Open Diameter Conformance Testing

Dario Djuric*, Ognjen Dobrijevic*, Darko Huljenic** and Maja Matijasevic*
* University of Zagreb Faculty of Electrical Engineering and Computing, Unska 3, HR-10000 Zagreb, Croatia
** Research and Development Centre, Ericsson Nikola Tesla, Krapinska 45, HR-10002 Zagreb, Croatia
{dario.djuric, ognjen.dobrijevic, maja.matijasevic}@fer.hr, darko.huljenic@ericsson.com

***Abstract*—The Diameter Base protocol, originally developed by the Internet Engineering Task Force (IETF), provides the Authentication, Authorization and Accounting (AAA) functionality needed for network access and mobility in IP-based networks. Diameter has also been adopted by the Third Generation Partnership Project (3GPP) and other standards bodies as an AAA framework in the IP Multimedia Subsystem (IMS). With the growing interest in Diameter, there is an ongoing effort to develop an open source Diameter implementation under the Open Diameter Project (http://www.opendiameter.org/). This paper presents the results of conformance testing of Open Diameter Release 1.0.7-i. We describe the test cases, our laboratory environment and test results, with the goal to provide a neutral evaluation of this implementation to the Open Diameter Community, as well as to potential future Diameter application testers and developers.**

I. INTRODUCTION

Authentication, Authorization and Accounting (AAA) are fundamental networking functions used to keep track of user activity and consumption of network resources. The first AAA protocol specified by the Internet Engineering Task Force (IETF) was the Remote Authentication Dial In User Service (RADIUS) protocol [5], later succeeded by the Diameter protocol.

Diameter [4] is specified by the IETF and adds several enhancements over RADIUS, such as the use of reliable transport protocols (e.g., Transmission Control Protocol, TCP), capability negotiation, network and transport level security, and more. It is generally referred to as the "next generation AAA protocol", especially after being adopted by the Third Generation Partnership Project (3GPP) for use in 3G networks, and specifically the IP Multimedia System (IMS) [6]. In IMS, Diameter is used over several signaling interfaces in the core network.

There are a number of Diameter implementations and a comparison may be found in our previous work [1]. To our knowledge, Open Diameter [2] is one of the most complete and stable open-source implementations. This paper focuses on testing the Open Diameter implementation and determining conformance of its base functionality with the Diameter base specification [4]. The conformance tests performed are based on an IETF Internet-Draft entitled "Diameter Base Protocol Interoperability Test Suite" [3]. The version under test was the Open Diameter Release 1.0.7-i and the testing was conducted in fall 2007. Results analysis was carried out by comparing functional behavior of prototype Open Diameter applications with required functionality, thus providing a reference to developers wishing to use this particular implementation, or incorporate it in their work.

The paper is organized as follows. Section II gives a brief overview of the Diameter protocol, while section III introduces Open Diameter as a Diameter implementation. Section IV describes how the conformance testing was performed in a laboratory environment. Results are presented and discussed in Section V. Section VI concludes the paper.

II. DIAMETER PROTOCOL

The Diameter protocol contains two components (Fig. 1): 1) a Base protocol, and 2) a set of Diameter applications on top of it. The Base protocol comprises the core protocol functionality. It takes care of transport protocols, network addressing and security, and allows applications developed by various vendors to use the core functionality relevant to them. The applications extend the Base protocol with additional functionality, such as additional charging methods, additional security mechanisms, etc.

Diameter is designed with the Peer-to-Peer (P2P) architecture in mind, meaning that every node can act as a client, a server, or both. Each node is located within a particular *realm*. Realm is basically a collection of common Diameter nodes, similarly to a domain in the Domain Name System (DNS). In addition to Diameter clients and servers, a Diameter node can act as an *agent* as well. Agents are intermediate nodes that support the communication between the client and the server. Diameter defines four types of agents: Relay Agent, Proxy Agent, Redirect Agent, and Translation Agent.

Diameter Base protocol defines 14 messages (Table I). The implementation of these messages is mandatory and applications built on top of the Base protocol make use of them, and may expand them if necessary. A Diameter message consists of a mandatory header and Attribute-Value Pairs (AVPs) that carry application-specific data.

Before an exchange of messages between adjacent nodes can begin, a *session* must be established. A session is a logical connection between two Diameter nodes, carried over a connection (such as a TCP connection). An example of Diameter nodes exchanging accounting messages is shown in Fig. 2. It contains four types of nodes: a client, a server, a Redirect Agent, and a Relay Agent.

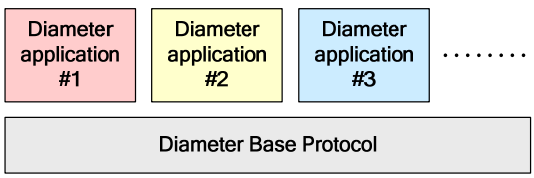

Figure 1. Diameter protocol architecture

TABLE I. DIAMETER BASE PROTOCOL MESSAGES

Message Name	Abbr.	Description
Accounting-Request	ACR	An accounting record request initiated by the client.
Accounting-Answer	ACA	Server's acknowledgement of an accounting record.
Capabilities-Exchange-Request	CER	A request message for peer connection.
Capabilities-Exchange-Answer	CEA	A response message to peer connection request.
Device-Watchdog-Request	DWR	A keep-alive message sent between two peers on a regular basis.
Device-Watchdog-Answer	DWA	An acknowledgement to a keep-alive request.
Disconnect-Peer-Request	DPR	A request for peer disconnection.
Disconnect-Peer-Answer	DPA	A response to a peer-disconnection request.
Abort-Session-Request	ASR	A session-termination request initiated by the client.
Abort-Session-Answer	ASA	Server's response to the session-termination request (ASR).
Session-Termination-Request	STR	A session-termination request initiated by the server.
Session-Termination-Answer	STA	Client's response to the session-termination request (STR).
Re-Auth-Request	RAR	Server initiated message for a user re-authorization.
Re-Auth-Answer	RRA	Client's response to a user re-authorization request.

The client is configured to send its messages (e.g., an ACR) to the Redirect Agent. However, the Redirect Agent does not handle Diameter messages by itself. Instead, it gives the client redirect instructions (AVP set to REDIRECT) to send the messages to the proper Relay Agent. The Relay Agent, on the other side, is configured to forward all received messages to the destination server. Corresponding answer (here, an ACA) follows the same routing path until it reaches the client.

III. OPEN DIAMETER

Open Diameter [2] is an open-source implementation of the Diameter protocol. The Base protocol implementation is available as a C++ library and is currently supported under Linux®, BSD®, and Microsoft Windows® operating systems.

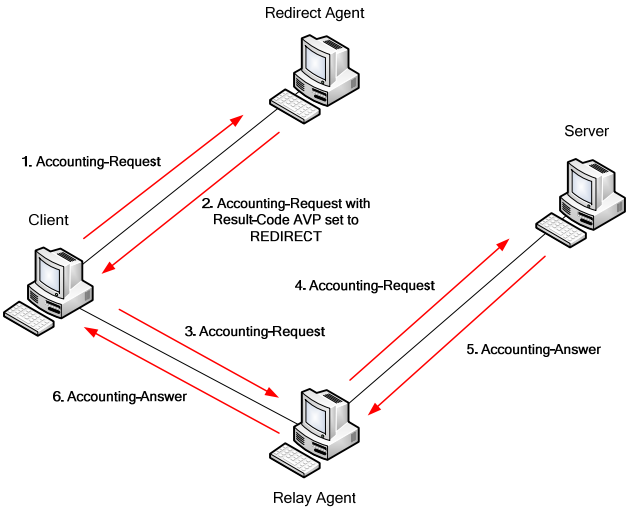

Figure 2. An example of exchanging Diameter accounting messages

For developers wishing to build Diameter applications on top of the Base protocol, Open Diameter provides an Application Programming Interface (API) [2]. As required by the protocol specification, Open Diameter supports numerous AVPs to be used for the Base protocol, as well as for the purposes of applications built on top of it. The selected AVPs referred to in this paper are given in Table II.

The Open Diameter distribution comes with several packages. The three most important packages are 1) the Base protocol implementation, 2) a message parser used for extracting AVP data from Diameter messages, and 3) a support library required for building applications on top of the Base protocol. Most of the libraries come with test applications that demonstrate basic functionalities of each library. For the purpose of this work, test applications of the Base protocol implementation in the Open Diameter Release 1.0.7-i have been used to conduct conformance testing.

An example of a test application is a process of exchanging messages between two Diameter nodes. The first step is the session establishment. This is done through the *Capabilities-Exchange* "negotiation". The client will send a CER, and the server will respond with a CEA. If this is done successfully, communication can begin. A typical "real-world" scenario would involve a user client requesting authentication/authorization from a network access server. In order to grant this user client access to the network and specific types of desirable service (e.g., bandwidth control), the access server could use Diameter to identify the user and determine the terms of access. This implies the access server incorporating a Diameter client, which will send a Diameter application *User-Authorization-Request* to a Diameter server providing AAA services. The Diameter server will respond with a Diameter application *User-Authorization-Answer*. The authentication/authorization is complete if the AVP contained within the answer equals to "Success".

TABLE II. THE SELECTED OPEN DIAMETER AVPS

AVP	Description
Session-Id	A unique number that is used to determine a particular session between two peers.
Result-Code	A number indicating the result of a request.
Origin-Host	The host name of the message initiator.
Origin-Realm	The realm of the message initiator.
Destination-Host	The host name of a peer the initiator is sending a message to.
Destination-Realm	The realm of a peer the initiator is sending a message to.

IV. CONFORMANCE TESTING

The Open Diameter conformance testing is based on the "Diameter Base Protocol Interoperability Test Suite" [3], a working Internet-Draft of the IETF. It describes a collection of functional test cases to aid in evaluating conformance of a given Diameter implementation to the protocol specification. The entire test suite is categorized by different applications and extensions, such as Diameter Base Protocol, Diameter Credit Control, Diameter Session Initiation Protocol (SIP), 3GPP Interfaces, etc. Each of these categories is further subdivided depending on required and optional functionality of the implementation.

Testing described in this paper focuses on required functionality from the Diameter Base Protocol category, which is composed of the following subcategories:

- *Connectivity and Peering* – tests related to the ability of two peers to establish a successful connection between each other (in various circumstances), the ability to successfully tear down the connection, and the ability to respond to an unexpected disconnection, in which case a failover process is required;
- *Routing* – tests relating to the process in which Diameter messages, sent by a client, are being forwarded through intermediate nodes before reaching a server. These tests are meant to show that Diameter can perform routing based on destination host and realm, or solely on destination realm, and that it can detect routing loops;
- *Relay Agent* – tests relating to the capabilities of a Relay Agent, the agent that performs message routing; and
- *Redirection Agent* – tests related to the capabilities of a Redirect Agent. Unlike Relay Agent, a Redirect Agent only provides routing instructions to other nodes.

In order to perform tests mentioned above, the test suite proposes a mesh topology of four nodes (Fig. 3). The topology shown in Fig. 3 was realized using a virtualization tool called VMware [7], combining one physical node (a personal computer, *imgtest*) with three virtual ones (*node1*, *node2*, and *node3*) running simultaneously.

The *imgtest* node was a Pentium IV 3.0 GHz system with 1 GB RAM. It was connected to a real Local Area Network (LAN) used in a laboratory and to the rest of nodes through virtual connections. All the nodes were running a Linux operating system (OS). The *imgtest* node was running SUSE® 10.1 OS, while other nodes were running Xubuntu® 7.04 OS.

During conformance testing a total of 32 tests were performed. The outcome of each test was one of the following:

- *Passed* – Open Diameter successfully passed the test, meaning that it performed according to the protocol specification;
- *Failed* – Open Diameter did not perform according to the specification, or it performed somehow differently, in either way causing the test to fail; and,
- *Not Completed* – the test was not performed due to limited resources or the absence of proper supporting applications.

A. Connectivity and Peering (a total of 22 tests)

In the *Connectivity and Peering* subcategory most of the tests required the use of only two nodes, leaving the remaining nodes offline. The tests relating to connection establishment required one side to advertise support for a certain application or security protocol (such as Transport Layer Security, TLS, or a mechanism within the IP security architecture, IPsec), and the other side accepting the connection. The connection could be established only in the case of both sides advertising support for the same application(s). If the connection establishment failed, then one of the sides would have to return the AVP indicating the reason for connection failure.

In the absence of Diameter messages, the two sides are required to exchange keep-alive messages in order to keep the connection established. The keep-alive messages are called *Watchdog* messages. The proper exchange of these messages was also tested within this category.

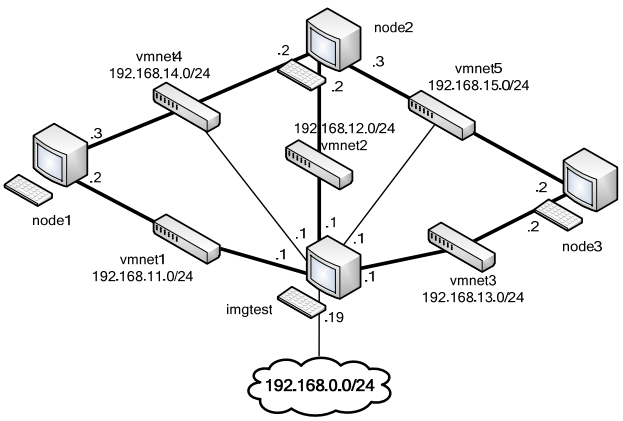

Figure 3. Testing environment

Each Diameter node holds a list of adjacent Diameter peers (*peer table*). The Diameter *daemon*, running the node functionality, will try to connect to each of these peers automatically at startup. A connection attempt from an unknown peer (not listed in the peer table) should be rejected. This test, termed "The unknown peer test", is shown in Fig. 4.

The next group of tests is related to the election process. This process requires that the two peers attempt to make a connection to each other simultaneously or within a time frame of several milliseconds. The process is then used to select the connection initiator and the connection responder.

Finally, the test suite is used to evaluate the disconnection, failover and failback procedures. The disconnection procedure is tested in three cases:

- regular disconnection, which is done through the exchange of the DPR/DPA messages;
- disconnection caused by system-level events, such as transport resets, socket errors, system link-down signals, etc.; and,
- disconnection after a *Watchdog* timeout.

Testing the failover procedure requires the use of all four nodes in the topology (Fig. 3). One test simulated disconnection of the *vmnet1* subnetwork (link) by manual shutdown of the 192.168.11.1 interface. After the shutdown, the remaining messages were sent through *node2*. When the interface was brought back up, the messages were routed again through *imgtest*.

B. Routing (a total of 7 tests)

Tests of the *Routing* subcategory showed how Open Diameter reacted to differing routing metric values (relating to number of "hops", or intermediate nodes). They also showed how Open Diameter reacted when it was supposed to route messages by means of the *Destination-Host* AVP or the *Destination-Realm* AVP. Routing tables can create routing loops. This case was also tested, and the node that detected a routing loop had to send a message back to the initiator, stating that it could not deliver the message.

C. Relay Agent (a total of 1 test)

The *Relay Agent* subcategory includes one test, but this test overlapped with the tests in the *Routing* subcategory.

D. Redirection Agent (a total of 1 test)

The *Redirection Agent* test was based upon a scenario in which *node1*'s primary route to *node3* was through *node2*, however *node2* gave a redirect instruction back to *node1*, stating that it should send its request messages to *imgtest* instead.

E. Optional functionality from the Diameter Base Protocol category (a total of 2 tests)

Moreover, there were two additional test cases considered of which the first one tested how Open Diameter reacted to a non-CEA message being received during CER/CEA exchange, while the second case tested how a dynamic connection to an unknown peer could be made.

Test	Negative test for unknown peers. Use the DIAMETER_UNKNOWN_PEER AVP in response to a CER or silently discard the CER to disconnect unknown peers. Intentionally configure *node1* to send an *Origin-Host* to *imgtest* that is not in *imgtest*'s peer table.
Comments	The *imgtest*'s peer table had a single entry with the identity name of *node112*, which was a non existing node name. *node1* attempted to make a connection to *imgtest*, however *imgtest* silently discarded the *node1*'s CER (the first message sent during connection attempt).
Passed	

Figure 4. The unknown peer test

V. RESULTS

Out of the 32 tests performed, 24 were completed successfully. Out of those, 18 tests passed and 6 of them failed due to Open Diameter not conforming to the protocol specification. An overview of the tests that failed is given with Table III. Due to limited space, this section will only elaborate tests that failed or were not completed. It is understood that the rest of the tests passed, with their description found in [3].

In addition, 8 tests defined by the test suite were not completed. One such test (from the *Connectivity and Peering* subcategory) required the TLS handshake to be altered in the code in order to simulate a failure. (The TLS handshake is a standard procedure for establishing a connection using the TLS protocol.) This test was not performed due to implementation of the TLS handshake being coded "deeply" in the operating system protocol stack. We were also incapable to test the election process (with a total of 3 tests from the *Connectivity and Peering* subcategory) since we were unable to make two simultaneous connection attempts within a time frame of several milliseconds. Finally, 4 tests from the *Connectivity and Peering* subcategory (with 2 of them relating to the disconnection tests, and two with the failover and failback tests) required simulation of a *Watchdog* timeout. The *Watchdog* timeout occurs when adjacent nodes do not exchange any messages over a given period of time. For testing purposes a transport-layer filter was required with the ability to discard Diameter *Watchdog* messages. We tried to emulate such a filter by using Linux *iptables* but failed.

VI. CONCLUSION

The purpose of this work was to carry out testing of Diameter protocol functionality within the Open Diameter Release 1.0.7-i implementation and to evaluate its conformance with the Diameter specification. Out of the 32 performed tests, which relate to the Diameter Base protocol, 24 were completed successfully. Out of those 24, 18 tests passed and 6 of them failed, showing partial incompatibility of this particular Open Diameter implementation with the protocol specification. The rest of the conformance tests were not completed as a result of the inability to simulate/emulate certain testing preconditions. The results provide an insight into Open Diameter and its applicability for developing Diameter applications. The results could also serve as a basis for

TABLE III. CONFORMANCE TESTS WHICH OPEN DIAMETER FAILED

Name of the test	Description of the test	Comments
No common security **(the *Connectivity and Peering* subcategory)**	Negative test where the DIAMETER_NO_COMMON_SECURITY ('*') AVP is returned by a peer with no common application supported. Intentionally configure *node1* to send Security-ID AVP with value 1 (TLS) that *imgtest* will not support.	*node1* advertised support for an application not common with *imgtest* and specified TLS connection to be used with *imgtest*. In the CER/CEA exchange, *node1* responded back with the AVP set to DIAMETER_NO_COMMON_APPLICATION instead of '*'. If a common application was advertised by *node1*, then *node1* responded back with '*'. That was not the behavior specified.
Realm based routing **(the *Routing* subcategory)**	Positive test for request routing from the originator. Request messages generated at *node1* should reach *node3* via *imgtest* if the *Destination-Realm* is *realmB* (of *node3*) and all links are up. *node1* and *imgtest* must perform realm routing to reach *node3*.	Realm-based routing requires that the sender of the request message does not provide the *Destination-Host* AVP. In each test, any node acting as the server would process the message successfully. However, it would not compose and send back an answer due to an unknown parser error. Relaying would not work either.
Multi-hop request routing **(the *Routing* subcategory)**	Positive test for multi-hop request routing. Request messages generated at *node1* with *Destination-Realm realmB* should reach *node3* via *node2* and *imgtest* if all links are up, except for *vmnet1* and *vmnet5*. *node1*, *node2* and *imgtest* must perform realm routing.	
Request routing **(the *Routing* subcategory)**	Negative test for request routing. If a request message generated at *node1* has *Destination-Realm realmB* with all links up, except for *vmnet2*, *vmnet5* and *vmnet1*, then *node2* must send an answer to *node1* with the AVP DIAMETER_UNABLE_TO_DELIVER.	
Loop detection **(the *Routing* subcategory)**	Positive test for loop detection can be done if a request originating at *node1* has a *Destination-Realm realmA* and *node1* is configured to route request for *realmA* to *node2*, *node2* will route request for *realmA* to *imgtest*, and *imgtest* will route request back to *node1*.	Loop detection requires that the sender of the request message does not provide the *Destination-Host* AVP. In this test, any node acting as the server would process the message successfully. However, it would not compose and send back an answer due to an unknown parser error. Relaying would not work either.
Connection establishment with unknown peer **(optional functionality from the *Diameter Base Protocol* category)**	Positive test for establishment of connection with unknown peer. Lifetimes of new entry in peer table and connection should be checked against each other. Intentionally configure *node1* to send an *Origin-Host* that is not in *imgtest*'s peer table.	Dynamic connection of *node1* to *imgtest* was successful, but only if *node1* was already listed in *imgtest*'s peer table. If not, *imgtest* would refuse *node1*'s CER message.

improving Open Diameter base functionality in order to achieve full specification conformance.

REFERENCES

[1] S. Tomac, M. Sikirica, L. Skorin-Kapov, and M. Matijasevic, "Implementation of the Diameter-based Cx interface in the IP multimedia subsystem," in *Proceedings of the 29th International Convention MIPRO 2006*, vol. 2, Opatija, Croatia, May 22-26, 2006, pp. 109–114.

[2] Open Diameter Project, Open Diameter Home Page. [Online]. Available: http://www.opendiameter.org/

[3] V. Fajardo et al., "Diameter Base Protocol Interoperability Test Suite", IETF Work in progress, July 2007.

[4] P. Calhoun et al., "Diameter Base Protocol," IETF RFC 3588, September 2003.

[5] C. Rigney et al., "Remote Authentication Dial In User Service (RADIUS)," IETF RFC 2865, June 2000.

[6] 3GPP TS 23.228: "IP Multimedia Subsystem (IMS); Stage 2 (Release 6)," March 2007.

[7] VMware, Inc., VMware Home Page. [Online]. Available: http://www.vmware.com/