|  |  |
| --- | --- |
|  | |
| Lightweight Integration LLD | |
| Version: v0.1  Date: 2016.08.11.  Author: Paróczi Zsolt | ENGINEER_klein |

# Modifications

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Modifier** | **Modification** |
| 2016.08.11 | v0.1 | Paróczi Zsolt | Initial version |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Table of content

[Modifications 2](#_Toc456096827)

[Table of content 3](#_Toc456096828)

[1 Introduction 4](#_Toc456096829)

[1.1 Lightweight integration summary 4](#_Toc456096830)

[1.2 Scope of the document 4](#_Toc456096831)

[1.3 Referenced documents 4](#_Toc456096832)

[1.4 Aim of the document 4](#_Toc456096833)

[1.5 Terminologies and acronyms 4](#_Toc456096834)

[2 High level requirements 6](#_Toc456096835)

[3 Design 7](#_Toc456096836)

[3.1 Architectural design 7](#_Toc456096837)

[3.1.1 System architecture 7](#_Toc456096838)

[3.1.2 Component structure 7](#_Toc456096839)

[3.1.2.1 Proxy 8](#_Toc456096840)

[3.1.3 Core 9](#_Toc456096841)

[3.2 Functional design 10](#_Toc456096842)

[3.2.1 Proxy layer 10](#_Toc456096843)

[3.2.1.1 HTTPS termination 10](#_Toc456096844)

[3.2.1.2 Load balancing 11](#_Toc456096845)

[3.2.1.3 Client username forward from client cert 11](#_Toc456096846)

[3.2.2 Core layer 11](#_Toc456096847)

[3.2.2.1 Configuration 12](#_Toc456096848)

[3.2.2.2 Routig 13](#_Toc456096849)

[3.2.2.3 Authentication and authorization 13](#_Toc456096850)

[3.2.2.4 Message logging 15](#_Toc456096851)

[3.2.2.5 Message validation 16](#_Toc456096852)

[3.2.2.6 Message throttling 16](#_Toc456096853)

[3.2.2.7 xrt 16](#_Toc456096854)

[4 Appendixes 17](#_Toc456096855)

[4.1 Security configuration schema 17](#_Toc456096856)

[4.2 Configuration schema 17](#_Toc456096857)

[4.3 Configuration sample 17](#_Toc456096858)

# Introduction

## Lightweight integration summary

There is a need for an alternate lightweight integration solution next to the heavyweight MT OSB standard integration.

The key feature of the new integration solution is the quick development free integration setup.

The new solution is selected for integration in the following circumstances:

* the published service not enterprise service
* API publication is the aim
* the integration is definitely 1:1

The Lightweight Integration solution is based on the HaProxy open source robust proxy application and the Wildfly(JBoss)’s Undertow servlet container. The HaProxy is responsible for the fail over and the load balancing in front of the integration and in the front of the applications. The customized – built in and developed plugins - Undertow module is responsible for the integration requirements.

## Scope of the document

Present document contains the implementation of the Lightweight Integration solution according to the functional specification (HLD).

## Referenced documents

* Lighweight\_Integration\_v7.docx ( requirements )
* Lightweight\_Integration\_HLD\_v1\_1.docx

## Aim of the document

The aim of the present high level document is to describe the implementation of the Lightweight Integration solution according to the high level design document.

## Terminologies and acronyms

|  |  |
| --- | --- |
| **Acronym** | **Description** |
|  |  |
| ESB | Enterprise Service Bus |
| HaProxy | Opensource proxy software solution |
| Lightweight | Simple way |
| OSB | Oracle ESB implementation |
| Undertow | Wildfly(JBoss) servlet container |
|  |  |
|  |  |
|  |  |

# Design

## Architectural design

### System architecture

The lightweight integration architecture looks like the following.



|  |  |  |
| --- | --- | --- |
| **Component** | **Solution** | **Role** |
| Lwi Proxy | Lightweight integration proxy. Based on a HaProxy application. | Responsible for the load balancing and the failover further the https termination and the client certificate attribute as username passing in the header. |
| Lwi Core | Wildfly Undertow customized webserver/servlet container | Responsible for the integrational requirements; log, authentication, authorization, message validation, throttling and routing. |

*Application zone components:*

The several backend applications are accessed from the LightWeight Integration through a proxy layer, therefore the LightWeight Integration has no to take care about the client load balancing.

### Component structure

The LightWeigh integration zone two key components are:

1. Proxy module
2. Customizetd http server module

#### Proxy

The proxy module is based on the high performance and reliable HaProxy load balancer application.

The proxy module responsible for:

* https termination
* load balancing
* Client cert username forward



|  |  |  |
| --- | --- | --- |
| **Component** | **Solution** | **Role** |
| Https termination | SSL setup | For security reason https connection is mandatory from the caller sides. The SSL setup provides the https connection in two manner: one way ( without client cert ) and two way ( with client cert ).  If the ssl is two way keep in mind that the authentication happens here. |
| Load balancing | Load balancer setup | The customized LightWeight Integration module (WildFlyUndertow) can run in several instances. |
| Client cert username forward | Http set header setup | During the https termination the original https encryption cannot be reproduce due to the lack of the client private key, therefore the client cert and its attributes can forwarded in the header for further investigation. This information is used for the authorization. the forwarded attribute is the CN. |

### Core

The core part of the LightWeight integration is the Customized WildFly/Undertow module, this part is responsible for:

* Runtime configuration
* Routing
* Authentication/authorization
* Logging
* Message validation
* Throttling



|  |  |  |
| --- | --- | --- |
| **Component** | **Solution** | **Role** |
| Configuration | Lightweight Integration config XML | This custom XML schema describes the LightWeight integration solution customization possibilities. |
| Configuration | Lightweight Integration config XML transformer | This component transforms the LightWeight Integration configuration into the proper WildFly/Undertow configuration. |
| Configuration | WildFly/Undertow configuration | The core LightWeight Integration solution standard implementation dependent configuration. |
| Routing | Reverse proxy handler | This handler is responsible for forwarding an appropriate incoming request towards the proper host application proxy. |
| Authentication / authorization | Wildfly security module | This component is responsible for the basic authentication and authorization. Authorization means which application has allowed tom call a particular service. |
| Logging | Custom log handler | This handler module is responsible for logging the incoming/outgoing messages in 3 manner: full, context, minimal. |
| Message validation | Custom message validation handler | This handler responsible for validating the incoming messages against its schema. this feature can be turned off. |
| Throttling | Request Limit handler | This handler is responsible for limiting the incoming parallel requests by messages. |

## Implementation design

### HaProxy setup

Fill out how to setup the HaProxy

#### HTTPS termination

Fill out how to setup the SSL termination in the configuration

sample needed

#### Load balancing

Fill out how to setup the LB in the configuration

sample needed

#### Client username forward from client cert

Fill out how the username forward are set

sample needed

### Wildfly/Undertow setup

The core layer based on the WildFly’s (JBoss) Undertow http server and servlet container.

The server application can be downloaded from the following url:

<http://wildfly.org/downloads/>

During the implementation we use the Wildfly 10.0 Final version, Servlet-Only Distribution.

*Setup:*

* Extract the downloaded zip into a folder on the target computer.
* Use the standalone.xml configuration and set the following ports properly:
  + management http/https
  + ajp
  + http
  + https (it won’t be used)

*<socket-binding-group name="standard-sockets" default-interface="public" port-offset="${jboss.socket.binding.port-offset:0}">*

*<socket-binding name="management-http" interface="management" port="${jboss.management.http.port:9190}"/>*

*<socket-binding name="management-https" interface="management" port="${jboss.management.https.port:9193}"/>*

*<socket-binding name="ajp" port="${jboss.ajp.port:8109}"/>*

*<socket-binding name="http" port="${jboss.http.port:8192}"/>*

*<socket-binding name="https" port="${jboss.https.port:8193}"/>*

*</socket-binding-group>*

The Wildfly can run in docker as well.

*Setup the log:*

* set logger

*<logger category="hu.telekom.lwi" use-parent-handlers="false">*

*<level name="DEBUG"/>*

*<handlers>*

*<handler name="LWI\_LOG\_FILE"/>*

*</handlers>*

*</logger>*

* set log file

*<size-rotating-file-handler name="FILE" autoflush="true">*

*<level name="INFO"/>*

*<formatter>*

*<pattern-formatter pattern="%d{HH:mm:ss,SSS} %-5p [%c] (%t) %s%E%n"/>*

*</formatter>*

*<file relative-to="jboss.server.log.dir" path="lwi\_server1.log"/>*

*<rotate-size value="100M"/>*

*<max-backup-index value="100"/>*

*<suffix value=".yyyy-MM-dd"/>*

*<append value="true"/>*

*</size-rotating-file-handler>*

### LWI setup

The LWI has own configuration schema, see in the Appendixes or the HLD.

There should be an own directory for the LWI on the same machine as the Wildfly.

*Structure:*

* lwi
  + bin
  + config
  + lib

|  |  |
| --- | --- |
| bin | transformation script |
| config | lwi configuration |
| lib | transformation binaries |

### LWI module setup

The LWI module – build by Maven - is installed as a Wildfly module.

component name:

* wfplugins-1.0.0.jar

module path:

* wildfly-servlet-10.0.0.Final\modules\hu\telekom\lwi\main\

module dir:

* module.xml
* wfplugins-1.0.0.jar

module.xml

<?xml version="1.0" encoding="UTF-8"?>

<module xmlns="urn:jboss:module:1.3" name="hu.telekom.lwi">

<resources>

<resource-root path="wfplugins-1.0.0.jar"/>

</resources>

<dependencies>

<module name="io.undertow.core"/>

<module name="org.jboss.xnio"/>

<module name="org.jboss.logging"/>

</dependencies>

</module>

### LWI components

The LWI components fulfil the following requirements:

* routing
* authentication
* log
* message validate
* throttling

#### Routing component

Built in Undertow handler:

* reverse-proxy handler

#### Authenticating component

Custom Undertow handler, part of the LWI module.

classes:

....

Tömör leírás

#### Log component

Custom Undertow handler, part of the LWI module.

classes:

....

Tömör leírás konfig lehetőségek is, működés, stb.

#### Message validation component

Custom Undertow handler, part of the LWI module.

classes:

....

Tömör leírás konfig lehetőségek is, működés, stb.

#### Throttling component

Built in Undertow handler:

RequestDumpingHandler

### LWI configuration

The LWI has its own configuration see schema in the appendixes.

#### Global LWI configuration

#### Configuring the consumer part

#### Configuring the provider part

The LWI main feature is to providing entry point for business services.

The followings should be setup properly for a new service.

|  |  |
| --- | --- |
|  |  |
| name |  |
| businessService |  |
| validationType |  |
| maxRequests |  |
| requestQueueLength |  |
|  |  |
|  |  |

*<?xml version="1.0" encoding="UTF-8"?>*

*<tns:lwiConfiguration xmlns:tns="http://telekom.hu/LwiConfiguration" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://telekom.hu/LwiConfiguration LwiConfiguration.xsd ">*

*<tns:log>*

*<tns:detail>FULL</tns:detail>*

*</tns:log>*

*<tns:consumers>*

*<tns:consumerApplication>*

*<tns:name>mami</tns:name>*

*<tns:password>12345</tns:password>*

*</tns:consumerApplication>*

*</tns:consumers>*

*<tns:providers>*

*<tns:provider>*

*<tns:name>cnr</tns:name>*

*<tns:loadBalancerHost>shark-dev</tns:loadBalancerHost>*

*<tns:loadBalancerPort>17001</tns:loadBalancerPort>*

*<tns:protocol>https</tns:protocol>*

*<tns:services>*

*<tns:service>*

*<tns:name>getMsisdn</tns:name>*

*<tns:businessService>cnrservice/getmsidn</tns:businessService>*

*<tns:validationType>CTX</tns:validationType>*

*<tns:maxRequests>20</tns:maxRequests>*

*<tns:requestQueueLength>40</tns:requestQueueLength>*

*<tns:allowedConsumers>*

*<tns:consumerName>mami</tns:consumerName>*

*</tns:allowedConsumers>*

*</tns:service>*

*</tns:services>*

*</tns:provider>*

*</tns:providers>*

# Appendixes

## Configuration schema

*<?xml version="1.0" encoding="UTF-8"?>*

*<schema targetNamespace="http://telekom.hu/LwiConfiguration"*

*elementFormDefault="qualified" xmlns="http://www.w3.org/2001/XMLSchema"*

*xmlns:tns="http://telekom.hu/LwiConfiguration">*

*<complexType name="LwiConfiguration">*

*<sequence>*

*<element name="log" type="tns:LwiLog" minOccurs="1"*

*maxOccurs="1">*

*</element>*

*<element name="consumers" minOccurs="1" maxOccurs="1">*

*<complexType>*

*<sequence>*

*<element name="consumerApplication" type="tns:LwiConsumer"*

*minOccurs="0" maxOccurs="unbounded">*

*</element>*

*</sequence>*

*</complexType>*

*</element>*

*<element name="providers" minOccurs="1" maxOccurs="1">*

*<complexType>*

*<sequence>*

*<element name="provider" type="tns:LwiProvider" minOccurs="0"*

*maxOccurs="unbounded"/>*

*</sequence>*

*</complexType>*

*</element>*

*</sequence>*

*</complexType>*

*<complexType name="LwiLog">*

*<sequence>*

*<element name="detail" minOccurs="1" maxOccurs="1">*

*<simpleType>*

*<restriction base="string">*

*<enumeration value="FULL"/>*

*<enumeration value="CTX"/>*

*<enumeration value="MIN"/>*

*</restriction>*

*</simpleType>*

*</element>*

*</sequence>*

*</complexType>*

*<complexType name="LwiConsumer">*

*<sequence>*

*<element name="name" type="string" minOccurs="1" maxOccurs="1">*

*</element>*

*<element name="password" type="string" minOccurs="0"*

*maxOccurs="1"/>*

*</sequence>*

*</complexType>*

*<complexType name="LwiService">*

*<sequence>*

*<element name="name" type="string" minOccurs="1" maxOccurs="1">*

*</element>*

*<element name="businessService" type="string" minOccurs="1"*

*maxOccurs="1">*

*</element>*

*<element name="validationType" minOccurs="1" maxOccurs="1">*

*<simpleType>*

*<restriction base="string">*

*<enumeration value="CTX"/>*

*<enumeration value="MSG"/>*

*<enumeration value="NO"/>*

*</restriction>*

*</simpleType>*

*</element>*

*<element name="maxRequests" type="int" minOccurs="1"*

*maxOccurs="1">*

*</element>*

*<element name="requestQueueLength" type="int" minOccurs="1"*

*maxOccurs="1">*

*</element>*

*<element name="allowedConsumers">*

*<complexType>*

*<sequence>*

*<element name="consumerName" type="string" minOccurs="0"*

*maxOccurs="unbounded"/>*

*</sequence>*

*</complexType>*

*</element>*

*</sequence>*

*</complexType>*

*<element name="lwiConfiguration" type="tns:LwiConfiguration"/>*

*<complexType name="LwiProvider">*

*<sequence>*

*<element name="name"*

*type="string" minOccurs="1" maxOccurs="1">*

*</element>*

*<element name="loadBalancerHost"*

*type="string" minOccurs="1" maxOccurs="1">*

*</element>*

*<element name="loadBalancerPort"*

*type="int" minOccurs="1" maxOccurs="1">*

*</element>*

*<element name="protocol" minOccurs="1" maxOccurs="1">*

*<simpleType>*

*<restriction base="string">*

*<enumeration value="http"></enumeration>*

*<enumeration value="https"></enumeration>*

*</restriction>*

*</simpleType>*

*</element>*

*<element name="services" minOccurs="1" maxOccurs="1">*

*<complexType>*

*<sequence>*

*<element name="service" type="tns:LwiService"*

*minOccurs="0" maxOccurs="unbounded">*

*</element>*

*</sequence>*

*</complexType>*

*</element>*

*</sequence>*

*</complexType>*

*</schema>*

## Configuration sample