displays the Wireshark interface with a network traffic capture. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for file operations, capture control, and analysis. A display filter bar shows 'Apply a display filter ... <Ctrl-/>'. The main packet list pane shows a list of captured packets with columns for No., Time, Source, Destination, Protocol, and Length. Packet 19 is selected, showing details for an HTTP POST request to /services/StockQuoteProxy/HttpSoap11Endpoint. The packet details pane shows the request structure, including headers like Accept-Encoding, Content-Type, SOAPAction, Content-Length, Host, Connection, and User-Agent. The packet bytes pane shows the raw data in hexadecimal and ASCII.

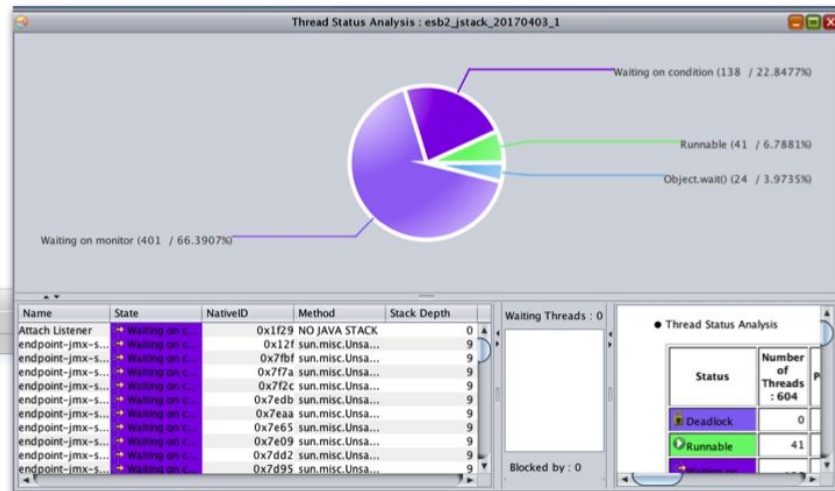
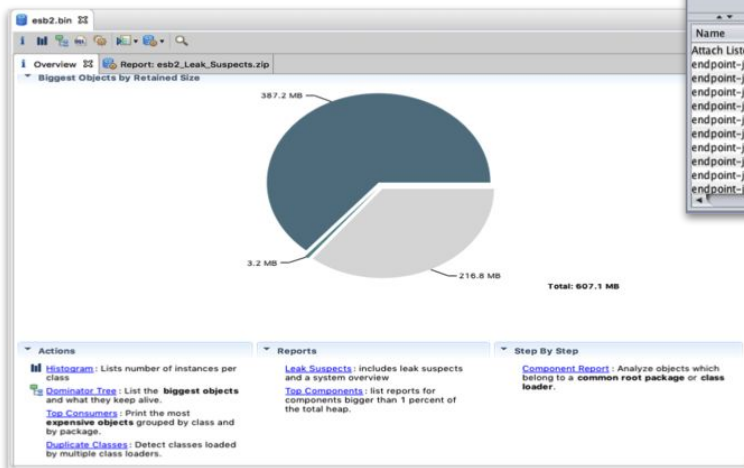
| No. | Time        | Source    | Destination | Protocol | Length | Info  |
|-----|-------------|-----------|-------------|----------|--------|---|
| 13  | 2.020806824 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 36596 → 8281 [FIN, ACK] Seq=1 Ack=1 Win=1382 Len=0 TSval=26449576 TSecr=26441669                        |
| 14  | 2.023140405 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 8281 → 36596 [FIN, ACK] Seq=1 Ack=2 Win=360 Len=0 TSval=26449577 TSecr=26449576                         |
| 15  | 2.023157687 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 36596 → 8281 [ACK] Seq=2 Ack=2 Win=1382 Len=0 TSval=26449577 TSecr=26449577                             |
| 16  | 4.434812164 | 127.0.0.1 | 127.0.0.1   | TCP      | 74     | 36616 → 8281 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=26450180 TSecr=26450180            |
| 17  | 4.434845344 | 127.0.0.1 | 127.0.0.1   | TCP      | 74     | 8281 → 36616 [SYN, ACK] Seq=0 Ack=1 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=26450180 TSecr=26450180 |
| 18  | 4.434873056 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 36616 → 8281 [ACK] Seq=1 Ack=1 Win=43776 Len=0 TSval=26450180 TSecr=26450180                            |
| 19  | 4.43488023  | 127.0.0.1 | 127.0.0.1   | HTTP/X.. | 659    | POST /services/StockQuoteProxy/HttpSoap11Endpoint HTTP/1.1  |
| 20  | 4.438797099 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 8281 → 36616 [ACK] Seq=1 Ack=594 Win=44928 Len=0 TSval=26450181 TSecr=26450181                          |
| 21  | 4.451120942 | 127.0.0.1 | 127.0.0.1   | HTTP/X.. | 626    | POST /services/SimpleStockQuoteService HTTP/1.1   |
| 22  | 4.451138889 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 9000 → 35570 [ACK] Seq=1 Ack=561 Win=368 Len=0 TSval=26450184 TSecr=26450184                            |
| 23  | 4.469514691 | 127.0.0.1 | 127.0.0.1   | TCP      | 213    | [TCP segment of a reassembled PDU]  |
| 24  | 4.469526110 | 127.0.0.1 | 127.0.0.1   | TCP      | 66     | 35570 → 9000 [ACK] Seq=561 Ack=148 Win=400 Len=0 TSval=26450189 TSecr=26450189                          |
| 25  | 4.474617311 | 127.0.0.1 | 127.0.0.1   | HTTP/X.. | 1110   | HTTP/1.1 200 OK   |

Packet 19 details:

- POST /services/StockQuoteProxy/HttpSoap11Endpoint HTTP/1.1\r\n
- Accept-Encoding: gzip,deflate\r\n
- Content-Type: text/xml;charset=UTF-8\r\n
- SOAPAction: "urn:mediate"\r\n
- Content-Length: 284\r\n
- Host: wso2123-ThinkPad-X1-Carbon-3rd:8281\r\n
- Connection: Keep-Alive\r\n
- User-Agent: Apache-HttpClient/4.1.1 (java 1.5)\r\n
- \r\n
- [Full request URI: http://wso2123-ThinkPad-X1-Carbon-3rd:8281/services/StockQuoteProxy/HttpSoap11Endpoint]
- [HTTP request 1/1]

# Debugging the Product Instance

- Analyze logs
- Analyze thread dumps
- Analyze heap dumps



# Monitoring Messages

Why message monitoring?

- Message payload sent by the Micro Integrator is not in the format that is expected by the back-end service.
- The content type of the sent message is not supported by the back-end service.
- Some of the required headers, such as 'Authorization Header', are missing in the request sent from the Micro Integrator.



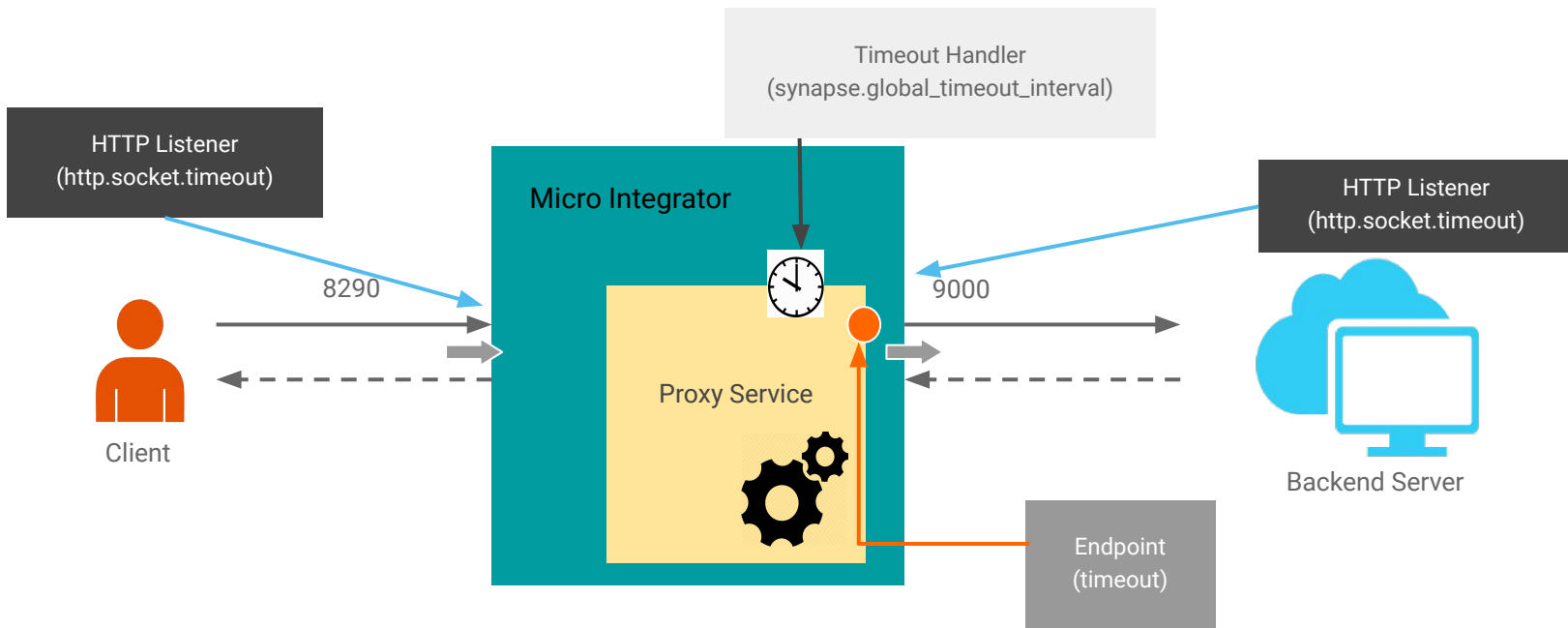
# Monitoring Messages

How to monitor messages?

- WireLogs
- Wireshark



# Troubleshooting Timeout Issues



# Troubleshooting Timeout Issues

## Client-to-Proxy Connection Timeout Parameters

### **http.socket.timeout**

- In the client-to-proxy connection, only one configuration parameter needs to be configured (`http.socket.timeout`).
- This can be configured from the *passthru-http.properties* file.
- This represents the socket timeout value of the passthrough http/https transport listener.
- The default value is 180000.



# Troubleshooting Timeout Issues

## Proxy-to-Backend Connection Timeout Parameters

### **http.socket.timeout**

- This represents the socket timeout value of the passthrough http/https transport sender. This is the same parameter used in the transport listener.

### **Endpoint timeout**

- Timeout config parameter that can be configured at the endpoint level.
- It allows us to configure different timeout values for different endpoints.
- For endpoints that don't have a timeout configuration, the global parameter value is considered as the timeout duration.



# Troubleshooting Timeout Issues

## Proxy-to-Backend Connection Timeout Parameters

### **`synapse.global_timeout_interval`**

- Synapse is a complete asynchronous engine, which does not block worker threads on network IO. It registers a callback and return.
- When the response is received, the registered callback is used to correlate it with the request and further processing is done. If the back-end server does not respond, it is required to clear the registered callbacks.
- Done by TimeoutHandler. The 'synapse.global\_timeout\_interval' parameter denotes for how long a callback should be kept in the callback store.



# Troubleshooting Timeout Issues

## Proxy-to-Backend Connection Timeout Parameters

- Use the following formula:

`Socket Timeout > max(Global endpoint timeout, Timeout of individual endpoints)`

- Set `http.socket.timeout` to a value higher than all other endpoint timeout values.

