DCR Plug-in Support One Pager

1 Introduction

1.1 Project/Component Working Name

Sailfin 2.0 / Converged Load Balancer

1.2 Name(s) and e-mail address of Document Author(s)/Supplier

Joel Binnquist : joel.xc.binnquist@ericsson.com

Change log:

Version	Date	Comments	Author
0.1	2009-01-26	First version	Joel Binnquist
0.2	2009-02-04	Added reference to requirements and specified HttpServletRequest as external interface.	Joel Binnquist
0.3	2009-02-09	Added reference to requirement 10565-0192/03445	Joel Binnquist

1.3 Date of This Document

2009-02-09

2 Project Summary

2.1 Project Description

The Data Centric Rules (DCR) is a part of the Converged Load Balancer making it possible to configure how the key, used as input to the consistent hash, is extracted from a request.

The rules for a specific CLB configuration can currently be specified using a proprietary language in XML. However, this language is somewhat limited, and moreover, it is hard to understand.

This one pager describes the enhancement, proposed to CLB, that would make it possible to configure the DCR using a plug-in in the form of a Java class.

2.2 Risks and Assumptions

-

3 Problem Summary

3.1 Problem Area

One of the requirements (105 65-0192/03443) of the Ericsson applications is that Sailfin shall support the ability to specify DCR rules where one can write a condition on a specific SIP method. Something like:

if *req.method equals REGISTER* ***then***
extract key from req.uri
else
extract key from req.myheader

Today one cannot specify a comparison condition on a request method; the only entity that one can express a comparison condition on is the session case. This could be solved by enhancing the existing language.

Another of the requirements (105 65-0192/03444) states that it should be possible to strip parameters from the request-URI in case it is used as hash key.

While analyzing these requirements it was suggested that applications should be able to configure DCR via plug-ins (written in Java) rather than enhancing the proprietary language. Thus when the CLB needs to extract the hash key from a request it calls the currently installed plug-in. This is suggested in the requirement 105 65-0192/03445.

It is important that Sailfin supports dynamic reconfiguration of the rules, which requires that the plug-in classes can be unloaded or reloaded.

It shall be possible to install/download a new plug-in via the CLI and Admin GUI without having to manually copy it to the file system of Application Server (in a manner similar to deploying applications).

If possible it should be possible to install the plug-in in the form of Java source code that is compiled by the container.

The old solution with the XML-based language should be deprecated or discontinued, to minimize future costs for maintenance.

3.2 Justification

The advantages with a plug-in solution would be:

- The language is well-known, which makes it easier for the application developer, who shall write the rules.
- The solution will be easier to maintain for the Sailfin development organization compared to a proprietary XML based language
- The solution will be much more flexible for the applications; the applications will be able to use the full power of the Java language when expressing the rules.
- It will be possible for the applications to verify and debug the rules using standard tools; today this is not possible.
- There will be no need to spend design hours for enhancement of a proprietary the language in the future.

4 Technical Description

4.1 Details

The solution re-uses the existing configuration interface for setting the DCR file by allowing the DCR file to be a jar-file containing an implementation of a DCR plugin-interface.

Optionally, the solution could also allow the file to be a java source code file that will be compiled in run-time (when setting the DCR file) by the container.

To configure the Data Centric function of the CLB a plug-in that extracts the hash key from a SIP/HTTP request is implemented and installed into the application server.

The class would be instantiated and then used by the CLB when extracting the hash key, used for consistent hash lookup, when routing requests (i.e. in exactly the same manner as the XLM-based DCR file is used).

The solution also allows for the plug-in developer to access some artifacts that are normally accessible in a servlet, via annotated resource injections:

- DataCentricUtil
- TelUrlResolver
- UriTools
- SipFactory

4.2 Bug/RFE Number(s)

4.2.1 Bug/RFE Numbers from Issue Tracker

-

4.2.2 Requirement Ids that are being addressed as a part of this proposal.

-

4.3 In Scope

-

4.4 Out of Scope

-

4.5 Interfaces

4.5.1 Exported Interfaces

No changes to existing interface (uses the existing interfaces for configuration of DCR).

4.5.2 Imported interfaces

javax.servlet.http.HttpServletRequest

javax.servlet.sip.SipServletRequest

4.5.3 Other interfaces (Optional)

-

4.6 Doc Impact

HA Admin Guide.

Administration Reference.

4.7 Admin/Config Impact

No changes to existing interface (uses the existing interfaces for configuration of DCR).

4.7.1 Configuration changes needed

-

4.7.2 CLI / GUI impact if any

-

4.8 HA Impact

-

4.9 I18N/L10N Impact

-

4.10 Packaging & Delivery

-

4.10.1 Binaries in which the code is delivered

-

4.11 Security Impact

-

4.12 Compatibility Impact

-

4.13 Dependencies

If run-time compilation shall be supported the tools.jar of the JDK must be included in the Sailfin installation.

4.13.1 Changes required in GlassFish

-

4.13.2 Third Party APIs

-

4.14 Miscellaneous

-

5 Open Issues

Shall run-time compilation be supported?

6 Reference Documents

7 Schedule

7.1 Projected Availability

Sailfin 2.0