

DIME
Internet-Draft
Expires: January 14, 2010

V. Fajardo
Telcordia Technologies
A. McNamee
Openet-Telecom
H. Tschofenig
Nokia Siemens Networks
J. Bournelle
France Telecom R&D
July 13, 2009

Diameter Base Protocol Interoperability Test Suite
draft-fajardo-dime-base-test-suite-02.txt

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on January 14, 2010.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (<http://trustee.ietf.org/license-info>). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Internet-Draft

Base Interoperability Test Suite

July 2009

Abstract

This document describes a collection of test cases to be used for Diameter base protocol interoperability testing.

Table of Contents

1.	Introduction	3
2.	Terminology	3
3.	Diameter Base Protocol Test Suite	3
3.1.	Required	3
3.1.1.	Connectivity and Peering	3
3.1.2.	Routing	7
3.1.3.	Relay Agent	10
3.1.4.	Redirection Agent	10
3.2.	Optional	10
3.2.1.	General Statemachine	10
3.2.2.	Dynamic Peer Discovery	11
4.	Diameter Base Protocol Application Support	11
4.1.	Authentication and/or Authorization	11
4.1.1.	Stateful Session	11
4.1.2.	Stateless Session	12
4.2.	Accounting	12
4.2.1.	Client Session	12
4.2.2.	Server Session	13
5.	Security Considerations	14
6.	IANA Considerations	14
7.	Normative References	14
	Authors' Addresses	14

[1.](#) Introduction

The document is meant to aid in the identifying the functional test cases of a Diameter implementation. The Diameter interoperability test suites are categorized by required and optional functionality. The required functionality is the baseline capability that an implementation must support to allow basic interoperability for that category. Optional functionality covers features that not all implementations support or may wish to test. This document also covers test suites for common application support provided by the diameter base protocol.

At its current state, this document provides only a collection of test cases designed for interoperability. Test plans may be included in future revisions of this work or maybe provided in some other document.

[2.](#) Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

Within this document the terms defined in [[RFC2119](#)] refer to the functionality that has to be provided by an implementation for the usage within this interoperability test document.

[3.](#) Diameter Base Protocol Test Suite

All implementation must conform to [[RFC3588](#)].

[3.1.](#) Required

[3.1.1.](#) Connectivity and Peering

Implementations must conform to [Section 5.6 of \[RFC3588\]](#). Typical test topology for statemachine test uses peer pairs as shown in Figure 1. It is left to the testers if one-to-many or many-to-one connections will be performed to test scalability and loading. The test cases described below references Figure 1 below.

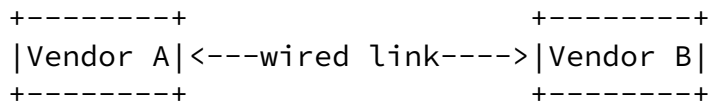


Figure 1: Peer Statemachine Test Topology

[3.1.1.1](#). Capabilities Negotiation

Implementations must be able to perform at least the following behavior described in [Section 5.3 of \[RFC3588\]](#).

- o Positive test for establishment of connection with test pairs advertising support for common application ids (auth, accounting or vendor). Vendor A initiates transport connection to B and trigger the process.
- o Positive test for establishment of connection with test pairs of which one of them is advertising support for only relay app and no other app is in common.
- o Positive test for establishment of connection advertising the relay app in auth app id, acct app id & VSA.
- o Positive test for establishment of connection with test pairs advertising support common transport security, specifically the use of TLS and/or IPsec. It is left up to the participants to generate appropriate keys and certificates specific for this test. Vendor A initiates transport connection to B and trigger the process and advertised proper Inband-Security support.
- o Positive test for DWR/DWA exchange after connection is established. Vendor A and B both exchange watchdogs as per [Section 3.4.1 of \[RFC3539\]](#).
- o Negative test where DIAMETER_NO_COMMON_APPLICATION is returned by a peer with no common application id (auth, accounting or vendor). Intentionally configure vendor A not to advertise any applications, different applications than B or vendor id's known only to A.

- o Negative test where DIAMETER_NO_COMMON_SECURITY is returned by a peer with no common application id. Intentionally configure vendor A to send Inband-Security-AVP with value 1 (TLS) that B will not support.
- o Negative test for unknown peers. Use of DIAMETER_UNKNOWN_PEER or silent discard to disconnect unknown peers. Intentionally configure vendor A to send an origin-host that is not in B's peer table.
- o Negative test case for TLS handshake failure. In the case where the negotiated Inband-Security involves subsequent TLS negotiation, participants can simulate a TLS handshake failure (i.e. via invalid certificates, TLS/SSL version mis-match etc) that must result in the peers being disconnected.

Verification of each test result can be done manually.

[3.1.1.2.](#) Election

This test case refers to [Section 5.6.4 of \[RFC3588\]](#). Responders must be able to resolve contention with initiator peers.

- o Positive test for establishment of connection with responder having higher identity than initiator. Vendor A initiates connection followed by B doing the same a few milliseconds later. Vendor A having a higher identity should close B's connection attempt.
- o Positive test for disconnection with initiator having lower identity than the responder. Vendor A initiates a connection followed by B doing the same a few milliseconds later. Vendor A having a lower identity should close its initial connection attempt.
- o Negative test for disconnection when initiator and responder have equal identity. Vendor A and B will advertise the equal identity. Verify that both peers closed the connection.

Verification of test results can be done manually.

[3.1.1.3.](#) Disconnection

Implementations must conform to [Section 5.6.4 of \[RFC3588\]](#) and [Section 3.4.1 of \[RFC3539\]](#). Peers must be able to quickly determine disconnection events. Verification of test results can be done manually.

- o Positive test for peer disconnection using DPR/DPA exchange.

Vendor A initiates shutdown while connected to B. Implementations behavior may vary depending on disconnection cause such as an eventual connection retry if a disconnection cause of REBOOTING is received.

- o Positive test for detecting disconnection via system level events (i.e., transport resets, socket error, system link-down signals, etc). Implementation must be able to initiate failover procedure. Implementation should also attempt re-connection with lost peer. Hard disconnection of vendor A and B's wired link can be done to simulate this scenario.
- o Positive test for detecting disconnection via watchdog timeout. If there is no activity after a watchdog timer expires with pending request then the peer becomes suspect and implementation must be able to initiate failover procedure. [\[RFC3539\]](#) suggest a minimum watchdog timeout at 6 sec. Vendor B can setup a transport level filter to silently drop AAA traffic from B to simulate unresponsiveness of B.
- o Positive test for resetting connection after at least two(2) watchdog has expired. If a connection is already suspect, the peers must reset the connection. Vendor A or B can setup a transport level filter to silently drop AAA traffic and simulate unresponsiveness of both peers.

Verification of test results can be done manually.

[3.1.1.4](#). Re-Connection Algorithms

Implementations must conform to [Section 2.1 of \[RFC3588\]](#). Although a vendor can implement other algorithms and policies than those proposed in [\[RFC3588\]](#), a default reconnection scheme must be implemented.

- o Positive test for peer re-connection after disconnection has been detected. The link between vendor A and B is temporarily disconnected until such time that disconnection is detected by both peers. The link can then be restored to test the re-connection behavior of both peers. Verify that at least three(3) watchdog exchanges occur before both peers are no longer suspect.

[3.1.1.5](#). Failover and Failback

Implementations must conform to [Section 5.4.5 of \[RFC3588\]](#) and [Section 3.4.2 of \[RFC3539\]](#). Testing failover mechanism requires alternate peer connections. A basic ring topology to test failover and failback is shown in Figure 2 where vendor A has a primary route to vendor C via vendor B and secondary route via vendor D. The same symmetry is applied to all other vendors. As an example, vendor C has a symmetric topology where D is its primary connection and B is its secondary. This allows the same tests to be performed for all vendors. For testing failover on vendor A and B, link0 can be disconnected. For vendor C and D, link2 can be disconnected and so on.

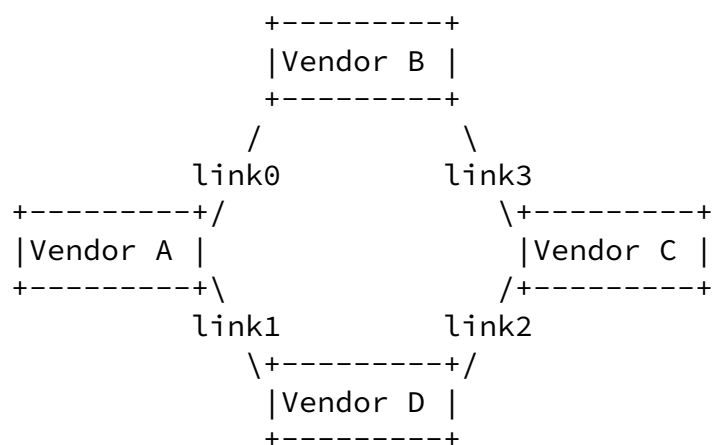


Figure 2: Failover Test Topology

The enumerated test cases refers only to vendor A but can be applied to any of the vendor implementations in Figure 2. Conditions for a positive test requires that realm routes to C is present in A and host routing is not used. Initial traffic should flow from A to C via B (as the primary peer of A).

- o Positive test for failover when link0 is disconnected. Vendor A should have pending requests queued prior to disconnection. Upon disconnection (see [Section 3.1.1.3](#)), verify that the pending request with T-flag set has been forwarded to C via D.
- o Positive test for failover by using device watchdog as a means of triggering link0 disconnection. Vendor A should have pending requests queued prior to disconnection. Upon disconnection (see [Section 3.1.1.3](#)), verify that the pending request with T-flag set

- has been forwarded to C via D.
- o Positive test for failback when link0 is restored and re-connection succeeds (See [Section 3.1.1.4](#)). Verify that new request message is routed back to B.
 - o Negative test to generate DIAMETER_UNABLE_TO_DELIVER on answer message from B to A when Destination-Host is set to C. This can be simulated when link3 is disconnected and Vendor C is not reachable from Vendor. Note that alternate path via Vendor D should not be used.
 - o Negative test to detect duplicate messages on C. Vendor B can disable watchdog processing but still allow request message forwarding. This makes B a suspect peer from A and trigger failover procedure. Forwarding of queue request will then be done through D. However, the original request messages would have reached C via B.

[3.1.2](#). Routing

Implementation must conform to [Section 6 of \[RFC3588\]](#). A basic topology to test Diameter routing is shown in Figure 3 where vendor A and vendor B can deploy two(2) Diameter peers to test host, realm and answer message routing. Vendor A1 and A2 shares the same realm (realmA). Vendor B1 and B2 share a different realm (realmB). Test between both realms are symmetric although the description focuses mostly to vendor A for editorial reasons. The topology is also designed so that multi-hop forwarding, message loopback and agent configuration can be tested. Note that some test cases in this section require link disconnection overlap with the test cases outline in [Section 3.1.1.3](#). An implementation experiencing link disconnection must update its peer and realm route table accordingly. Verification for usage of proper routing AVPs in [Section 6.7 of \[RFC3588\]](#) must be done when testing routing functionality.

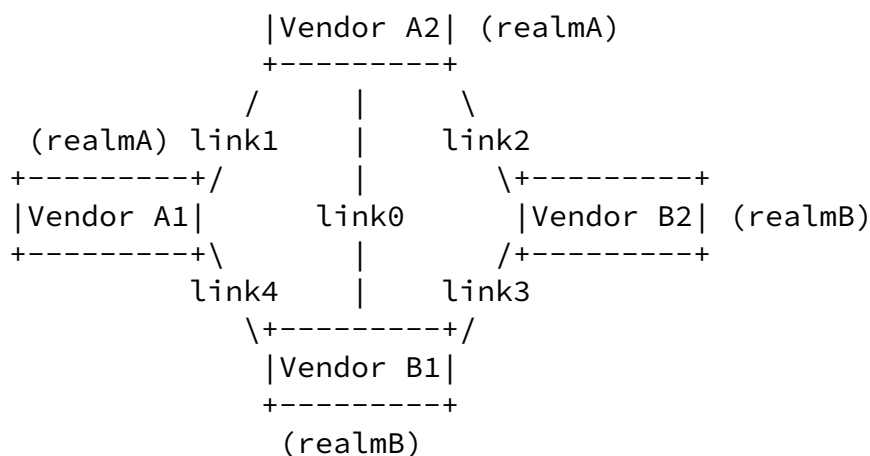


Figure 3: Routing Test Topology

3.1.2.1. Peer Based Request Routing

Implementation must conform to [Section 6.1.5 of \[RFC3588\]](#). In order to perform the test cases the peer requesting the AAA routing must have the destination-host and the destination-realm present in the request message.

- o Positive test for request forwarding from originator. Request messages generated from A1 should reach B2 via B1 if destination-host of the request is B2 and destination-realm is realmB and all links are up. A1 must perform realm routing to reach B1 and B1 must perform forwarding to reach B2. Verification of routing can be done manually if message has reached B2 via link4 and link3.
- o Positive test for multi-hop request forwarding. Request messages generated from A1 with destination-host B2 and destination-realm realmB should reach B2 via A2 and B1 if all links are up except for link4 and link2. A1 and A2 must perform realm routing while B1 performs forwarding. A1 and A2 must be able to route the request message to B1 even if it does not have B2 in its peer table. Verification of routing can be done manually if message has reached B2 via link1, link0 and link3.
- o Negative test for request forwarding. If a request message generated from A1 has a destination-host B2 and destination-realm realmB with all links up except for link0, link2 and link4 then A2 must send an answer message to A1 with result-code DIAMETER_UNABLE_TO_DELIVER. Verification can be done manually if the answer message has reached A1 with E-bit set.

3.1.2.2. Realm Based Routing

Implementation must conform to [Section 6.1](#) and [Section 6.1.6 of \[RFC3588\]](#). Test cases for realm-based request routing must have

destination-realm present but must not have destination-host present in the request message. Note that there is some test overlap with the test cases defined in [Section 3.1.2.1](#).

- o Positive test for request routing from originator. Request messages generated from A1 should reach B2 via B1 if the destination-realm is realmB and all links are up. A1 and B1 must perform realm routing to reach B2. The request must have an id (app, auth or vendor) that B1 must route and B2 must process locally. Verification of routing can be done manually if message has reached B2 via link4 and link3.
- o Positive test for multi-hop request routing. Request messages generated from A1 with destination-realm realmB should reach B2 via A2 and B1 if all links are up except for link4 and link2. A1, A2 and B1 must perform realm routing. The request must have an id (app, auth or vendor) that A1, A2 and B1 must route and B2 must process locally. Verification of routing can be done manually if message has reached B2 via link1, link0 and link3.
- o Negative test for request routing. If a request message generated from A1 has a destination-realm realmB with all links up except for link0, link2 and link4 then A2 must send an answer message to A1 with result-code DIAMETER_UNABLE_TO_DELIVER. Verification can be done manually if the answer message has reached A1 with E-bit set.

[3.1.2.3](#). Answer Message Routing

Implementations must conform to [Section 6.2 of \[RFC3588\]](#). Answer routing can be verified using test cases in [Section 3.1.2.1](#) and [Section 3.1.2.2](#).

[3.1.2.4](#). Loop Detection

Implementation must conform to [Section 6.1.3 of \[RFC3588\]](#). All forwarders must verify that their local identity is not present in the route-record of the request. If it is present, the forwarder must send an answer with result-code DIAMETER_LOOP_DETECTED. If it is not present, implementations must also insert route-records into the request messages.

- o Positive test for loop detection can be done if a request originating from A1 has a destination-realm realmA and A1 is configured to route request for realmA to A2, A2 will route request for realmA to B1 and B1 will route request back to A1. Though A1 originated the request, it must be able to send an answer message with the E-bit set through the request path.

[3.1.3.](#) Relay Agent

Implementations must conform to [Section 2.8.1](#), 6.1.8 and 6.2.2 of [\[RFC3588\]](#). The topology shown in Figure 3 is also used for testing relay agent functionality. Note that an overlap exists with the test case described in [Section 3.1.2](#) when testing relay agents and those test cases should be used here as well. Verification for usage of routing AVPs in [Section 6.7 of \[RFC3588\]](#) must be done when testing agent functionality. Testing of proxy agents that keep vendor specific state, such as proxy-info, proxy-state, proxy-host, is out of scope of this document and can be done in parallel or independent of the test cases enumerated here.

[3.1.4.](#) Redirection Agent

Implementation must conform to [Section 6.1.7 of \[RFC3588\]](#).

Verification can be made by inspecting the redirect answer message whether the result-code is set to DIAMETER_REDIRECT_INDICATION with the E-bit enabled and redirect-hosts added.

- o Positive test for redirection. Request messages generated from A1 should reach B2 via B1 using redirect from A2 and all links are up except link0 and link2. A1 must be configured to forward request message for realmB/B2 via A2. A2 must be configured to act as a redirect agent and signal a redirect indication to A1 to use B1 instead. Verification of redirection can be done manually if messages have reached B2 and re-direct indication was processed by A1.

[3.2.](#) Optional

Implementations must conform to [Section 5.6 of \[RFC3588\]](#). Test topology uses Figure 1. This section describes optional test cases.

[3.2.1.](#) General Statemachine

Implementations must conform to [Section 5.6.1 of \[RFC3588\]](#). The same topology in Figure 1 can be used to perform the test scenarios listed in this section.

- o Negative test for non-CEA message received during CER/CEA

exchange. Silent discard and peer disconnection. Vendor B can initiate a non Diameter server listening on a Diameter defined port number to simulate unrecognizable messages from vendor B. Or the AAA peer of vendor B is modified to generate a non-CEA message once a transport connection setup has been initiated. Verify that vendor A has closed the connection.

[3.2.2.](#) Dynamic Peer Discovery

Implementations must conform to [Section 5.3 of \[RFC3588\]](#).

Implementations must be able to perform at least the following behavior.

- o Positive test for establishment of connection with unknown peer. The topology for this test is Figure 1. Test case is dependent on implementation accepting dynamic peer table updates. In such case, lifetime of new peer entry should be check against lifetime of connection. Intentionally configure vendor A to send an origin-host that is not in B's peer table. Verification of result can be done manually by inspecting the resulting peer table of B.
- o Positive test after redirection ([Section 3.1.4](#)). The topology for this test is Figure 3. Additional verification can be done if [Section 3.1.4](#) is successful. Redirect-host routes can be cached by an implementation as a new route entry. Same scenarios as in the redirect test case except subsequent request messages will be forwarded to B1 by A1. Verify that only the initial message results in a redirect process.

[4.](#) Diameter Base Protocol Application Support

[4.1.](#) Authentication and/or Authorization

Applications intending to use authentication and/or authorization must conform to the statemachine specification in [Section 8.1 of \[RFC3588\]](#). Since these test cases are session level, any topology can be used by a pair of vendors performing interoperability. The minimum topology will be based on Figure 1. Note that majority of these test are performed as part of other Diameter application test cases. Therefore, implementations must be able to comply with these

common cases.

4.1.1. Stateful Session

Implementations must conform to [Section 8.1 of \[RFC3588\]](#).

Implementations must be able to perform at least the following behavior.

- o Positive test for proper stateful session establishment. Verify that auth-session-state with STATE_MAINTAINED is enforced in the client session. Verify that auth-session-lifetime and auth-session-grace-period are negotiated properly and enforced between vendor implementations. Must conform to [Section 8.4 of \[RFC3588\]](#).
- o Positive test for proper stateful session re-auth. Verify server initiated RAR/RAA exchange occurs on auth-lifetime and auth-grace period expiration. Must conform to [Section 8.3 of \[RFC3588\]](#).

Fajardo, et al.

Expires January 14, 2010

[Page 11]

Internet-Draft

Base Interoperability Test Suite

July 2009

- o Positive test for proper stateful session disconnection. Verify client initiated STR/STA exchange occurs for auth failure and session timeout. Verify values of auth-lifetime and auth-grace period against session-lifetime according to [Section 8.9 of \[RFC3588\]](#). Verify application id value carried by the STR/STA message is that of the target application.
- o Positive test for proper stateful session disconnection. Verify server initiated ASR/ASA exchange occurs when server decides to discontinue service. Implementations that allow for hard session termination should be able to perform these tests. Must conform to [Section 8.5 of \[RFC3588\]](#). Verify application id value carried by the STR/STA message is that of the target application.
- o Positive test for proper stateful session disconnection using origin-state-id. Verify a vendor implementation can at least cleanup stateful sessions once it has received a value of origin-state-id greater than a previously known value from the same issuer. Verification can be done in the absence of an STR/STA exchange. Must conform to [Section 8.6 of \[RFC3588\]](#).

4.1.2. Stateless Session

Implementations must conform to [Section 8.1 of \[RFC3588\]](#).

Implementations must be able to perform at least the following behavior.

- o Positive test for proper stateless session establishment. Verify

that auth-session-state negotiation between vendor implementation with NO_STATE_MAINTAINED is enforced in the client session.

- o Positive test for proper stateless session disconnection. Verify that session-lifetime is enforced in the client session.

4.2. Accounting

Applications intending to use Diameter accounting may conform to [Section 8.2](#) and 9 of [\[RFC3588\]](#) if the particular application has not already defined its own statemachine. Since these test cases are also session level, any topology can be used by a pair of vendors performing interoperability. The minimum topology will be based on Figure 1. Note that majority of these test are performed as part of other Diameter application test cases. Therefore, implementations must be able to comply with these common cases.

4.2.1. Client Session

Implementations must conform to [Section 8.2 of \[RFC3588\]](#).

Implementations must be able to perform at least the following behavior. Verification of test results for these cases can be done manually.

- o Positive test for proper client session establishment. Verify that sub-session id is supported and that the client can support event record generation at the least. Verify that the client should at least be able to support DELIVER_AND_GRANT. Test entities must be able to configure their server implementation to send this avp. Must conform to [Section 9.4](#) and 9.8.7 of [\[RFC3588\]](#).
- o Positive test for proper client session termination. Verify that session termination causes transmission of stop record events if any and that all records generated are accounted for. Validation of accounting records can be Diameter application specific and is left to the tester to confirm.
- o Negative test for client session when server reports a failure. Verify that client session can cope wistatemachine draft submissionth failed accounting starts or server storage failure and act accordingly based on [Section 8.2 \[RFC3588\]](#). Behavior of the client can be policy and implementation specific and is left to the tester to confirm. Failed accounting starts and storage

failures can be simulated by mis-configuration of the server test peer.

- o Negative test for client session when connectivity fails. Verify that client session can cope with connectivity failure and act accordingly based on [Section 9.4 \[RFC3588\]](#). The test can overlap with [Section 3.1.1.3](#) and [Section 3.1.1.5](#).

[4.2.2](#). Server Session

Implementations must conform to [Section 8.2](#) and 9 of [\[RFC3588\]](#). Implementations must be able to perform at least the following behavior. Verification of test results for these cases can be done manually. Since server sessions must support record storage it is left to the testers to validate storage ([Section 9.5 \[RFC3588\]](#)), sequencing and co-relation ([Section 9.6 \[RFC3588\]](#)) of records.

- o Positive test for proper server session establishment. Verify that sub-session id is supported and that the server enforces record generation on the client based on accounting-record-type. Verify that supervision timer is enforced when using stateful sessions. Must conform to [Section 9.5 of \[RFC3588\]](#).
- o Positive test for proper server session termination. Verify that expiration of supervision timer in a stateful session terminates both client and server session on any vendor implementation.
- o Negative test for server session when local storage failure occurs. Verify that server can notify client of its state and act accordingly based on [Section 8.2 of \[RFC3588\]](#). Validation is policy and implementation specific and is left to the tester to confirm. Storage failure can be simulated by mis-configuration on the server test peer. This test is mostly a local validation but it can be used in parallel with [Section 4.2.1](#).

[5](#). Security Considerations

This document defines test cases and therefore tests various aspects of the Diameter base specification and various Diameter applications.

[6](#). IANA Considerations

This document does not require actions by IANA.

7. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC3539] Aboba, B. and J. Wood, "Authentication, Authorization and Accounting (AAA) Transport Profile", [RFC 3539](#), June 2003.
- [RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", [RFC 3588](#), September 2003.

Authors' Addresses

Victor Fajardo
Telcordia Technologies
1 Telcordia Drive #1S-222
Piscataway, NJ 08854
USA

Email: vfajardo@research.telcordia.com

Alan McNamee
Openet Telecom Inc
6 Beckett Way, Park West Business Park
Clondalkin, Dublin 12
Ireland

Phone: +353 1 620 4600
Email: alan.mcnamee@openet-telecom.com

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600

Finland

Phone: +358 (50) 4871445

Email: Hannes.Tschofenig@gmx.net

URI: <http://www.tschofenig.priv.at>

Julien Bournelle

France Telecom R&D

38-40 rue du general Leclerc

Issy-Les-Moulineaux 92794

France

Email: julien.bournelle@orange-ftgroup.com