

# Ultra Messaging® Configuration Guide



## Ultra Messaging® Configuration Guide

Published March 2014

Copyright © 2004 - 2014 Informatica Corporation

Informatica Ultra Messaging

Version 5.3

March 2014

Copyright (c) 1998-2014 Informatica Corporation. All rights reserved.

This software and documentation contain proprietary information of Informatica Corporation and are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright law. Reverse engineering of the software is prohibited. No part of this document may be reproduced or transmitted in any form, by any means (electronic, photocopying, recording or otherwise) without prior consent of Informatica Corporation. This Software may be protected by U.S. and/or international Patents and other Patents Pending.

Use, duplication, or disclosure of the Software by the U.S. Government is subject to the restrictions set forth in the applicable software license agreement and as provided in DFARS 227.7202-1(a) and 227.7702-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988), FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as applicable.

The information in this product or documentation is subject to change without notice. If you find any problems in this product or documentation, please report them to us in writing.

Informatica, Informatica Platform, Informatica Data Services, PowerCenter, PowerCenterRT, PowerCenter Connect, PowerCenter Data Analyzer, PowerExchange, PowerMart, Metadata Manager, Informatica Data Quality, Informatica Data Explorer, Informatica B2B Data Transformation, Informatica B2B Data Exchange Informatica On Demand, Informatica Identity Resolution, Informatica Application Information Lifecycle Management, Informatica Complex Event Processing, Ultra Messaging and Informatica Master Data Management are trademarks or registered trademarks of Informatica Corporation in the United States and in jurisdictions throughout the world. All other company and product names may be trade names or trademarks of their respective owners.

Portions of this software and/or documentation are subject to copyright held by third parties, including without limitation: Copyright DataDirect Technologies. All rights reserved. Copyright (c) Sun Microsystems. All rights reserved. Copyright (c) RSA Security Inc. All Rights Reserved. Copyright (c) Ordinal Technology Corp. All rights reserved. Copyright (c) Aandacht c.v. All rights reserved. Copyright Genivia, Inc. All rights reserved. Copyright Isomorphic Software. All rights reserved. Copyright (c) Meta Integration Technology, Inc. All rights reserved. Copyright (c) Intalio. All rights reserved. Copyright (c) Oracle. All rights reserved. Copyright (c) Adobe Systems Incorporated. All rights reserved. Copyright (c) DataArt, Inc. All rights reserved. Copyright (c) ComponentSource. All rights reserved. Copyright (c) Microsoft Corporation. All rights reserved. Copyright (c) Rogue Wave Software, Inc. All rights reserved. Copyright (c) Teradata Corporation. All rights reserved. Copyright (c) Yahoo! Inc. All rights reserved. Copyright (c) Glyph & Cog, LLC. All rights reserved. Copyright (c) Thinkmap, Inc. All rights reserved. Copyright (c) Clearpace Software Limited. All rights reserved. Copyright (c) Information Builders, Inc. All rights reserved. Copyright (c) OSS Nokalva, Inc. All rights reserved. Copyright Edifecs, Inc. All rights reserved. Copyright Cleo Communications, Inc. All rights reserved. Copyright (c) International Organization for Standardization 1986. All rights reserved. Copyright (c) ej-technologies GmbH. All rights reserved. Copyright (c) Jaspersoft Corporation. All rights reserved. Copyright (c) is International Business Machines Corporation. All rights reserved. Copyright (c) yWorks GmbH. All rights reserved. Copyright (c) Lucent Technologies. All rights reserved. Copyright (c) University of Toronto. All rights reserved. Copyright (c) Daniel Veillard. All rights reserved. Copyright (c) Unicode, Inc. Copyright IBM Corp. All rights reserved. Copyright (c) MicroQuill Software Publishing, Inc. All rights reserved. Copyright (c) PassMark Software Pty Ltd. All rights reserved. Copyright (c) LogiXML, Inc. All rights reserved. Copyright (c) 2003-2010 Lorenzi Davide, All rights reserved. Copyright (c) Red Hat, Inc. All rights reserved. Copyright (c) The Board of Trustees of the Leland Stanford Junior University. All rights reserved. Copyright (c) EMC Corporation. All rights reserved. Copyright (c) Flexera Software. All rights reserved. Copyright (c) Jinfonet Software. All rights reserved. Copyright (c) Apple Inc. All rights reserved. Copyright (c) Telerik Inc. All rights reserved.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>), and/or other software which is licensed under various versions of the Apache License (the "License"). You may obtain a copy of these Licenses at <http://www.apache.org/licenses/>. Unless required by applicable law or agreed to in writing, software distributed under these Licenses is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the Licenses for the specific language governing permissions and limitations under the Licenses.

This product includes software which was developed by Mozilla (<http://www.mozilla.org/>), software copyright The JBoss Group, LLC, all rights reserved; software copyright (c) 1999-2006 by Bruno Lowagie and Paulo Soares and other software which is licensed under various versions of the GNU Lesser General Public License Agreement, which may be found at <http://www.gnu.org/licenses/lgpl.html>. The materials are provided free of charge by Informatica, "as-is", without warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose.

The product includes ACE(TM) and TAO(TM) software copyrighted by Douglas C. Schmidt and his research group at Washington University, University of California, Irvine, and Vanderbilt University, Copyright (c) 1993-2006, all rights reserved.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (copyright The OpenSSL Project. All Rights Reserved) and redistribution of this software is subject to terms available at <http://www.openssl.org> and <http://www.openssl.org/source/license.html>.

This product includes Curl software which is Copyright 1996-2007, Daniel Stenberg, <[daniel@haxx.se](mailto:daniel@haxx.se)>. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://curl.haxx.se/docs/copyright.html>. Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

The product includes software copyright 2001-2005 (c) MetaStuff, Ltd. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://www.dom4j.org/license.html>.

The product includes software copyright (c) 2004-2007, The Dojo Foundation. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://dojotoolkit.org/license>.

This product includes ICU software which is copyright International Business Machines Corporation and others. All rights reserved. Permissions and limitations regarding this software are subject to terms available at <http://source.icu-project.org/repos/icu/icu/trunk/license.html>.

This product includes software copyright (c) 1996-2006 Per Bothner. All rights reserved. Your right to use such materials is set forth in the license which may be found at <http://www.gnu.org/software/kawa/Software-License.html>.

This product includes OSSP UUID software which is Copyright (c) 2002 Ralf S. Engelschall, Copyright (c) 2002 The OSSP Project Copyright (c) 2002 Cable & Wireless Deutschland. Permissions and limitations regarding this software are subject to terms available at <http://www.opensource.org/licenses/mit-license.php>.

This product includes software developed by Boost (<http://www.boost.org/>) or under the Boost software license. Permissions and limitations regarding this software are subject to terms available at [http://www.boost.org/LICENSE\\_1\\_0.txt](http://www.boost.org/LICENSE_1_0.txt).

This product includes software copyright (c) 1997-2007 University of Cambridge. Permissions and limitations regarding this software are subject to terms available at <http://www.pcre.org/license.txt>.

This product includes software copyright (c) 2007 The Eclipse Foundation. All Rights Reserved. Permissions and limitations regarding this software are subject to terms available at <http://www.eclipse.org/org/documents/epl-v10.php>.

This product includes software licensed under the terms at <http://www.tcl.tk/software/tcltk/license.html>,

<http://www.bosrup.com/web/overlib/?License>, <http://www.stlport.org/doc/license.html>, <http://asm.ow2.org/license.html>,

<http://www.cryptix.org/LICENSE.TXT>, <http://hsqldb.org/web/hsqldbLicense.html>, <http://httpunit.sourceforge.net/doc/license.html>,

<http://jung.sourceforge.net/license.txt>, [http://www.gzip.org/zlib/zlib\\_license.html](http://www.gzip.org/zlib/zlib_license.html), <http://www.openldap.org/software/release/license.html>,

<http://www.libssh2.org>, <http://slf4j.org/license.html>, <http://www.sente.ch/software/OpenSourceLicense.html>,

<http://fusesource.com/downloads/license-agreements/fuse-message-broker-v-5-3-license-agreement>; <http://antlr.org/license.html>;

<http://aopalliance.sourceforge.net/>; <http://www.bouncycastle.org/licence.html>; <http://www.jgraph.com/jgraphdownload.html>;

<http://www.jcraft.com/jsch/LICENSE.txt>; [http://jotm.objectweb.org/bsd\\_license.html](http://jotm.objectweb.org/bsd_license.html); .

<http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231>; <http://www.slf4j.org/license.html>;

<http://nanoxml.sourceforge.net/orig/copyright.html>; <http://www.json.org/license.html>; <http://forge.ow2.org/projects/javaservice/>,

<http://www.postgresql.org/about/licence.html>, <http://www.sqlite.org/copyright.html>, <http://www.tcl.tk/software/tcltk/license.html>,

<http://www.jaxen.org/faq.html>, <http://www.jdom.org/docs/faq.html>, <http://www.slf4j.org/license.html>;

<http://www.iodbc.org/dataspace/iodbc/wiki/iODBC/License>; <http://www.keplerproject.org/md5/license.html>;

<http://www.toedter.com/en/jcalendar/license.html>; <http://www.edankert.com/bounce/index.html>; <http://www.net-snmp.org/about/license.html>;

<http://www.openmdx.org/#FAQ>; [http://www.php.net/license/3\\_01.txt](http://www.php.net/license/3_01.txt); <http://srp.stanford.edu/license.txt>; <http://www.schneier.com/blowfish.html>;

<http://www.jmock.org/license.html>; <http://xsom.java.net>; and <http://benalman.com/about/license/>;

<https://github.com/CreateJS/EaselJS/blob/master/src/easeljs/display/Bitmap.js>; <http://www.h2database.com/html/license.html#summary>; and <http://jsoncpp.sourceforge.net/LICENSE>.

This product includes software licensed under the Academic Free License (<http://www.opensource.org/licenses/afl-3.0.php>), the Common Development and Distribution License (<http://www.opensource.org/licenses/cddl1.php>) the Common Public License (<http://www.opensource.org/licenses/cpl1.0.php>), the Sun Binary Code License Agreement Supplemental License Terms, the BSD License (<http://www.opensource.org/licenses/bsd-license.php>) the MIT License (<http://www.opensource.org/licenses/mit-license.php>) and the Artistic License (<http://www.opensource.org/licenses/artistic-license-1.0>).

This product includes software copyright (c) 2003-2006 Joe Waines, 2006-2007 XStream Committers. All rights reserved. Permissions and limitations regarding this software are subject to terms available at <http://xstream.codehaus.org/license.html>. This product includes software developed by the Indiana University Extreme! Lab. For further information please visit <http://www.extreme.indiana.edu/>.

This Software is protected by U.S. Patent Numbers 5,794,246; 6,014,670; 6,016,501; 6,029,178; 6,032,158; 6,035,307; 6,044,374; 6,092,086; 6,208,990; 6,339,775; 6,640,226; 6,789,096; 6,820,077; 6,823,373; 6,850,947; 6,895,471; 7,117,215; 7,162,643; 7,243,110; 7,254,590; 7,281,001; 7,421,458; 7,496,588; 7,523,121; 7,584,422; 7,676,516; 7,720,842; 7,721,270; and 7,774,791, international Patents and other Patents Pending.

DISCLAIMER: Informatica Corporation provides this documentation "as is" without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of noninfringement, merchantability, or use for a particular purpose. Informatica Corporation does not warrant that this software or documentation is error free. The information provided in this software or documentation may include technical inaccuracies or typographical errors. The information in this software and documentation is subject to change at any time without notice.

#### NOTICES

This Informatica product (the "Software") includes certain drivers (the "DataDirect Drivers") from DataDirect Technologies, an operating company of Progress Software Corporation ("DataDirect") which are subject to the following terms and conditions:

1. THE DATADIRECT DRIVERS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.
2. IN NO EVENT WILL DATADIRECT OR ITS THIRD PARTY SUPPLIERS BE LIABLE TO THE END-USER CUSTOMER FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR OTHER DAMAGES ARISING OUT OF THE USE OF THE ODBC DRIVERS, WHETHER OR NOT INFORMED OF THE POSSIBILITIES OF DAMAGES IN ADVANCE. THESE LIMITATIONS APPLY TO ALL CAUSES OF ACTION, INCLUDING, WITHOUT LIMITATION, BREACH OF CONTRACT, BREACH OF WARRANTY, NEGLIGENCE, STRICT LIABILITY, MISREPRESENTATION AND OTHER TORTS.

# Table of Contents

<b>1. Configuring Ultra Messaging Options .....</b>	<b>1</b>
1.1. Overview .....	1
1.1.1. Assignment Methods .....	1
1.1.2. Assignment Flow .....	2
1.1.3. Definitions.....	2
1.1.4. Which Method Should I Use?.....	3
1.1.5. Configuration Files .....	3
1.2. Plain Text Configuration Files.....	4
1.2.1. Reading Plain Text Configuration Files.....	4
1.2.2. Plain Text Configuration File Format .....	4
1.3. XML Configuration Files .....	5
1.3.1. Reading XML Configuration Files .....	5
1.3.2. Using XML Configuration Files With a UM Application .....	6
1.3.3. XML Configuration File Format .....	6
1.3.4. Sample XML Configuration File .....	22
1.3.5. XML Configuration File DTD.....	25
1.4. Configuration File Restrictions .....	26
1.5. Attributes Objects.....	26
1.5.1. Creating An Attributes Object .....	27
1.5.2. Setting an Option from a Binary Value.....	28
1.5.3. Setting an Option from a String Value.....	29
1.5.4. Getting an Option as a Binary Value .....	30
1.5.5. Getting an Option as a String Value .....	31
1.5.6. Deleting an Attributes Object .....	32
1.5.7. Restrictions .....	32
1.6. Modifying Current Attributes.....	32
1.6.1. Setting An Option from a Binary Value .....	33
1.6.2. Setting An Option from a String Value.....	33
1.6.3. Restrictions .....	34
1.7. Retrieving Current Option Values .....	34
1.7.1. Getting An Option as a Binary Value .....	34
1.7.2. Getting An Option as a String Value .....	35
<b>2. Example Configuration Scenarios .....</b>	<b>36</b>
2.1. Highest Throughput.....	36
2.2. Lowest Latency .....	36
2.3. Creating Multicast Sources .....	37
2.4. Disabling Aspects of Topic Resolution .....	38
2.4.1. Disabling Topic Advertisements.....	38
2.4.2. Disabling Receiver Topic Queries .....	38
2.4.3. Disabling Wildcard Topic Queries.....	39
2.4.4. Disabling All But the Minimum Topic Resolution Traffic .....	39
2.4.5. Re-establish Pre-4.0 Topic Resolution.....	39
2.5. Unicast Resolver.....	40
2.5.1. Unicast Resolution Across Administrative Domains .....	40
2.6. Configure Previous Port Defaults .....	42

2.7. Configure New Port Defaults .....	42
2.8. Interrelated Configuration Options.....	43
2.8.1. Preventing NAK Storms with NAK Interval Options.....	43
2.8.2. Preventing Tail Loss With TSNI and NAK Interval Options .....	44
2.8.3. Preventing IPC Receiver Deafness With Keepalive Options.....	45
2.8.4. Preventing Erroneous LBT-RM/LBT-RU Session Timeouts.....	46
2.8.5. Preventing Errors Due to Bad Multicast Address Ranges .....	47
2.8.6. Preventing Store or Queue Timeouts .....	47
2.8.7. Preventing ULB Timeouts .....	48
2.8.8. Preventing Unicast Resolver Daemon Timeouts .....	49
2.8.9. Preventing Undetected Late Join Loss.....	49
2.8.10. Preventing Undetected Loss .....	50
<b>3. Common Tasks .....</b>	<b>52</b>
3.1. Configuring Multi-Homed Hosts .....	52
3.2. Traversing a Firewall .....	52
3.3. Running Multiple Applications .....	52
<b>4. Reference .....</b>	<b>54</b>
4.1. Introduction .....	54
4.1.1. Case Sensitivity.....	54
4.1.2. Specifying Interfaces .....	54
4.1.3. Socket Buffer Sizes.....	54
4.1.4. Reference Entry Format.....	55
4.1.5. Network Compatibility Mode.....	55
4.2. Major Options .....	60
4.2.1. context_event_function (context) .....	60
4.2.2. context_name (context).....	60
4.2.3. fd_management_type (context) .....	60
4.2.4. message_selector (receiver).....	61
4.2.5. network_compatibility_mode (context).....	61
4.2.6. operational_mode (context).....	62
4.2.7. ordered_delivery (receiver).....	62
4.2.8. rcv_sync_cache (receiver) .....	63
4.2.9. rcv_sync_cache_timeout (receiver) .....	63
4.2.10. receive_thread_pool_size (context) .....	64
4.2.11. resolver_context_advertisement_interval (context).....	64
4.2.12. resolver_source_notification_function (context) .....	64
4.2.13. source_cost_evaluation_function (context) .....	64
4.2.14. source_event_function (context).....	65
4.2.15. source_includes_topic_index (context) .....	65
4.2.16. transport (source) .....	65
4.2.17. transport_demux_tablesz (receiver).....	66
4.2.18. transport_session_multiple_sending_threads (context).....	66
4.2.19. transport_source_side_filtering_behavior (source).....	67
4.2.20. transport_topic_sequence_number_info_active_threshold (source) .....	67
4.2.21. transport_topic_sequence_number_info_interval (source).....	67
4.2.22. use_extended_reclaim_notifications (source).....	68
4.2.23. use_transport_thread (receiver) .....	68

4.3. Resolver Operation Options .....	69
4.3.1. Minimum Values for Advertisement and Query Intervals .....	69
4.3.2. disable_extended_topic_resolution_message_options (context) .....	70
4.3.3. resolution_no_source_notification_threshold (receiver) .....	70
4.3.4. resolution_number_of_sources_query_threshold (receiver) .....	71
4.3.5. resolver_advertisement_maximum_initial_interval (source) .....	71
4.3.6. resolver_advertisement_minimum_initial_duration (source) .....	71
4.3.7. resolver_advertisement_minimum_initial_interval (source) .....	71
4.3.8. resolver_advertisement_minimum_sustain_duration (source) .....	72
4.3.9. resolver_advertisement_send_immediate_response (source) .....	72
4.3.10. resolver_advertisement_sustain_interval (source) .....	73
4.3.11. resolver_cache (context) .....	73
4.3.12. resolver_datagram_max_size (context) .....	73
4.3.13. resolver_initial_advertisement_bps (context) .....	74
4.3.14. resolver_initial_advertisements_per_second (context) .....	74
4.3.15. resolver_initial_queries_per_second (context) .....	74
4.3.16. resolver_initial_query_bps (context) .....	75
4.3.17. resolver_query_maximum_initial_interval (receiver) .....	75
4.3.18. resolver_query_minimum_initial_duration (receiver) .....	75
4.3.19. resolver_query_minimum_initial_interval (receiver) .....	76
4.3.20. resolver_query_minimum_sustain_duration (receiver) .....	76
4.3.21. resolver_query_sustain_interval (receiver) .....	76
4.3.22. resolver_receiver_map_tablesz (context) .....	77
4.3.23. resolver_send_initial_advertisement (source) .....	77
4.3.24. resolver_source_map_tablesz (context) .....	77
4.3.25. resolver_string_hash_function (context) .....	78
4.3.26. resolver_string_hash_function_ex (context) .....	78
4.3.27. resolver_sustain_advertisement_bps (context) .....	79
4.3.28. resolver_sustain_advertisements_per_second (context) .....	79
4.3.29. resolver_sustain_queries_per_second (context) .....	80
4.3.30. resolver_sustain_query_bps (context) .....	80
4.3.31. resolver_unicast_activity_timeout (context) .....	80
4.3.32. resolver_unicast_change_interval (context) .....	81
4.3.33. resolver_unicast_check_interval (context) .....	81
4.3.34. resolver_unicast_force_alive (context) .....	81
4.3.35. resolver_unicast_keepalive_interval (context) .....	82
4.4. Multicast Resolver Network Options .....	82
4.4.1. resolver_multicast_address (context) .....	82
4.4.2. resolver_multicast_incoming_address (context) .....	83
4.4.3. resolver_multicast_incoming_port (context) .....	83
4.4.4. resolver_multicast_interface (context) .....	83
4.4.5. resolver_multicast_outgoing_address (context) .....	84
4.4.6. resolver_multicast_outgoing_port (context) .....	84
4.4.7. resolver_multicast_port (context) .....	84
4.4.8. resolver_multicast_receiver_socket_buffer (context) .....	85
4.4.9. resolver_multicast_ttl (context) .....	85
4.5. Unicast Resolver Network Options .....	85
4.5.1. resolver_unicast_daemon (context) .....	86

4.5.2. resolver_unicast_interface (context).....	87
4.5.3. resolver_unicast_port_high (context) .....	87
4.5.4. resolver_unicast_port_low (context) .....	87
4.5.5. resolver_unicast_receiver_socket_buffer (context) .....	88
4.6. Transport TCP Network Options.....	88
4.6.1. transport_tcp_interface (receiver).....	88
4.6.2. transport_tcp_interface (source) .....	89
4.6.3. transport_tcp_maximum_ports (context).....	89
4.6.4. transport_tcp_port (source).....	89
4.6.5. transport_tcp_port_high (context) .....	90
4.6.6. transport_tcp_port_low (context) .....	90
4.7. Transport TCP Operation Options .....	90
4.7.1. transport_session_maximum_buffer (source).....	90
4.7.2. transport_tcp_activity_method (receiver).....	91
4.7.3. transport_tcp_activity_timeout (receiver).....	91
4.7.4. transport_tcp_coalesce_threshold (source).....	91
4.7.5. transport_tcp_datagram_max_size (context).....	92
4.7.6. transport_tcp_exclusiveaddr (source) .....	92
4.7.7. transport_tcp_listen_backlog (source).....	92
4.7.8. transport_tcp_multiple_receiver_behavior (source) .....	93
4.7.9. transport_tcp_multiple_receiver_send_order (source) .....	93
4.7.10. transport_tcp_nodelay (source) .....	94
4.7.11. transport_tcp_receiver_socket_buffer (context) .....	94
4.7.12. transport_tcp_reuseaddr (source).....	94
4.7.13. transport_tcp_sender_socket_buffer (source).....	95
4.8. Transport LBT-RM Network Options .....	95
4.8.1. transport_lbtrm_destination_port (source) .....	96
4.8.2. transport_lbtrm_multicast_address (source).....	96
4.8.3. transport_lbtrm_multicast_address_high (context) .....	96
4.8.4. transport_lbtrm_multicast_address_low (context) .....	97
4.8.5. transport_lbtrm_source_port_high (context) .....	97
4.8.6. transport_lbtrm_source_port_low (context) .....	97
4.9. Transport LBT-RM Reliability Options .....	98
4.9.1. LBT-RM Datagram Loss Resulting in Unrecovered Message Loss.....	98
4.9.2. LBT-RM Source Ignoring NAKs for Efficiency.....	99
4.9.3. LBT-RM Receiver Suppressing NAK Generation.....	100
4.9.4. transport_lbtrm_ignore_interval (source).....	101
4.9.5. transport_lbtrm_nak_backoff_interval (receiver) .....	101
4.9.6. transport_lbtrm_nak_generation_interval (receiver) .....	102
4.9.7. transport_lbtrm_nak_initial_backoff_interval (receiver).....	102
4.9.8. transport_lbtrm_nak_suppress_interval (receiver) .....	102
4.9.9. transport_lbtrm_receiver_socket_buffer (context).....	103
4.9.10. transport_lbtrm_send_naks (receiver) .....	103
4.9.11. transport_lbtrm_source_socket_buffer (context).....	104
4.9.12. transport_lbtrm_transmission_window_limit (source).....	104
4.9.13. transport_lbtrm_transmission_window_size (source) .....	104
4.10. Transport LBT-RM Operation Options .....	105
4.10.1. transport_lbtrm_activity_timeout (receiver) .....	106



4.10.2. transport_lbtrm_coalesce_threshold (source).....	106
4.10.3. transport_lbtrm_data_rate_limit (context).....	107
4.10.4. transport_lbtrm_datagram_max_size (context).....	107
4.10.5. transport_lbtrm_preactivity_timeout (receiver).....	107
4.10.6. transport_lbtrm_rate_interval (context).....	108
4.10.7. transport_lbtrm_retransmit_rate_limit (context).....	108
4.10.8. transport_lbtrm_sm_maximum_interval (source).....	108
4.10.9. transport_lbtrm_sm_minimum_interval (source).....	109
4.10.10. transport_lbtrm_tgsz (source).....	109
4.11. Transport LBT-RU Network Options.....	109
4.11.1. transport_lbtru_interface (receiver).....	110
4.11.2. transport_lbtru_interface (source).....	110
4.11.3. transport_lbtru_maximum_ports (context).....	111
4.11.4. transport_lbtru_port (source).....	111
4.11.5. transport_lbtru_port_high (context).....	111
4.11.6. transport_lbtru_port_high (receiver).....	112
4.11.7. transport_lbtru_port_low (context).....	112
4.11.8. transport_lbtru_port_low (receiver).....	112
4.12. Transport LBT-RU Reliability Options.....	112
4.12.1. transport_lbtru_ignore_interval (source).....	113
4.12.2. transport_lbtru_nak_backoff_interval (receiver).....	113
4.12.3. transport_lbtru_nak_generation_interval (receiver).....	113
4.12.4. transport_lbtru_nak_suppress_interval (receiver).....	114
4.12.5. transport_lbtru_receiver_socket_buffer (context).....	114
4.12.6. transport_lbtru_source_socket_buffer (context).....	114
4.12.7. transport_lbtru_transmission_window_limit (source).....	115
4.12.8. transport_lbtru_transmission_window_size (source).....	115
4.13. Transport LBT-RU Operation Options.....	115
4.13.1. transport_lbtru_acknowledgement_interval (receiver).....	116
4.13.2. transport_lbtru_activity_timeout (receiver).....	116
4.13.3. transport_lbtru_client_activity_timeout (source).....	117
4.13.4. transport_lbtru_client_map_size (source).....	117
4.13.5. transport_lbtru_coalesce_threshold (source).....	117
4.13.6. transport_lbtru_connect_interval (receiver).....	118
4.13.7. transport_lbtru_data_rate_limit (context).....	118
4.13.8. transport_lbtru_datagram_max_size (context).....	118
4.13.9. transport_lbtru_maximum_connect_attempts (receiver).....	119
4.13.10. transport_lbtru_rate_interval (context).....	119
4.13.11. transport_lbtru_retransmit_rate_limit (context).....	120
4.13.12. transport_lbtru_sm_maximum_interval (source).....	120
4.13.13. transport_lbtru_sm_minimum_interval (source).....	120
4.13.14. transport_lbtru_use_session_id (source).....	121
4.14. Transport LBT-IPC Operation Options.....	121
4.14.1. transport_lbtipc_activity_timeout (receiver).....	121
4.14.2. transport_lbtipc_behavior (source).....	122
4.14.3. transport_lbtipc_datagram_max_size (context).....	122
4.14.4. transport_lbtipc_id (source).....	123
4.14.5. transport_lbtipc_id_high (context).....	123

4.14.6. transport_lbtpc_id_low (context).....	123
4.14.7. transport_lbtpc_maximum_receivers_per_transport (source).....	123
4.14.8. transport_lbtpc_receiver_operational_mode (context).....	124
4.14.9. transport_lbtpc_receiver_thread_behavior (context).....	124
4.14.10. transport_lbtpc_sm_interval (source).....	125
4.14.11. transport_lbtpc_transmission_window_size (source).....	125
4.15. Transport LBT-RDMA Operation Options .....	126
4.15.1. transport_lbtrdma_datagram_max_size (context) .....	126
4.15.2. transport_lbtrdma_interface (source).....	126
4.15.3. transport_lbtrdma_maximum_ports (context) .....	126
4.15.4. transport_lbtrdma_port (source) .....	127
4.15.5. transport_lbtrdma_port_high (context).....	127
4.15.6. transport_lbtrdma_port_low (context) .....	127
4.15.7. transport_lbtrdma_receiver_thread_behavior (context).....	127
4.15.8. transport_lbtrdma_transmission_window_size (source) .....	128
4.16. Transport Acceleration Options .....	128
4.16.1. dbl_lbtrm_acceleration (context).....	129
4.16.2. dbl_lbtru_acceleration (context).....	129
4.16.3. dbl_mim_acceleration (context).....	130
4.16.4. dbl_resolver_acceleration (context).....	130
4.16.5. resolver_ud_acceleration (context).....	130
4.16.6. ud_acceleration (context).....	131
4.17. Multicast Immediate Messaging Network Options.....	131
4.17.1. mim_address (context).....	132
4.17.2. mim_destination_port (context).....	132
4.17.3. mim_incoming_address (context) .....	132
4.17.4. mim_incoming_destination_port (context) .....	132
4.17.5. mim_outgoing_address (context) .....	133
4.17.6. mim_outgoing_destination_port (context) .....	133
4.18. Multicast Immediate Messaging Reliability Options.....	133
4.18.1. mim_ignore_interval (context) .....	133
4.18.2. mim_nak_backoff_interval (context).....	134
4.18.3. mim_nak_generation_interval (context).....	134
4.18.4. mim_nak_initial_backoff_interval (context) .....	134
4.18.5. mim_nak_suppress_interval (context) .....	135
4.18.6. mim_send_naks (context) .....	135
4.18.7. mim_transmission_window_limit (context) .....	135
4.18.8. mim_transmission_window_size (context) .....	135
4.19. Multicast Immediate Messaging Operation Options.....	136
4.19.1. immediate_message_receiver_function (context) .....	136
4.19.2. immediate_message_topic_receiver_function (context) .....	136
4.19.3. mim_activity_timeout (context) .....	136
4.19.4. mim_delivery_control_activity_check_interval (context).....	137
4.19.5. mim_delivery_control_activity_timeout (context) .....	137
4.19.6. mim_delivery_control_order_tablesz (context).....	137
4.19.7. mim_implicit_batching_interval (context) .....	138
4.19.8. mim_implicit_batching_minimum_length (context).....	138
4.19.9. mim_ordered_delivery (context) .....	138

4.19.10. mim_sm_maximum_interval (context).....	139
4.19.11. mim_sm_minimum_interval (context) .....	139
4.19.12. mim_sqn_window_increment (context) .....	139
4.19.13. mim_sqn_window_size (context) .....	139
4.19.14. mim_src_deletion_timeout (context) .....	140
4.19.15. mim_tgsz (context) .....	140
4.19.16. mim_unrecoverable_loss_function (context) .....	140
4.20. Late Join Options .....	141
4.20.1. Late Join Recovery .....	141
4.20.2. late_join (source) .....	142
4.20.3. retransmit_initial_sequence_number_request (receiver) .....	142
4.20.4. retransmit_message_caching_proximity (receiver) .....	143
4.20.5. retransmit_request_generation_interval (receiver) .....	143
4.20.6. retransmit_request_interval (receiver) .....	143
4.20.7. retransmit_request_maximum (receiver) .....	144
4.20.8. retransmit_request_outstanding_maximum (receiver) .....	144
4.20.9. retransmit_retention_age_threshold (source) .....	144
4.20.10. retransmit_retention_size_limit (source) .....	145
4.20.11. retransmit_retention_size_threshold (source) .....	145
4.20.12. use_late_join (receiver) .....	145
4.21. Off-Transport Recovery Options .....	146
4.21.1. otr_request_duration (receiver) .....	146
4.21.2. otr_request_initial_delay (receiver) .....	146
4.21.3. otr_request_log_alert_cooldown (receiver) .....	146
4.21.4. otr_request_maximum_interval (receiver) .....	147
4.21.5. otr_request_minimum_interval (receiver) .....	147
4.21.6. otr_request_outstanding_maximum (receiver) .....	147
4.21.7. use_otr (receiver) .....	148
4.22. Request Network Options .....	148
4.22.1. request_tcp_bind_request_port (context) .....	148
4.22.2. request_tcp_interface (context) .....	149
4.22.3. request_tcp_port (context) .....	149
4.22.4. request_tcp_port_high (context) .....	149
4.22.5. request_tcp_port_low (context) .....	150
4.23. Request Operation Options .....	150
4.23.1. request_tcp_exclusiveaddr (context) .....	150
4.23.2. request_tcp_listen_backlog (context) .....	150
4.23.3. request_tcp_reuseaddr (context) .....	151
4.24. Response Operation Options .....	151
4.24.1. response_session_maximum_buffer (context) .....	151
4.24.2. response_session_sender_socket_buffer (context) .....	151
4.24.3. response_tcp_deletion_timeout (context) .....	152
4.24.4. response_tcp_interface (context) .....	152
4.24.5. response_tcp_nodelay (context) .....	152
4.25. Implicit Batching Options .....	153
4.25.1. implicit_batching_interval (source) .....	153
4.25.2. implicit_batching_minimum_length (source) .....	153
4.25.3. implicit_batching_type (source) .....	153

4.26. Delivery Control Options .....	154
4.26.1. channel_map_tablesz (receiver) .....	156
4.26.2. delivery_control_loss_check_interval (receiver) .....	156
4.26.3. delivery_control_loss_tablesz (receiver) .....	157
4.26.4. delivery_control_maximum_burst_loss (receiver) .....	157
4.26.5. delivery_control_maximum_total_map_entries (context).....	157
4.26.6. delivery_control_order_tablesz (receiver) .....	157
4.26.7. mim_delivery_control_loss_check_interval (context).....	158
4.26.8. null_channel_behavior (receiver) .....	158
4.26.9. source_notification_function (receiver).....	158
4.26.10. unrecognized_channel_behavior (receiver) .....	159
4.27. Wildcard Receiver Options .....	159
4.27.1. pattern_callback (wildcard_receiver) .....	159
4.27.2. pattern_type (wildcard_receiver) .....	160
4.27.3. receiver_create_callback (wildcard_receiver) .....	160
4.27.4. receiver_delete_callback (wildcard_receiver) .....	161
4.27.5. resolver_no_source_linger_timeout (wildcard_receiver) .....	161
4.27.6. resolver_query_maximum_interval (wildcard_receiver).....	161
4.27.7. resolver_query_minimum_duration (wildcard_receiver) .....	161
4.27.8. resolver_query_minimum_interval (wildcard_receiver) .....	162
4.27.9. resolver_wildcard_queries_per_second (context) .....	162
4.27.10. resolver_wildcard_query_bps (context) .....	162
4.27.11. resolver_wildcard_receiver_map_tablesz (context) .....	163
4.28. Event Queue Options.....	163
4.28.1. event_queue_name (event_queue).....	163
4.28.2. queue_age_enabled (event_queue) .....	163
4.28.3. queue_cancellation_callbacks_enabled (event_queue) .....	164
4.28.4. queue_count_enabled (event_queue).....	164
4.28.5. queue_delay_warning (event_queue) .....	164
4.28.6. queue_enqueue_notification (event_queue) .....	165
4.28.7. queue_objects_purged_on_close (event_queue) .....	165
4.28.8. queue_service_time_enabled (event_queue) .....	166
4.28.9. queue_size_warning (event_queue).....	166
4.29. Ultra Messaging Persistence Options.....	166
4.29.1. ume_ack_batching_interval (context).....	166
4.29.2. ume_activity_timeout (receiver).....	166
4.29.3. ume_activity_timeout (source) .....	167
4.29.4. ume_allow_confirmed_delivery (receiver) .....	167
4.29.5. ume_confirmed_delivery_notification (source) .....	168
4.29.6. ume_consensus_sequence_number_behavior (receiver) .....	168
4.29.7. ume_consensus_sequence_number_behavior (source) .....	169
4.29.8. ume_explicit_ack_only (receiver) .....	169
4.29.9. ume_flight_size (source).....	170
4.29.10. ume_flight_size_behavior (source).....	170
4.29.11. ume_flight_size_bytes (source) .....	170
4.29.12. ume_force_reclaim_function (source).....	171
4.29.13. ume_late_join (source) .....	171
4.29.14. ume_message_stability_notification (source).....	172

4.29.15. ume_proxy_source (source).....	172
4.29.16. ume_receiver_liveness_interval (context) .....	173
4.29.17. ume_receiver_paced_persistence (receiver) .....	173
4.29.18. ume_receiver_paced_persistence (source) .....	173
4.29.19. ume_recovery_sequence_number_info_function (receiver).....	174
4.29.20. ume_registration_extended_function (receiver) .....	174
4.29.21. ume_registration_function (receiver).....	174
4.29.22. ume_registration_interval (receiver).....	175
4.29.23. ume_registration_interval (source) .....	175
4.29.24. ume_repository_ack_on_reception (source).....	175
4.29.25. ume_repository_disk_file_size_limit (source).....	176
4.29.26. ume_repository_size_limit (source) .....	176
4.29.27. ume_repository_size_threshold (source).....	177
4.29.28. ume_retention_intergroup_stability_behavior (source).....	177
4.29.29. ume_retention_intragroup_stability_behavior (source).....	178
4.29.30. ume_retention_size_limit (source) .....	178
4.29.31. ume_retention_size_threshold (source) .....	179
4.29.32. ume_retention_unique_confirmations (source) .....	179
4.29.33. ume_retransmit_request_generation_interval (receiver) .....	179
4.29.34. ume_retransmit_request_interval (receiver) .....	179
4.29.35. ume_retransmit_request_maximum (receiver) .....	180
4.29.36. ume_retransmit_request_outstanding_maximum (receiver) .....	180
4.29.37. ume_session_id (context) .....	180
4.29.38. ume_session_id (receiver) .....	181
4.29.39. ume_session_id (source).....	181
4.29.40. ume_source_liveness_timeout (context).....	182
4.29.41. ume_state_lifetime (receiver) .....	182
4.29.42. ume_state_lifetime (source).....	182
4.29.43. ume_store (source).....	182
4.29.44. ume_store_activity_timeout (source) .....	183
4.29.45. ume_store_behavior (source).....	183
4.29.46. ume_store_check_interval (source) .....	184
4.29.47. ume_store_group (source) .....	184
4.29.48. ume_store_name (source).....	184
4.29.49. ume_use_ack_batching (receiver) .....	185
4.29.50. ume_use_late_join (receiver).....	185
4.29.51. ume_use_store (receiver).....	185
4.29.52. ume_user_receiver_registration_id (context) .....	186
4.29.53. ume_write_delay (source) .....	186
4.30. Ultra Messaging Queuing Options.....	186
4.30.1. umq_command_interval (context).....	187
4.30.2. umq_command_outstanding_maximum (context) .....	187
4.30.3. umq_delayed_consumption_report_interval (receiver) .....	187
4.30.4. umq_flight_size (context) .....	187
4.30.5. umq_flight_size (source) .....	188
4.30.6. umq_flight_size_behavior (context) .....	188
4.30.7. umq_flight_size_behavior (source).....	189
4.30.8. umq_hold_interval (receiver).....	189

4.30.9. umq_index_assignment_eligibility_default (receiver) .....	189
4.30.10. umq_message_retransmission_interval (context) .....	190
4.30.11. umq_message_stability_notification (context) .....	190
4.30.12. umq_message_stability_notification (source) .....	191
4.30.13. umq_msg_total_lifetime (context) .....	191
4.30.14. umq_msg_total_lifetime (source) .....	191
4.30.15. umq_queue_activity_timeout (context) .....	192
4.30.16. umq_queue_check_interval (context) .....	192
4.30.17. umq_queue_name (source) .....	192
4.30.18. umq_queue_participants_only (source) .....	193
4.30.19. umq_queue_participation (receiver) .....	193
4.30.20. umq_queue_query_interval (context) .....	194
4.30.21. umq_queue_registration_id (context) .....	194
4.30.22. umq_receiver_type_id (receiver) .....	194
4.30.23. umq_require_queue_authentication (context) .....	195
4.30.24. umq_retention_intergroup_stability_behavior (context) .....	195
4.30.25. umq_retention_intergroup_stability_behavior (source) .....	196
4.30.26. umq_retention_intragroup_stability_behavior (context) .....	196
4.30.27. umq_retention_intragroup_stability_behavior (source) .....	197
4.30.28. umq_retransmit_request_interval (receiver) .....	197
4.30.29. umq_retransmit_request_outstanding_maximum (receiver) .....	198
4.30.30. umq_session_id (context) .....	198
4.30.31. umq_ulb_application_set (source) .....	198
4.30.32. umq_ulb_application_set_assignment_function (source) .....	199
4.30.33. umq_ulb_application_set_events (source) .....	199
4.30.34. umq_ulb_application_set_load_factor_behavior (source) .....	199
4.30.35. umq_ulb_application_set_message_lifetime (source) .....	200
4.30.36. umq_ulb_application_set_message_max_reassignments (source) .....	200
4.30.37. umq_ulb_application_set_message_reassignment_timeout (source) .....	201
4.30.38. umq_ulb_application_set_receiver_activity_timeout (source) .....	201
4.30.39. umq_ulb_application_set_receiver_heartbeat_interval (source) .....	201
4.30.40. umq_ulb_application_set_round_robin_bias (source) .....	202
4.30.41. umq_ulb_check_interval (source) .....	202
4.30.42. umq_ulb_events (source) .....	202
4.30.43. umq_ulb_flight_size (source) .....	203
4.30.44. umq_ulb_flight_size_behavior (source) .....	203
4.30.45. umq_ulb_receiver_events (source) .....	204
4.30.46. umq_ulb_receiver_portion (source) .....	204
4.30.47. umq_ulb_receiver_priority (source) .....	204
4.30.48. umq_ulb_source_activity_timeout (receiver) .....	205
4.30.49. umq_ulb_source_check_interval (receiver) .....	205
4.31. Ultra Messaging JMS Options .....	205
4.31.1. client_id (ConnectionFactory) .....	206
4.31.2. context_index (ConnectionFactory) .....	206
4.31.3. create_queue_browser_context (ConnectionFactory) .....	206
4.31.4. debug (ConnectionFactory) .....	206
4.31.5. default_message_type (ConnectionFactory) .....	207
4.31.6. default_temp_topic_type (ConnectionFactory) .....	207

4.31.7. default_topic_type (ConnectionFactory) .....	208
4.31.8. DestType (Destination) .....	208
4.31.9. queue_browser_creation_delay (ConnectionFactory) .....	208
4.31.10. queue_browser_timeout (ConnectionFactory) .....	209
4.31.11. RECEIVER_CREATION_DELAY (ConnectionFactory) .....	209
4.31.12. regid (Destination) .....	209
4.31.13. SOURCE_CREATION_DELAY (ConnectionFactory) .....	210
4.31.14. SOURCE_REGISTRATION_TIMEOUT (ConnectionFactory) .....	210
4.31.15. Topic (Destination) .....	210
4.31.16. Type (Destination) .....	211
4.31.17. use_app_header (ConnectionFactory) .....	211
4.31.18. use_index_queuing (ConnectionFactory) .....	211
4.31.19. use_ump_session_ids (ConnectionFactory) .....	212
4.31.20. wait_for_source_registration (ConnectionFactory) .....	212
4.31.21. Wildcard (Destination) .....	213
4.32. Hot Failover Operation Options .....	213
4.32.1. delivery_control_loss_check_interval (hfx) .....	213
4.32.2. delivery_control_max_delay (hfx) .....	213
4.32.3. delivery_control_maximum_burst_loss (hfx) .....	214
4.32.4. delivery_control_maximum_total_map_entries (hfx) .....	214
4.32.5. duplicate_delivery (hfx) .....	214
4.32.6. hf_duplicate_delivery (receiver) .....	215
4.32.7. hf_optional_messages (receiver) .....	215
4.32.8. hf_receiver (wildcard_receiver) .....	216
4.32.9. ordered_delivery (hfx) .....	216
4.33. Automatic Monitoring Options .....	216
4.33.1. monitor_appid (context) .....	217
4.33.2. monitor_appid (event_queue) .....	217
4.33.3. monitor_interval (context) .....	217
4.33.4. monitor_interval (event_queue) .....	218
4.33.5. monitor_transport (context) .....	218
4.33.6. monitor_transport (event_queue) .....	219
4.33.7. monitor_transport_opts (context) .....	219
4.33.8. monitor_transport_opts (event_queue) .....	219
4.34. Deprecated Options .....	220
4.34.1. resolver_active_source_interval (context) .....	220
4.34.2. resolver_active_threshold (context) .....	220
4.34.3. resolver_maximum_advertisements (context) .....	221
4.34.4. resolver_maximum_queries (context) .....	221
4.34.5. resolver_query_interval (context) .....	221
4.34.6. resolver_query_max_interval (wildcard_receiver) .....	222
4.34.7. resolver_unicast_address (context) .....	222
4.34.8. resolver_unicast_destination_port (context) .....	222
4.34.9. resolver_unicast_port (context) .....	223
4.34.10. retransmit_message_map_tablesz (source) .....	223
4.34.11. transport_datagram_max_size (context) .....	223
4.34.12. transport_lbtipc_acknowledgement_interval (receiver) .....	224
4.34.13. transport_lbtipc_client_activity_timeout (source) .....	224

4.34.14. ume_message_map_tablesz (source).....	224
4.34.15. ume_primary_store_address (source).....	225
4.34.16. ume_primary_store_port (source) .....	225
4.34.17. ume_registration_id (source) .....	225
4.34.18. ume_secondary_store_address (source) .....	226
4.34.19. ume_secondary_store_port (source).....	226
4.34.20. ume_tertiary_store_address (source) .....	226
4.34.21. ume_tertiary_store_port (source).....	227
4.35. <b>UMS</b> Port Values.....	227
4.35.1. <b>UMS</b> UDP Port Values .....	227
4.35.2. <b>UMS</b> TCP Port Values.....	228
4.36. <b>UMS</b> Multicast Group Values .....	228
4.37. <b>UMS</b> Timer Interval Values .....	229
4.38. Options That May Be Set During Operation.....	232
4.39. Options (Callbacks) That Cannot Be Set From a UM Configuration File .....	233
<b>Index</b> .....	<b>234</b>



# List of Tables

1-1. UM Objects and Corresponding Attributes Objects.....	27
1-2. UM API Functions For Working With <code>lbm_context_attr_t</code> Attributes Objects .....	27
1-3. UM API Functions For Working With <code>lbm_event_queue_t</code> Objects .....	33
1-4. UM API Functions For Retrieving Option Values from <code>lbm_event_queue_t</code> Objects.....	34
4-1. Default UMS UDP Port Values .....	227
4-2. Default UMS TCP Port Values.....	228
4-3. Default UMS Multicast Group Values .....	228
4-4. Default UMS Timer Interval Values.....	229
4-5. Options That May Be Set During Operation.....	232
4-6. Options That Cannot Be Set From a UM Configuration File.....	233

# Chapter 1. Configuring Ultra Messaging Options

**Ultra Messaging® (UM)** offers configuration options that let you set a variety of operational parameters for customization to your needs. These options can reside in configuration files, or can be set individually using function calls. Option values can be assigned to objects upon or after object creation. Within an object, the implemented option values are referred to as attributes.

## 1.1. Overview

**UM** uses intelligent default values for each of its configuration options, enabling it to run reasonably well "out of the box." However, expect to customize Ultra Messaging options to optimize your operating environment. To provide maximum flexibility, **UM** offers several ways to configure option default and customized value value assignments.

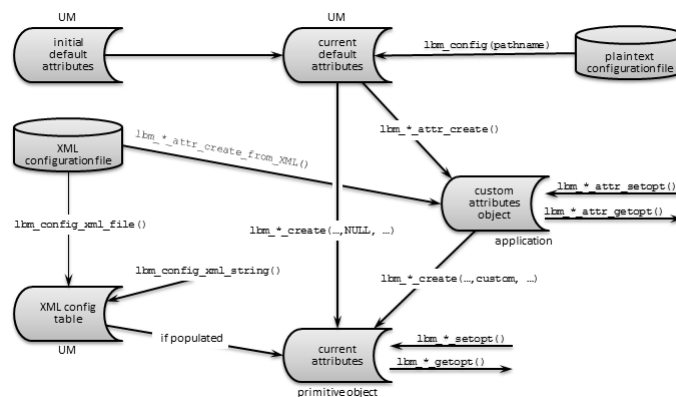
### 1.1.1. Assignment Methods

**UM** provides four basic methods for using configuration options to set attributes:

- XML configuration files - customized defaults used during object creation
- plain text configuration files - a different manner of customized defaults, also used during object creation
- attributes objects - application-specific option values used during object creation
- function calls (setopt) - used after object creation

Figure 1-1 shows the different ways option values are stored and assigned before, during, and after primitive object creation (primitive objects being sources, receivers, wildcard receivers, event queues, contexts, or HFX objects). The ultimate result is a primitive object created with the desired values residing in current attributes.

**Figure 1-1. Attributes value assignment methods**



The *initial default attributes* is the set of factory defaults residing in **UM**. The *current default attributes* are derived from the initial default attributes, but modified by option setting in the *plain text configuration file*. This is the starting point for all created primitive objects.

An instantiated primitive object has its attribute settings held in *current attributes*, which are influenced by the *current default attributes*, the *XML config table*, and, if applicable, any settings stored in a *custom attributes objects*.

An *XML configuration file* can pass its setting to an object being created in two ways: by directly populating **UM**'s *XML config table*, or by creating a *custom attributes object* with its settings.

## 1.1.2. Assignment Flow

The above diagram implies, but does not fully explain, the flow of attribute value assignment that **UM** performs when an application creates a primitive object. This flow is described below, and is important in understanding how and when default values are overridden:

1. If applicable, copy plain text configuration file values to current default attributes.
2. Start creating object.
3. Custom attributes object(s) created/populated (if applicable).
4. If `lbm_*_create()` has a NULL attr, copy *current default attributes* into *current attributes*. Otherwise, copy *custom attributes object* values into *current attributes*.
5. Read applicable options from the *XML config table* into the *current attributes*. Do not overwrite options set with `lbm_config()`, or `lbm_*_attr_setopt()`, which were tagged when modified.
6. Finish object creation.
7. *current attributes* can be changed further (only certain options) via `lbm_*_setattr()`.

## 1.1.3. Definitions

Before discussing how **UM** options can be set, some terminology is in order.

- **Option** - A single configuration item that controls some aspect of **UM** operation. An option typically resides in a configuration file, but can also be assigned a value via a function call. We use options to assign values to an object's attributes.
- **Attribute** - An operational characteristic of an object. An attribute's value is set by an option, hence, there is a one-to-one correspondence between options and attributes. (Note: This use of the term "attribute" is unrelated to, and not to be confused with, "attribute" in XML syntax. In this document, we refer to the latter as "XML attribute".)
- **XML attribute** - See above. In XML syntax, XML attributes are parameters for XML elements.
- **Custom attributes object** A **UM** object that contains custom attribute values (set by options) for a specific **UM** object. Separate (and multiple) sets of attributes can exist for each application, though only one can be used when creating a primitive object.
- **Initial default attributes** - The default attributes values built into **UM**. **UM** and your applications use these if you have not set any options for the attributes.

- **Primitive object** - Specifically, an object that is a source, receiver, wildcard receiver, event queue, context, or HFX object.
- **Configuration file** - This comes in two types: XML and plain text. Configuration files contain assigned values for options, but the different types are read/copied at different times during the creation of an object.
- **XML config table** - Contains option values that are read from the XML configuration file.
- **Current default attributes** - The attributes values used to create an object in the absence of custom attributes values.
- **Current attributes** - The attribute values for an instantiated **UM** object that control the current operation of that object.
- **Scope** - The type of object to which an option can apply. Possible scopes are context, source, receiver, wildcard\_receiver, event\_queue, and hfx.

### 1.1.4. Which Method Should I Use?

For the four basic assignment methods listed above, following are some scenarios where specific methods are selected.

- To **change a default option value** and apply it to all objects you create, call `lbm_config()` for one or more configuration files. For example, to use LBT-RM rather than TCP for all sources, create a plain text configuration file containing

```
source transport LBTRM
```

and pass its file name to `lbm_config()`.

**Note:** The C API ([./API/index.html](#)) offers functions `lbm*_attr_create_default()` to change a current default value back to the initial (factory) default value. No such corresponding method exists for the Java or .NET APIs.

- To **customize specific options before an object is created** for a specific object instance, use a custom attributes object. Also, you can assign XML data to the XML config table directly from your application via `lbm_config_xml_string()`.
- To **create sets of custom values** to be used when creating primitive objects, call `lbm_config_xml_file()` and specify an XML configuration file. This is useful for setting specific default options on a per-topic or per-context basis, which cannot be done with a plain text configuration file. For an example where a sending application uses specific options and values, create an XML configuration file with the application's name (optional) that specifies those options and values. Then pass the XML file name and application name to `lbm_config_xml_file()`.
- To change an option **after an object is created**, modify the current attributes for the object. (Note that many options cannot be changed after an object has been created.)

These methods can be used in combination. Figure 1-1 illustrates the relationships between attributes and the various **UM** API function calls that affect them.

## 1.1.5. Configuration Files

There are two types of **UM** Configuration files:

- *Plain Text Configuration Files*
- *XML Configuration Files*

You can read Configuration files either by function call, or automatically upon application launch by specifying a file name in an environment variable. See Figure 1-1 and *Assignment Flow* for details on how these options replace or override default values.

## 1.2. Plain Text Configuration Files

The plain text configuration file, when invoked, writes option values into **UM**'s current default attributes. These are then read and used in the creation of all objects.

See Example Configuration Scenarios (`../Config/examples.html`) for example configuration files.

### 1.2.1. Reading Plain Text Configuration Files

There are two ways to read a plain text configuration file to set values in current default attributes.

- **API function** `lbm_config()` - You can call the function multiple times with different file names to set configuration options in phases.

When you create **UM** objects (such as a context or receiver), **UM** sets attributes for that object using the current default attributes. Hence, you must call `lbm_config()` before creating objects (`lbm*_create()`).

- **Environment variable** `LBM_DEFAULT_CONFIG_FILE` - reads configuration file when your application is started. You can set this variable to a full pathname or a URL; for example:

```
export LBM_DEFAULT_CONFIG_FILE=/home/lbm/lbtrm.cfg
```

(You can still use the `lbm_config()` function on a different file to make additional changes.)

### 1.2.2. Plain Text Configuration File Format

A plain text configuration file contains lines that each take the form

```
scope_keyword option_name option_value
```

where

*scope\_keyword* - the scope to which the option applies,

*option\_name* - the predefined name for the option, and

*option\_value* - the new value to be assigned to that option.

Allowable values for these parameters are given throughout the rest of this document. Any text following a hash character # (also known as a pound sign, number sign, or octothorpe) is interpreted as comment text and is ignored.

For example:

```
# Set transport_tcp_port_low to 4901
context transport_tcp_port_low 4901
# And set transport_tcp_port_high to 4920
context transport_tcp_port_high 4920
```

## 1.3. XML Configuration Files

XML configuration files let you address many different applications and operating requirements, removing the need to programmatically set and reset options for them. A single XML file can contain options for multiple applications. Moreover, for a single application, you can configure multiple named contexts, event queues, etc., with different values for the same options.

See Example Configuration Scenarios ([../Config/examples.html](#)) for example configuration files.

### 1.3.1. Reading XML Configuration Files

There are several ways to read an XML configuration file to assign values while creating a primitive object.

- **API function** `lbm_config_xml_file()` - reads an XML configuration file into **UM's XML config table**. Call this before the primitive create function. This does not change the current default attributes.
- **API function** `lbm_config_xml_string()` - populates the XML config table directly from your application. Call this before the primitive create function. This does not change the current default attributes.
- **API function** `lbm*_attr_create_from_XML()` - creates a custom attributes object containing the values from an XML configuration file. The values can then be applied to a primitive object being created by calling function `lbm*_create()` and specifying this custom attributes object in the second parameter.
- **Environment variable** `LBM_XML_CONFIG_FILENAME` - reads the file into **UM's XML config table**. These settings are then available to all applications when they start.

```
export LBM_XML_CONFIG_FILENAME=filename
```

- **Environment variable** `LBM_XML_CONFIG_APPNAME` - reads options for a specific application from the above variable's filename. This initiates the specified application's configuration; set this environment variable for every application.

```
export LBM_XML_CONFIG_APPNAME=application_name
```

- **Environment variable** `LBM_UMM_INFO` - initiates UMM Daemon to read options for an application and user from the above variable's filename. Set this variable for every application/user combination, in this format:

```
export LBM_UMM_INFO=application_name:user_name:password@ip:port
```

**Note:** Since these API functions and environment variables can be used without the UMM Daemon, no username or password can be set.

## 1.3.2. Using XML Configuration Files With a UM Application

The following procedure describes a general approach to implementing XML configuration files.

1. Create an XML configuration file using an XML editor or text editor. Just for this example, name the file, `UM_CONFIG.XML`.
2. Insert any desired templates in the `<templates>` element to hold configuration option values shared by multiple applications or primitive **UM** objects (context, source, receiver, wildcard receiver or event queue). You can create and apply multiple templates to applications and primitive **UM** objects, however, if the same option appears in multiple templates, the option value in the last template overrides the option value in the previous template. See `<templates>`.
3. Insert an `<application>` element for your **UM** application in the `<applications>` element and include any relevant templates created in the previous step. Just for this example, name the application, `SENDAPP`. See `<applications>`.
4. Within the `<Contexts>` element, configure the application's `<Context>` element and context options. And since our example application, `SENDAPP` is a sending application, also configure its Source options. (If this was a receiving application, you would configure Receiver or Wildcard Receiver options. If your application creates multiple Contexts, enter multiple `<Context>` elements within the Contexts element, inserting the appropriate source, receiver or wildcard receiver options. See `<contexts>`.
5. Configure the applications Event Queue options. See `<event-queues>`
6. Save the XML configuration file, `UM_CONFIG.XML`, and load it onto the machine where the application (`SENDAPP`) runs.
7. Set the following environment variables on the machine where `SENDAPP` runs.
  - Set `LBM_XML_CONFIG_FILENAME` to `UM_CONFIG.XML`.
  - Set `LBM_XML_CONFIG_APPNAME` to `SENDAPP`.
  - Optionally, you could also use `lbm_config_xml_file (UM_CONFIG.XML, SENDAPP)` in the `SENDAPP` source.
8. Start `SENDAPP`.

## 1.3.3. XML Configuration File Format

An XML Configuration File follows standard XML conventions. Element declarations or a pointer to a DTD file are not needed, as these are handled by **UM**.

An XML configuration file generally comprises two primary elements: `templates` and `applications`. Organized and contained within these are option value assignments. Applications containers let you set options for specific

applications. To provide more global control over applications, or to simply reduce repetition, you can create templates to hold option settings that are to be used in one or more different applications.

XML configuration files use the high-level structure shown in the following example. This example includes only some container elements, and no options.

```
<um-configuration version="1.0"/>
  <templates>
    <template name="SENDING">
      <options type="source">
      </options>
      <options type="context">
      </options>
    </template>
  </templates>
  <applications>
    <application name="SENDING-TOPIC1">
      <contexts>
        <context name="SENDING-LBTRM">
          <sources>
            <topic topicname="TOPIC1">
              <options type="source">
              </options>
            </topic>
          </sources>
        </context>
      </contexts>
      <event-queues>
        <event-queue/>
        <event-queue name="EQ-1"/>
      </event-queues>
    </application>
  </applications>
</um-configuration>
```

Following are descriptions of the XML configuration file elements.

- *<um-configuration>*
- *<license>*
- *<options>*
- *<option>*
- *<allow>*
- *<deny>*
- *<templates>*
- *<template>*
- *<applications>*
- *<application>*



- `<contexts>`
- `<context>`
- `<sources>`
- `<topic>`
- `<receivers>`
- `<wildcard-receivers>`
- `<wildcard-receiver>`
- `<event-queues>`
- `<event-queue>`
- `<hfxs>`
- `<application-data>`

See also *Sample XML Configuration File* and *XML Configuration File DTD*.

### 1.3.3.1. `<um-configuration>`

**Description.** The `<um-configuration>` element is a required container for all **UM** configuration options residing in the XML configuration file. This is the top-level element.

**Parents.** None.

**Children.** `<templates>` `<applications>` `<license>`

**XML Attributes:**

XML Attribute	Description	Default Value
version	The version of the DTD.	none

**Example:**

```
<um-configuration version="1.0">
  <applications>
    <application>
      . . .
    </application>
  </applications>
</um-configuration>
```

#### 1.3.3.1.1. `<license>`

**Description.** The `<license>` element identifies the **UM** product license, either as the license key or as a pointer to a license file, as an alternative to setting it in an environment variable.

**Parents.** `<um-configuration>`

**Children.** None.

**XML Attributes:**

XML Attribute	Description	Default Value
format	The format for the license element data. <code>filename</code> points to the file containing the license key. <code>string</code> identifies the data as the license key itself.	string
xml:space	How whitespace is handled. <code>default</code> trims leading and trailing whitespace (e.g., tabs, spaces, linefeeds, etc.), and compresses multiple whitespace characters into a single space character. <code>preserve</code> preserves the whitespace exactly as read.	default

**Example:**

```
<um-configuration>
  <license format=filename>
    path/license-file-name
  </license>
  <applications>
    <application>
      . . .
```

**1.3.3.2. <options>**

**Description.** The `<options>` element is a container element for individual options. You specify the primitive object in the attribute `type`.

**Parents.** `<template>` `<context>` `<topic>` `<wildcard-receiver>` `<event-queue>`

**Children.** `<option>` `<application-data>`

**XML Attributes:**

XML Attribute	Description	Default Value
type	The type of primitive object, which can be <code>event-queue</code> , <code>context</code> , <code>source</code> , <code>receiver</code> , <code>wildcard-receiver</code> , or <code>hfx</code> ).	None

**Example:**

```
<options type="context">
  <option/>
  . . .
  <application-data/>
</options>
```

**1.3.3.2.1. <option>**

**Description.** The `<option>` element corresponds to any **UM** configuration option.

**Parents.** `<options>`

**Children.** `<allow>` `<deny>`

**XML Attributes:**

XML Attribute	Description	Default Value
name	Name of the <b>UM</b> configuration option. See Reference ( <code>../Config/reference.html</code> ) for all options.	N/A
default-value	The value you are setting for this option.	The default value for the option.
order	Permit or restrict particular option values. Valid values are <code>deny, allow</code> (deny what you specify, allow everything else) or <code>allow, deny</code> (allow what you specify, deny everything else). If using this XML attribute, follow this element with <code>&lt;allow&gt;</code> or <code>&lt;deny&gt;</code> elements as needed. See also <i>Sample XML Configuration File</i> .	<code>deny, allow</code>

#### Examples:

To permit any application to choose any transport method except LBT-RU, configure the following in a template included in sending applications.

```
<option default-value="tcp" name="transport" order="deny, allow">
  <deny>LBTRU</deny>
</option>
```

To restrict any application to only the LBT-RM or LBR-RU transport method, configure the following in a template included in sending applications.

```
<option default-value="tcp" name="transport" order="allow, deny">
  <allow>LBTRU</allow>
  <allow>LBTRM</allow>
</option>
```

#### 1.3.3.2.2. `<allow>`

**Description.** Use the `<allow>` element with `<option>` to set a condition for for that option to permit only a certain subset of possible default value values for the option. See also *Using the Order and Rule XML Attributes*.

**Parents.** `<option>`

**Children.** None.

**XML Attributes:**

XML Attribute	Description	Default Value
---------------	-------------	---------------

XML Attribute	Description	Default Value
xml:space	How whitespace is handled. <code>default</code> trims leading and trailing whitespace (e.g., tabs, spaces, linefeeds, etc.), and compresses multiple whitespace characters into a single space character. <code>preserve</code> preserves the whitespace exactly as read.	default

**Example:**

```
<option default-value="tcp" name="transport" order="allow,deny">
  <allow>LBTRU</allow>
  <allow>LBTRM</allow>
</option>
```

**1.3.3.2.3. <deny>**

**Description.** Use the `<deny>` element with `<option>` to set a condition for that option that restricts certain (otherwise) possible `default` value values from being used by the option. See also *Using the Order and Rule XML Attributes*.

**Parents.** `<option>`

**Children.** None.

**XML Attributes:**

XML Attribute	Description	Default Value
xml:space	How whitespace is handled. <code>default</code> trims leading and trailing whitespace (e.g., tabs, spaces, linefeeds, etc.), and compresses multiple whitespace characters into a single space character. <code>preserve</code> preserves the whitespace exactly as read.	default

**Example:**

```
<option default-value="tcp" name="transport" order="deny,allow">
  <deny>LBTRU</deny>
</option>
```

**1.3.3.3. <templates>**

**Description.** The `<templates>` element is a container element for all templates that contain configuration options that can be used in other templates or applications. A template can be very specific, such as configuring options only for LBT-RM sources, or more comprehensive, configuring common options for your applications.

Insert any desired templates in the `<templates>` element to hold configuration option values shared by multiple applications or primitive objects. You can create and apply multiple templates to applications and primitive UM objects in a comma separated value (CSV) format. However, if the same option appears in multiple templates, the option value in the last or lower-level template overrides the option value in the previous or higher-level template.

**Parents.** `<um-configuration>`

**Children.** `<template>`

**XML Attributes:** None.

**Example:**

```
<templates>
  <template name="SENDING">
    <options/>
  </template>
</templates>
```

#### 1.3.3.3.1. `<template>`

**Description.** The `<template>` element is a container for one uniquely named set of options.

**Parents.** `<templates>`

**Children.** `<options>`

**XML Attributes:**

XML Attribute	Description	Default Value
name	Name of the configuration template, which can be referenced elsewhere in this XML configuration file to assign to other configuration elements. Multiple templates can be specified in a comma separated value (CSV) format.	None

**Example:**

```
<templates>
  <template name="SENDING", name="SENDING-LBTRM">
    <options/>
  </template>
</templates>
```

#### 1.3.3.4. `<applications>`

**Description.** The `<application>` element is a container element for all applications configured in the XML configuration file. **UM** lets you configure one or more applications.

**Parents.** `<um-configuration>`

**Children.** `<application>`

**XML Attributes:** None.

**Example:**

```
<applications>
```

```

<application name="SENDING-IXCM-LBTRM" template="SENDING">
  <contexts/>
  <event-queues/>
  <application-data/>
</application>
</applications>

```

#### 1.3.3.4.1. <application>

**Description.** The `application` element contains option values for all object elements within a single, uniquely named, application.

**Parents.** `<applications>`

**Children.** `<application-data>` `<contexts>` `<event-queues>` `<hfxs>`

**XML Attributes:**

XML Attribute	Description	Default Value
name	Name of the application. Used as an optional parameter for <code>lbm_config_from_xml()</code> . If a name is not supplied, this must be the only occurrence of this element in the XML configuration file.	None
template	Name of the configuration template to use for the application.	None

**Example:**

```

<applications>
  <application name="SENDING-IXCM-LBTRM" template="SENDING">
    <application-data/>
    <contexts/>
    <event-queues/>
  </application>
  <application name="SENDING-IXCM-TCP" template="SENDING">
    <application-data/>
    <contexts/>
    <event-queues/>
  </application>
</applications>

```

#### 1.3.3.5. <contexts>

**Description.** The `<contexts>` element is a container element for all **UM** contexts (`../Design/lbm-objects.html#CONTEXT-OBJECT`) configured for an application. **UM** lets you create one or more contexts for an application.

**Parents.** `<application>`

**Children.** `<context>`.

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to apply to each individual context object configured within this element. Multiple templates can be applied by specifying them in a comma-separated-value manner, i.e., "SENDING1, SENDING2". Can be overridden by a different template configured for an individual context.	None
order	Establishes the permission semantic for each individual context configured within this element. Valid values are <code>deny, allow</code> (deny what you specify, allow everything else) or <code>allow, deny</code> (allow what you specify, deny everything else). Works in conjunction with the <code>&lt;context&gt;</code> XML attribute, <code>rule</code> .	<code>deny, allow</code>

**Example:**

```

<applications>
  <application>
    <contexts template="SENDING" order="deny, allow">
      <context name="SENDING-95" template="SENDING-LBTRM" rule="allow">
        <sources/>
        <receivers/>
        <wildcard-receivers/>
        <options/>
      </context>
    </contexts>
    <event-queues/>
    <application-data/>
  </application>
</applications>

```

**1.3.3.5.1. `<context>`**

**Description.** The `<context>` element contains option values for a single context, organized into its child elements.

**Parents.** `<contexts>`

**Children.** `<sources>` `<receivers>` `<wildcard-receivers>` `<options>`

**XML Attributes:**

XML Attribute	Description	Default Value
name	Name of the context. Used as a parameter for <code>lbm_context_attr_create_from_xml()</code> and <code>lbm_context_attr_set_from_xml()</code> . If no name is supplied, the contained settings are matched with all NULL-named contexts.	None
template	Name of the configuration template to use for the context object's options.	None
rule	Permits or restricts the creation of the context object. If <code>rule="deny"</code> , the context object errors upon creation.	<code>allow</code>

**Example:**

```

<applications>
  <application>
    <contexts template="SENDING" order="deny,allow">
      <context name="SENDING-95" template="SENDING-LBTRM" rule="allow">
        <sources/>
        <receivers/>
        <wildcard-receivers/>
        <options/>
      </context>
    </contexts>
  </application>
</applications>

```

### 1.3.3.6. <sources>

**Description.** The <sources> element is a container for all **UM** sources (../Design/lbm-objects.html#SOURCE-OBJECT) configured for an application. **UM** lets you create one or more sources for an application.

**Parents.** <context>

**Children.** <topic>

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to apply to each individual source object configured within this element. Multiple templates can be applied by specifying them in a comma-separated-value manner, i.e., "SENDING1,SENDING2". Can be overridden by a different template configured for an individual source.	None
order	Establishes the permission semantic for each individual source configured within this element. Valid values are <i>deny,allow</i> (deny what you specify, allow everything else) or <i>allow,deny</i> (allow what you specify, deny everything else). Works in conjunction with the <topic> XML attribute <i>rule</i> .	<i>deny,allow</i>

**Example:**

```

<applications>
  <application>
    <contexts>
      <context>
        <sources template="SENDING" order="deny,allow">
          <topic topicname="ICXM" template="SENDING-LBTRM" rule="allow"/>
        </sources>
        <receivers/>
        <wildcard-receivers/>
        <options/>
      </context>
    </contexts>
  </application>
</applications>

```



```

    </context>
  </contexts>
</event-queues>
</application-data/>
</application>
</applications>

```

#### 1.3.3.6.1. <topic>

**Description.** The <topic> element contains option values for a single source or receiver.

**Parents.** <hfxs> <receivers>. <sources>

**Children.** <options>.

**XML Attributes:**

XML Attribute	Description	Default Value
topicname	The topic string for the topic that the source sends or the receiver accepts. Used as a parameter for <code>lbm_src_topic_alloc()</code> , <code>lbm_rcv_topic_lookup()</code> , <code>lbm_src_attr_create_from_xml()</code> , <code>lbm_src_attr_set_from_xml()</code> , <code>lbm_rcv_attr_create_from_xml()</code> and <code>lbm_rcv_attr_set_from_xml()</code> . Do not use with the <code>pattern</code> attribute.	None
template	Name of the configuration template to use for this topic's source or receiver options.	None
rule	Permits or restricts the creation of the source or receiver object. If <code>rule="deny"</code> , the object errors upon creation.	allow
pattern	Identify the set of options for this topic with a topic string pattern. Any source created with a topic string that matches this pattern receives the configured option values. Do not use with the <code>topicname</code> attribute.	None

**Example:**

```

<applications>
<application name="Sending">
  <contexts order="deny,allow">
    <context rule="allow" template="Sending-LBTRM">
      <sources order="deny,allow">
        <topic rule="allow" topicname="IXCM">
          <options type="source">
            <option default-value="224.12.5.101" name="transport_lbtrm_multicast_address"/>
          </options>
        </topic>
      </sources>
    </context>
  </contexts>
</application>

```

```
</applications>
```

### 1.3.3.7. <receivers>

**Description.** The <receivers> element is a container element for all **UM** receivers (../Design/lbm-objects.html#RECEIVER-OBJECT) configured for an application. You can create one or more receivers for an application.

**Parents.** <context>

**Children.** <topic>

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to apply to each individual receiver object configured within this element. You can apply multiple templates by specifying them in a comma-separated-value manner, e.g., "RECEIVING1,RECEIVING2". A template applied to an individual receiver will override a <receivers>-level template.	None
order	Establishes the permission semantic for each individual receiver configured within this element. Valid values are <i>deny, allow</i> (deny what you specify, allow everything else) or <i>allow, deny</i> (allow what you specify, deny everything else). Works in conjunction with the <topic> XML attribute rule.	<i>deny, allow</i>

**Example:**

```
<applications>
  <application>
    <contexts>
      <context>
        <sources/>
        <receivers template="RECEIVING" order="deny,allow">
          <topic topicname="ICXM" template="RECEIVING" rule="allow"/>
        </receivers>
        <wildcard-receivers/>
        <options/>
      </context>
    <event-queues/>
    <application-data/>
  </application>
</applications>
```

**1.3.3.8. <wildcard-receivers>**

**Description.** The <wildcard-receivers> element is a container element for all **UM** wildcard receivers (../Design/lbm-objects.html#WILDCARD-RECEIVER) configured for an application. **UM** lets you create one or more wildcard receivers for an application.

**Parents.** <context>

**Children.** <wildcard-receiver>.

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to apply to each individual wildcard receiver object configured within this element. Multiple templates can be applied by specifying them in a comma-separated-value manner, i.e., "RECEIVING1, RECEIVING2". Can be overridden by a different template configured for an individual wildcard receiver.	None
order	Establishes the permission semantic for each individual wildcard receiver configured within this element. Valid values are <code>deny, allow</code> (deny what you specify, allow everything else) or <code>allow, deny</code> (allow what you specify, deny everything else). Works in conjunction with the <wildcard-receiver> XML attribute <code>rule</code> .	<code>deny, allow</code>

**Example:**

```
<applications>
  <application>
    <contexts>
      <context>
        <sources>
        <receivers/>
        <wildcard-receivers template="RECEIVING" order="deny, allow">
          <wildcard-receiver template="RECEIVING-LBTRM" rule="allow" pattern="I*M" pattern-type="p
        </wildcard-receivers>
        <options/>
      </context>
    <event-queues/>
    <application-data/>
  </application>
</applications>
```

**1.3.3.8.1. <wildcard-receiver>**

**Description.** The <wildcard-receiver> element contains option values for a single wildcard receiver.

**Parents.** <wildcard-receivers>.

**Children.** <options>.

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to use for the wildcard receiver object's options.	None
rule	Permits or restricts the creation of the wildcard receiver object. If <code>rule="deny"</code> , the object errors upon creation.	allow
pattern	The wildcard receiver topic string pattern for this wildcard receiver object.	None
pattern-type	The type of pattern matching to use for the wildcard receiver object. Valid values are <code>pcre</code> , <code>regex</code> or <code>application-callback</code> .	<code>pcre</code>

**Example:**

```

<applications>
  <application>
    <contexts>
      <context>
        <wildcard-receivers template="RECEIVING" order="deny,allow">
          <wildcard-receiver template="RECEIVING-LBTRM" rule="allow" pattern="I*M" pattern-type="pcre">
        </wildcard-receivers>
      </context>
    </application>
  </applications>

```

**1.3.3.9. <event-queues>**

**Description.** The `<event-queues>` Element is a container element for all **UM** event queues (`../Design/lbm-objects.html#EVENT-QUEUE-OBJECT`) configured for an application. **UM** lets you create one or more event queues for an application.

**Parents.** `<application>`.

**Children.** `<event-queue>`.

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to apply to each individual event queue object configured within this element. You can apply multiple templates specifying them in a comma-separated-value manner, e.g., <code>"EVQ-1, EVQ-2"</code> . A template applied to an individual event queue will override an <code>&lt;event-queues&gt;</code> -level template.	None
order	Establishes the permission semantic for each individual event queue configured within this element. Valid values are <code>deny, allow</code> (deny what you specify, allow everything else) or <code>allow, deny</code> (allow what you specify, deny everything else). Works in conjunction with the <code>&lt;event-queue&gt;</code> XML attribute, <code>rule</code> .	<code>deny, allow</code>

**Example:**

```

<applications>
  <application>
    <contexts/>
    <event-queues template="RECEIVING" order="deny,allow">
      <event-queue name="EVQ-1" template="SENDING-LBTRM" rule="allow"/>
    </event-queues>
    <application-data/>
  </application>
</applications>

```

**1.3.3.9.1. <event-queue>**

**Description.** The <event-queue> element contains option values for a single event queue.

**Parents.** <event-queues>.

**Children.** <options>.

**XML Attributes:**

XML Attribute	Description	Default Value
name	Name of the event queue. Used as a parameter for <code>lbm_event_queue_attr_create_from_xml()</code> and <code>lbm_event_queue_attr_set_from_xml()</code> .	None
template	Name of the configuration template to use for the event queue object's options.	None
rule	Permits or restricts the creation of the event queue object. If <code>rule="deny"</code> , the object errors upon creation.	allow

**Example:**

```

<applications>
  <application>
    <contexts/>
    <event-queues template="RECEIVING" order="deny,allow">
      <event-queue name="EVQ-1" template="SENDING-LBTRM" rule="allow"/>
    </event-queues>
    <application-data/>
  </application>
</applications>

```

**1.3.3.10. <hfxs>**

**Description.** The <hfxs> element is a container for all UM HFX objects (`../Design/lbm-features.html#HOT-FAILOVER`) configured for an application. Within the <hfxs> element, options are organized by topic.

**Parents.** `<application>`

**Children.** `<topic>`

**XML Attributes:**

XML Attribute	Description	Default Value
template	Name of the configuration template to apply to each individual HFX object configured within this element. Multiple templates can be applied by specifying them in a comma-separated-value manner, i.e., "SENDING1, SENDING2". Can be overridden by a different template configured for an individual HFX object.	None
order	Establishes the permission semantic for each individual HFX object configured within this element. Valid values are <code>deny, allow</code> (deny what you specify, allow everything else) or <code>allow, deny</code> (allow what you specify, deny everything else). Works in conjunction with the <code>&lt;topic&gt;</code> XML attribute, <code>rule</code> .	<code>deny, allow</code>

**Example:**

```
<applications>
  <application>
    <hfxs template="SENDING" order="deny, allow">
      <topic topicname="ICXM" template="SENDING-LBTRM" rule="allow"/>
    </hfxs>
  </application>
</applications>
```

### 1.3.3.11. `<application-data>`

**Description.** The `<application-data>` element is a free-form text comment field that you can use to store application-specific or options-group-specific metadata. When defined at the options level, this content overrides `<application-data>` elements defined at the application level.

Your application can retrieve this data via the `lbm_*_attr_getopt()` and `lbm_*_attr_str_getopt()` API functions under the option name `application_data`. You can also programmatically set it using the equivalent `*_setopt()` APIs. The `application_data` option is defined for all option scopes.

Also, you can set or retrieve this value at runtime via the `*_getopt()` and `*_setopt()` functions defined for the following types:

- `lbm_context_t`
- `lbm_src_t`
- `lbm_rcv_t`
- `lbm_wildcard_rcv_t`
- `lbm_event_queue_t`
- `lbm_hfx_t`

**Parents.** `<application>`, `<options>`.

**Children.** None.

**XML Attributes:**

XML Attribute	Description	Default Value
<code>xml:space</code>	How whitespace is handled. <code>default</code> trims leading and trailing whitespace (e.g., tabs, spaces, linefeeds, etc.), and compresses multiple whitespace characters into a single space character. <code>preserve</code> preserves the whitespace exactly as read.	<code>default</code>

**Example:**

```
<applications>
  <application name="SENDING-IXCM-LBTRM" template="SENDING">
    <application-data>
      SENDING-IXCM-LBTRM options application data string
    </application-data>
    <contexts/>
    <options type="context">
      <option/>
      . . .
    </options>
    <event-queues/>
  </application>
</applications>
```

### 1.3.4. Sample XML Configuration File

A sample XML configuration file appears below and has the following notable aspects.

- Contains object attributes for a **UM** context and source.
- Application name is `Sending`.
- Uses a template of attributes also called `Sending-LBTRM`.
- The template, `Sending-LBTRM`, uses the `order` attribute for the `fd_management_type` to allow all file descriptor types except `DEVPOLL`. However the `Sending-LBTRM` application further restricts the file descriptor types to exclude `EPOLL` in addition to `DEVPOLL`.

```
<um-configuration version="1.0">
  <templates>
    <template name="Sending-LBTRM">
      <options type="source">
        <option default-value="0" name="late_join"/>
      </options>
    </template>
  </templates>
</um-configuration>
```

```

<option default-value="500" name="resolver_advertisement_maximum_initial_interval"/>
<option default-value="5000" name="resolver_advertisement_minimum_initial_duration"/>
<option default-value="10" name="resolver_advertisement_minimum_initial_interval"/>
<option default-value="60" name="resolver_advertisement_minimum_sustain_duration"/>
<option default-value="1000" name="resolver_advertisement_sustain_interval"/>
<option default-value="lbtrm" name="transport"/>
<option default-value="14400" name="transport_lbtrm_destination_port"/>
<option default-value="0.0.0.0" name="transport_lbtrm_multicast_address"/>
</options>
<options type="context">
  <option default-value="wsaeventselect" name="fd_management_type" order="deny,allow">
    <deny>wincompport</deny>
  </option>
  <option default-value="5000" name="mim_delivery_control_activity_check_interval"/>
  <option default-value="60000" name="mim_delivery_control_activity_timeout"/>
  <option default-value="6000" name="mim_delivery_control_loss_check_interval"/>
  <option default-value="2000000" name="resolver_initial_advertisement_bps"/>
  <option default-value="2000" name="resolver_initial_advertisements_per_second"/>
  <option default-value="2000" name="resolver_initial_queries_per_second"/>
  <option default-value="2000000" name="resolver_initial_query_bps"/>
</options>
</template>
</templates>
<applications>
<application name="Sending">
  <contexts order="deny,allow">
    <context rule="allow" template="Sending-LBTRM">
      <sources order="deny,allow">
        <topic rule="allow" topicname="IXCM">
          <options type="source">
            <option default-value="1" name="late_join"/>
            <option default-value="lbtrm" name="transport"/>
            <option default-value="14488" name="transport_lbtrm_destination_port"/>
            <option default-value="224.12.5.101" name="transport_lbtrm_multicast_address"/>
          </options>
        </topic>
      </sources>
      <receivers order="deny,allow"/>
      <wildcard-receivers order="deny,allow"/>
      <options type="context">
        <option default-value="224.9.10.11" name="resolver_multicast_address"/>
        <option default-value="224.9.10.11" name="resolver_multicast_incoming_address"/>
        <option default-value="12965" name="resolver_multicast_incoming_port"/>
        <option default-value="224.9.10.11" name="resolver_multicast_outgoing_address"/>
        <option default-value="12965" name="resolver_multicast_outgoing_port"/>
        <option default-value="12965" name="resolver_multicast_port"/>
        <option default-value="224.9.10.12" name="resolver_multicast_interface"/>
        <option default-value="0" name="resolver_multicast_receiver_socket_buffer"/>
        <option default-value="wsaeventselect" name="fd_management_type" order="deny,allow">
          <deny>wincompport</deny>
        </option>
      </options>
    </context>
  </contexts>

```



```

</contexts>
<event-queues order="deny,allow">
  <event-queue rule="allow">
    <options type="event-queue">
      <option default-value="lbm" name="monitor_transport"/>
      <option default-value="" name="monitor_appid"/>
    </options>
  </event-queue>
</event-queues>
</application>
</applications>
</um-configuration>

```

### 1.3.4.1. Using the Order and Rule XML Attributes

The `order` and `rule` XML attributes combine to enable you to permit or restrict the creation of primitive **UM** objects. The container elements such as the `<contexts>`, `<sources>`, `<receivers>`, etc. have the `order` attribute. The single object elements, such as the `<context>`, `<topic>`, etc., have the `rule` attribute. The default for both attributes allows creation of all objects. You can however, exert some administrative control over your applications by allowing the creation of only certain objects.

You can vary the `order` attribute values to suit whether permission or restriction is more prevalent. In the example below, only a single topic needs to be restricted, so we use the default values for the `order` attribute with only a single topic restricted with a `rule="deny"` attribute.

```

<sources order="deny,allow">
  <topic topicname="CDEF" rule="deny"/>
  <!-- all other source topics allowed -->
</sources>

```

In contrast, the following example requires the creation of only a single receiver topic object, so you can change the `order` attribute to `allow, deny`, which restricts the creation of all receiver topic objects except the one allowed.

```

<receivers order="allow,deny">
  <topic topicname="AARM" rule="allow"/>
  <!-- all other receive topics denied -->
</receivers>

```

You can also combine topic names with topic patterns. In the example below, we set the `order` attribute to the default. Topic `ISM` is denied with its `order` attribute. Topics `IRM` and `SRM` satisfy both their own `allow` rules and the pattern `*R*` deny rule. So when you allocate a source topic with `lbm_src_topic_alloc()`, **UM** accepts the rule that matches the `order` attribute default, which is `allow`.

```

<sources order="deny,allow">
  <topic topicname="ISM" rule="deny"/>
  <topic topicname="IRM" rule="allow"/>
  <topic pattern="*R*" rule="deny"/>
  <topic topicname="SRM" rule="allow"/>
</sources>

```

As a result of the above configuration, **UM** allows the creation of source topic objects **IRM** and **SRM**, and all other topics, except those that match the pattern **\*R\***.

### 1.3.5. XML Configuration File DTD

The XML configuration file DTD is integrated into **UM** and appears below.

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT um-configuration (license | templates | applications)*>
<!ATTLIST um-configuration version CDATA #REQUIRED>

<!ELEMENT license ( #PCDATA )>
<!ATTLIST license format (filename | string) "string">
<!ATTLIST license xml:space (default | preserve) "default">

<!ELEMENT templates (template*)>

<!ELEMENT template (options+)>
<!ATTLIST template name CDATA #REQUIRED>

<!ELEMENT options (option | application-data)*>
<!ATTLIST options type (event-queue | context | source | receiver | wildcard-receiver | hfx) #IMPLIED>

<!ELEMENT option (allow | deny)*>
<!ATTLIST option name CDATA #REQUIRED>
<!ATTLIST option default-value CDATA #IMPLIED>
<!ATTLIST option order CDATA #IMPLIED>

<!ELEMENT application-data ( #PCDATA )>
<!ATTLIST application-data xml:space (default | preserve) "default">

<!ELEMENT allow ( #PCDATA )>
<!ATTLIST allow xml:space (default | preserve) "default">

<!ELEMENT deny ( #PCDATA )>
<!ATTLIST deny xml:space (default | preserve) "default">

<!ELEMENT applications (application*)>

<!ELEMENT application (contexts | event-queues | hfxs | application-data)+>
<!ATTLIST application name CDATA #IMPLIED>
<!ATTLIST application template CDATA #IMPLIED>

<!ELEMENT contexts (context*)>
<!ATTLIST contexts template CDATA #IMPLIED>
<!ATTLIST contexts order CDATA #IMPLIED>

<!ELEMENT event-queues (event-queue*)>
<!ATTLIST event-queues template CDATA #IMPLIED>
<!ATTLIST event-queues order CDATA #IMPLIED>
```

```

<!ELEMENT hfxs (topic*)>
<!ATTLIST hfxs template CDATA #IMPLIED>
<!ATTLIST hfxs order CDATA #IMPLIED>

<!ELEMENT event-queue (options*)>
<!ATTLIST event-queue name CDATA #IMPLIED>
<!ATTLIST event-queue template CDATA #IMPLIED>
<!ATTLIST event-queue rule (allow | deny) "allow">

<!ELEMENT context (sources | receivers | wildcard-receivers | options)+>
<!ATTLIST context name CDATA #IMPLIED>
<!ATTLIST context template CDATA #IMPLIED>
<!ATTLIST context rule (allow | deny) "allow">

<!ELEMENT sources (topic*)>
<!ATTLIST sources template CDATA #IMPLIED>
<!ATTLIST sources order CDATA #IMPLIED>

<!ELEMENT receivers (topic*)>
<!ATTLIST receivers template CDATA #IMPLIED>
<!ATTLIST receivers order CDATA #IMPLIED>

<!ELEMENT wildcard-receivers (wildcard-receiver*)>
<!ATTLIST wildcard-receivers template CDATA #IMPLIED>
<!ATTLIST wildcard-receivers order CDATA #IMPLIED>

<!ELEMENT topic (options*)>
<!ATTLIST topic template CDATA #IMPLIED>
<!ATTLIST topic rule (allow | deny) "allow">
<!ATTLIST topic pattern CDATA #IMPLIED>
<!ATTLIST topic topicname CDATA #IMPLIED>

<!ELEMENT wildcard-receiver (options*)>
<!ATTLIST wildcard-receiver template CDATA #IMPLIED>
<!ATTLIST wildcard-receiver rule (allow | deny) "allow">
<!ATTLIST wildcard-receiver pattern CDATA #IMPLIED>
<!ATTLIST wildcard-receiver pattern-type (pcre | regex | application-callback) #IMPLIED>

```

## 1.4. Configuration File Restrictions

The only options that you cannot set via configuration file are those that require function pointers as their value. Some examples include `context_resolver_source_notification_function` and `wildcard_receiver_pattern_callback`. you can set these options via only API functions. See [Options \(Callbacks\) That Cannot Be Set From a UM Configuration File \(../Config/cannotbesetfromconfigfile.html\)](#) for a list of these options.

## 1.5. Attributes Objects

Many **UM** primitive objects have a corresponding *attributes object*, which lets you create *custom attributes*. From here you can set options specific to an object (but different from default option settings) prior to creating that object. The following table lists the **UM** primitive objects and corresponding attributes objects.

**Table 1-1. UM Objects and Corresponding Attributes Objects**

UM object	Corresponding Attributes Object(s)
lbm_context_t	lbm_context_attr_t
lbm_topic_t	lbm_src_topic_attr_t, lbm_rcv_topic_attr_t
lbm_wildcard_rcv_t	lbm_wildcard_rcv_attr_t
lbm_event_queue_t	lbm_event_queue_attr_t
lbm_hfx_t	lbm_hfx_attr_t

You call API functions to create attributes objects and set, retrieve, or delete their values. These function names are based on the attributes object name and are shown in the following table, using the context object as an example. See the C API ([./API/index.html](#)) for all context attribute functions.

**Table 1-2. UM API Functions For Working With lbm\_context\_attr\_t Attributes Objects**

Action	UM API function
Create	lbm_context_attr_create()
Set Option from Binary Value	lbm_context_attr_setopt()
Set Option from String Value	lbm_context_attr_str_setopt()
Get Option as Binary Value	lbm_context_attr_getopt()
Get Option as String Value	lbm_context_attr_str_getopt()
Delete	lbm_context_attr_delete()
For other object types, replace context with event_queue, hfx, rcv_topic, src_topic, or wildcard_rcv.	

The following sections describe in detail the use of these **UM** API functions. The functions related to `lbm_context_attr_t` objects are used for the purpose of illustration, but the instructions (if not the specifics) apply to all **UM** attributes objects.

### 1.5.1. Creating An Attributes Object

In the following example, the call to `lbm_context_attr_create()` creates the custom attributes object, and initializes each attribute from the current default values. Subsequent calls to `lbm_context_attr_setopt()` or `lbm_context_attr_str_setopt()` modify only the attributes object values.

```
lbm_context_attr_t * attrib;
int rc;
int errnum;
const char * errmsg;
```

```
rc = lbm_context_attr_create(&attrib);
if (rc != 0)
{
    errnum = lbm_errnum();
    errmsg = lbm_errmsg();
    fprintf(stderr, "Error %d returned from lbm_context_attr_create(), %s\n",
        errnum, errmsg);
}
```

This example also illustrates the proper way to determine the success or failure of an **UM** API call. Most **UM** API calls return 0 to indicate success, and -1 to indicate failure. To retrieve the specific **UM** error code for the failure, call `lbm_errnum()`. To retrieve a text string describing the error code, call `lbm_errmsg()`.

## 1.5.2. Setting an Option from a Binary Value

For an option of type other than "string", call `lbm_context_attr_setopt()` to set its value. (See the C API reference for details on this function.) The final two parameters in the function are a pointer to a variable containing the option value, and a pointer to a variable of type `size_t` that contains the correct length of the option value variable.

**UM** options are of three general types that:

- accept values in a well-defined range (Examples include `context transport_tcp_port_low` and `context transport_tcp_port_high`. Each requires a value which corresponds to a valid TCP port number.)
- accept values from an enumerated set, (For example, `context operational_mode`. Manifest constants are provided in `lbm.h` for each permitted value. In the case of `context operational_mode`, those constants are `LBM_CTX_ATTR_OP_EMBEDDED` and `LBM_CTX_ATTR_OP_SEQUENTIAL`.)
- act as switches, enabling or disabling a particular feature (For example, `context resolver_cache`. The set of allowed values is limited to 0 (indicating off, no, false, or disabled), and 1, indicating on, yes, true, or enabled.)

The example code below sets four options. First, we set the operational mode to sequential. Then we set the transport TCP port low and high values to 4901 and 4920, respectively. Finally, we tell **UM** that our application will not be using multiple sending threads per transport session.

```
lbm_context_attr_t * attrib; /* Must have already been created */
int rc;
unsigned short int optval;
size_t optlen;

/* Set the operational_mode */
optlen = sizeof(optval);
optval = LBM_CTX_ATTR_OP_SEQUENTIAL;
rc = lbm_context_attr_setopt(attrib, "operational_mode", &optval, optlen);
if (rc != 0)
{
    /* Handle error */
}

/* Set transport_tcp_port_low */
optlen = sizeof(optval);
```

```

optval = 4901;
rc = lbm_context_attr_setopt (attrib, "transport_tcp_port_low", &optval, optlen);
if (rc != 0)
{
    /* Handle error */
}

/* Set transport_tcp_port_high */
optlen = sizeof(optval);
optval = 4920;
rc = lbm_context_attr_setopt (attrib, "transport_tcp_port_high", &optval, optlen);
if (rc != 0)
{
    /* Handle error */
}

/* Set transport_session_multiple_sending_threads */
optlen = sizeof(optval);
optval = 0;
rc = lbm_context_attr_setopt (attrib, "transport_session_multiple_sending_threads",
                             &optval, optlen);

if (rc != 0)
{
    /* Handle error */
}

```

### 1.5.3. Setting an Option from a String Value

Setting an option from a string value effectively does the same thing that setting an option from a binary value does. However, the option value is passed as a null-terminated string, rather than as value and length pointers. **UM** uses this mechanism to process options in a configuration file. Thus, the format used for option values must match the format you would use in a configuration file.

In the following example, as before, we set the operational mode to sequential. Then we set the transport TCP port low and high values to 4901 and 4920, respectively. Finally, we tell **UM** that our application will not be using multiple sending threads per transport session.

```

lbm_context_attr_t * attrib; /* Must have already been created */
int rc;

/* Set the operational_mode */
rc = lbm_context_attr_str_setopt (attrib, "operational_mode", "sequential");
if (rc != 0)
{
    /* Handle error */
}

/* Set transport_tcp_port_low */
rc = lbm_context_attr_str_setopt (attrib, "transport_tcp_port_low", "4901");
if (rc != 0)
{

```

```

    /* Handle error */
}

/* Set transport_tcp_port_high */
rc = lbm_context_attr_str_setopt(attrib, "transport_tcp_port_high", "4920");
if (rc != 0)
{
    /* Handle error */
}

/* Set transport_session_multiple_sending_threads */
rc = lbm_context_attr_str_setopt(attrib, "transport_session_multiple_sending_threads",
                                "0");
if (rc != 0)
{
    /* Handle error */
}

```

### 1.5.4. Getting an Option as a Binary Value

Getting an option as a binary value is very similar to setting an option from a binary value: it requires knowledge of not only the option name, but its type as well. The final two parameters in the call to `lbm_context_attr_getopt()` are a pointer to a variable to receive the current option value, and a pointer to a variable of type `size_t` which contains the length of the option value variable. This length must be correct for the specified option.

In the example code below, we set the option values for operational mode, the transport TCP port low and high values, and retrieve multiple sending threads.

```

lbm_context_attr_t * attrib; /* Must have already been created */
int rc;
unsigned short int optval;
size_t optlen;

/* Get the operational_mode */
optlen = sizeof(optval);
rc = lbm_context_attr_getopt(attrib, "operational_mode", &optval, &optlen);
if (rc != 0)
{
    /* Handle error */
}
/* optval now contains LBM_CTX_ATTR_OP_EMBEDDED or LBM_CTX_ATTR_OP_SEQUENTIAL */

/* Get transport_tcp_port_low */
optlen = sizeof(optval);
rc = lbm_context_attr_getopt(attrib, "transport_tcp_port_low", &optval, &optlen);
if (rc != 0)
{
    /* Handle error */
}
/* optval now contains the value of transport_tcp_port_low, which should be 4901 */

```

```

/* Get transport_tcp_port_high */
optlen = sizeof(optval);
rc = lbm_context_attr_getopt (attrib, "transport_tcp_port_high", &optval, &optlen);
if (rc != 0)
{
    /* Handle error */
}
/* optval now contains the value of transport_tcp_port_high, which should be 4920 */

/* Get transport_session_multiple_sending_threads */
optlen = sizeof(optval);
rc = lbm_context_attr_getopt (attrib, "transport_session_multiple_sending_threads",
                             &optval, &optlen);

if (rc != 0)
{
    /* Handle error */
}
/* optval now contains the value of transport_session_multiple_sending_threads,
   which should be 0. */

```

### 1.5.5. Getting an Option as a String Value

Getting an option as a string value effectively does the same thing that getting an option as a binary value does. However, the option value is returned as a null-terminated string, just as you would specify the option value in a configuration file. The final two parameters in the call to `lbm_context_attr_str_getopt()` are a pointer to a string variable to receive the current option value, and a pointer to a variable of type `size_t` which contains the maximum size of the option value string variable.

In the example code below, the option values for operational mode, the transport TCP port low and high values, and multiple sending threads are retrieved.

```

lbm_context_attr_t * attrib; /* Must have already been created */
int rc;
char optval_string[256];

/* Get the operational_mode */
optlen = sizeof(optval_string);
rc = lbm_context_attr_str_getopt (attrib, "operational_mode", optval_string, &optlen);
if (rc != 0)
{
    /* Handle error */
}
/* optval_string now contains either "embedded" or "sequential" */

/* Get transport_tcp_port_low */
optlen = sizeof(optval_string);
rc = lbm_context_attr_str_getopt (attrib, "transport_tcp_port_low",
                                 optval_string, &optlen);

if (rc != 0)
{

```



```

    /* Handle error */
}
/* optval_string now contains the string value of transport_tcp_port_low,
   which should be "4901" */

/* Get transport_tcp_port_high */
optlen = sizeof(optval_string);
rc = lbm_context_attr_str_getopt(attrib, "transport_tcp_port_high",
                                optval_string, &optlen);

if (rc != 0)
{
    /* Handle error */
}
/* optval_string now contains the string value of transport_tcp_port_high,
   which should be "4920" */

/* Get transport_session_multiple_sending_threads */
optlen = sizeof(optval_string);
rc = lbm_context_attr_str_getopt(attrib, "transport_session_multiple_sending_threads",
                                optval_string, &optlen);

if (rc != 0)
{
    /* Handle error */
}
/* optval_string now contains the value of transport_session_multiple_sending_threads,
   which should be "0". */

```

### 1.5.6. Deleting an Attributes Object

Once the attributes object is no longer needed, it should be deleted.

```

lbm_context_attr_t * attrib; /* Must have already been created */
int rc;

rc = lbm_context_attr_delete(attrib);
if (rc != 0)
{
    /* Handle error */
}

```

### 1.5.7. Restrictions

There are no restrictions on setting options via attributes objects. Any option which can be set via a configuration file, can also be set via an attributes object. In addition, attributes objects allow setting certain options (such as function pointers) which cannot be set with a configuration file.

## 1.6. Modifying Current Attributes

A few options within an **UM** object's current attributes can be set after the object is created. **UM** API functions supporting such actions operate on the object itself, rather than on an attributes object. In addition to modifying the current attributes, the value of options from the current attributes can be fetched.

The **UM** objects which support these actions are `lbm_src_t`, `lbm_rcv_t`, `lbm_context_t`, and `lbm_event_queue_t`. For each such object, there are corresponding API functions to set an option from a binary value, set an option from a string value, get an option as a binary value, and get an option as a string value. These function names are based on the object name, suffixed with `_setopt()`, `_str_setopt()`, `_getopt()`, and `_str_getopt()`. As an illustration of this convention, the API functions for working with `lbm_event_queue_t` objects are shown in the following table.

**Table 1-3. UM API Functions For Working With `lbm_event_queue_t` Objects**

Action	UM API function
Set Option from a Binary Value	<code>lbm_event_queue_setopt()</code>
Set Option from a String Value	<code>lbm_event_queue_str_setopt()</code>

The following sections describe in detail the use of these **UM** API functions. The functions related to `lbm_event_queue_t` objects are used for the purpose of illustration, but the instructions (if not the specifics) apply to all such **UM** objects.

### 1.6.1. Setting An Option from a Binary Value

Setting an option from a binary value requires knowledge of not only the option name, but its type and allowable values as well. The final two parameters in the call to `lbm_event_queue_setopt()` are a pointer to a variable which contains the option value to be set, and a pointer to a variable of type `size_t` which contains the length of the option value variable. This length must be correct for the specified option.

In the example code below, we set the queue size warning to 5000 events.

```
unsigned long int optval;
size_t optlen;
lbm_event_queue_t evq; /* must be previously created */
int rc;

/* Set the queue size warning */
optlen = sizeof(optval);
optval = 5000;
rc = lbm_event_queue_setopt(&evq, "queue_size_warning", &optval, &optlen);
if (rc != 0)
{
    /* Handle error */
}
```

## 1.6.2. Setting An Option from a String Value

Setting an option from a string value effectively does the same thing that setting an option from a binary value does. However, the option value is passed as a null-terminated string, rather than as value and length pointers. This is similar to the mechanism used by **UM** to process options in a configuration file. Thus, the format used for option values must match the format you would use in a configuration file.

As before, we set the queue size warning to 5000 events.

```
lbm_event_queue_t evq; /* must be previously created */
int rc;

/* Set the queue size warning */
rc = lbm_event_queue_setopt(&evq, "queue_size_warning", "5000");
if (rc != 0)
{
    /* Handle error */
}
```

## 1.6.3. Restrictions

Modifying the current attributes of a object allows only a very limited subset of options to be set or retrieved. Consult subsequent sections of this document to determine if a particular option can be specified.

## 1.7. Retrieving Current Option Values

Most **UM** objects allow their current attributes' option values to be retrieved during operation. **UM** API functions supporting such actions operate on the object itself.

The **UM** objects which support these actions are `lbm_src_t`, `lbm_rcv_t`, `lbm_context_t`, and `lbm_event_queue_t`. For each such object, there are corresponding API functions to get an option as a binary value, and get an option as a string value. These function names are based on the object name, suffixed with `_getopt()`, and `_str_getopt()`. As an illustration of this convention, the API functions for working with `lbm_event_queue_t` objects are shown in the following table.

**Table 1-4. UM API Functions For Retrieving Option Values from `lbm_event_queue_t` Objects**

Action	UM API function
Get Option as a Binary Value	<code>lbm_event_queue_getopt()</code>
Get Option as a String Value	<code>lbm_event_queue_str_getopt()</code>

The following sections describe in detail the use of these **UM** API functions. The functions related to `lbm_event_queue_t` objects are used for the purpose of illustration, but the instructions (if not the specifics) apply to all such **UM** objects.

### 1.7.1. Getting An Option as a Binary Value

Getting an option as a binary value is very similar to setting an option from a binary value: it requires knowledge of not only the option name, but its type as well. The final two parameters in the call to `lbm_event_queue_getopt()` are a pointer to a variable to receive the current option value, and a pointer to a variable of type `size_t` which contains the length of the option value variable. This length must be correct for the specified option.

In the example code below, the option value for the queue size warning is retrieved.

```
unsigned long int optval;
size_t optlen;
lbm_event_queue_t evq; /* must be previously created */
int rc;

/* Get the queue size warning value */
optlen = sizeof(optval);
rc = lbm_event_queue_getopt(&evq, "queue_size_warning", &optval, &optlen);
if (rc != 0)
{
    /* Handle error */
}
/* optval now contains the value of queue_size_warning, which should be 5000 */
```

### 1.7.2. Getting An Option as a String Value

Getting an option as a string value effectively does the same thing that getting an option as a binary value does. However, the option value is returned as a null-terminated string, just as you would specify the option value in a configuration file. The final two parameters in the call to `lbm_event_queue_str_getopt()` are a pointer to a string variable to receive the current option value, and a pointer to a variable of type `size_t` which contains the maximum size of the option value string variable.

In the example code below, the option value for the queue size warning is retrieved.

```
char optval_string[256];
size_t optlen;
lbm_event_queue_t evq; /* must be previously created */
int rc;

/* Get the queue size warning value */
optlen = sizeof(optval_string);
rc = lbm_event_queue_str_getopt(&evq, "queue_size_warning", optval_string, &optlen);
if (rc != 0)
{
    /* Handle error */
}
/* optval now contains the value of queue_size_warning, which should be "5000" */
```

# Chapter 2. Example Configuration Scenarios

This chapter contains some example configuration scenarios.

## 2.1. Highest Throughput

The following configuration option tunes **UMS** for the highest possible throughput.

```
#
# LBM can be configured to make efficient use of CPU time, leading
# to the highest-possible throughput (bytes per second or messages
# per second). This may come at the expense of latency at low
# message rates. The following line configures LBM to accumulate
# 8KB of messages (or for wait implicit_batching_interval) before sending.
#
source implicit_batching_minimum_length 8192
```

You may download the file ([Examples/HighestThroughput.cfg](#)). Most browsers let you right-click on the link and use the *save link target* function, or some variation.

## 2.2. Lowest Latency

This is an example configuration that favors low latency at the expense of higher CPU utilization and potentially lower throughput.

```
#
# Latency can be reduced at the expense of network efficiency and
# system CPU time by adjusting implicit batching parameters. The
# default parameters hold messages for up to 200 milliseconds or until
# 2048 bytes are waiting to go. The lowest possible latency is
# obtained by setting the minimum batching length to 1 byte, which
# effectively disables the implicit batching feature. For example:
#
context mim_implicit_batching_minimum_length 1
source      implicit_batching_minimum_length 1
#
# Latency can be kept to a minimum with UM by writing receiving
# applications that can accept messages in the order they arrive.
# See https://communities.informatica.com/infakb/faq/5/Pages/80043.aspx and
# http://www.29West.Com/docs/THPM/tcp-latency.html#TCP-RECEIVER-SIDE-LATENCY
# for more information. Here's how to use arrival-order delivery:
#
receiver ordered_delivery 0
#
# Disable Nagel's algorithm (batching) for TCP responses to eliminate
# queuing latency when sending only single responses.
#
context response_tcp_nodelay 1
```

```
#
# If you are running a LAN environment with under 100 machines, you can
# drastically improve your recovery related latencies without significant
# additional network overhead by using the following UM loss
# recovery parameter. See https://communities.informatica.com/infakb/faq/5/Pages/80070.aspx
# for additional information about this and other recovery parameters.
#
receiver transport_lbtrm_nak_backoff_interval 10
#
# Use of a zero value for the following parameter sends an immediate NAK upon
# loss detection, which can further reduce repair latency. (Immediate NAKs do
# not elicit an NCF by the source.) It is critical you understand the implications
# of this feature and we recommend that you contact http://29west.com/support to
# learn more before enabling it.
#
# receiver transport_lbtrm_nak_initial_backoff_interval 0
#
```

You may download the file (Examples/LowestLatency.cfg). Most browsers let you right-click on the link and use the *save link target* function, or some variation.

## 2.3. Creating Multicast Sources

This is an example configuration file that changes the default transport to reliable multicast so all sources created send messages over LBT-RM.

```
#
# UM can be configured to create sources using the LBT-RM reliable
# multicast protocol instead of the default TCP.
#
source transport LBT-RM
#
# Stable and reliable operation with multicast requires careful
# setting of rate control limits. See
# http://www.29west.com/docs/THPM/thpm.html#GROUP-RATE-CONTROL
# for background information.
#
# It's generally best to start with small limits and gradually
# increase them after testing indicates that they can be safely
# sustained on your network.
#
# The following example limits (new) data to 10 Mbps and retransmissions
# to 1 Mbps (10%). Note that when changing the data rate limit, the
# limit retransmission limit should be changed as well. A good value
# for most purposes is between 2% and 10% of the data rate limit, with
# a lower limit of 1,000,000.
#
#
context transport_lbtrm_data_rate_limit 10000000
context transport_lbtrm_retransmit_rate_limit 1000000
```

You may download the file (Examples/MulticastSource.cfg). Most browsers let you right-click on the link and use the *save link target* function, or some variation.

## 2.4. Disabling Aspects of Topic Resolution

If you need to reduce the amount of Topic Resolution traffic on your network, use the following Configuration options and values in a **Ultra Messaging** Configuration file.

**Note:** **Ultra Messaging** does not recommend disabling both advertisements and queries because topics may not resolve at all.

### 2.4.1. Disabling Topic Advertisements

You can disable topic advertisements in the Initial Phase, Sustaining Phase or both phases of topic resolution.

#### 2.4.1.1. Disabling Initial Phase Advertisements

Use one or both of the following options to disable topic advertisements in only the Initial Phase.

```
source resolver_advertisement_minimum_initial_interval 0
source resolver_advertisement_maximum_initial_interval 0
```

#### 2.4.1.2. Disabling Sustaining Phase Advertisements

Use the following option to disable topic advertisements in only the Sustaining Phase.

```
source resolver_advertisement_sustain_interval 0
```

### 2.4.2. Disabling Receiver Topic Queries

You can disable the querying of topics by receivers in the Initial Phase, Sustaining Phase or both phases of topic resolution.

#### 2.4.2.1. Disabling Initial Phase Queries

Use one or both of the following options to disable topic queries in only the Initial Phase.

```
receiver resolver_query_minimum_initial_interval 0
receiver resolver_query_maximum_initial_interval 0
```

### 2.4.2.2. Disabling Sustaining Phase Queries

Use one or both of the following options to disable topic queries in only the Sustaining Phase.

```
receiver_resolver_query_sustain_interval 0
receiver_resolution_number_of_sources_query_threshold 0
```

### 2.4.3. Disabling Wildcard Topic Queries

Use one or both of the following options to disable topic queries by wildcard receivers.

```
wildcard_receiver_resolver_query_minimum_interval 0
wildcard_receiver_resolver_query_maximum_interval 0
```

### 2.4.4. Disabling All But the Minimum Topic Resolution Traffic

A minimalist approach to topic resolution can take different forms based on your requirements. One approach is to disable all traffic except for queries in the sustaining phase. Add the following settings to your **Ultra Messaging** configuration file to implement this approach.

```
source_resolver_advertisement_minimum_initial_interval 0
source_resolver_advertisement_sustain_interval 0
receiver_resolver_query_minimum_initial_interval 0
receiver_resolution_number_of_sources_query_threshold 1
wildcard_receiver_resolver_query_minimum_interval 0
```

### 2.4.5. Re-establish Pre-4.0 Topic Resolution

**Ultra Messaging** topic resolution prior to **LBM** Version 4.0 did not have resolution phases. To implement pre-4.0 topic resolution, include the following configuration option changes in your **Ultra Messaging** configuration file.

```
# ----- Disable Advertisements in 4.0 Initial Phase
source_resolver_advertisement_minimum_initial_interval 0

# ----- Re-establish pre-4.0 Advertisement Behavior
source_resolver_advertisement_minimum_sustain_duration 0
context_resolver_sustain_advertisement_bps 0

# ----- Disable Queries in 4.0 Initial Phase
receiver_resolver_query_minimum_initial_interval 0

# ----- Re-establish pre-4.0 Query Behavior
receiver_resolver_query_sustain_interval 100
receiver_resolver_query_minimum_sustain_duration 0
context_resolver_sustain_query_bps 0
receiver_resolution_number_of_sources_query_threshold 1
```



```
# ----- Re-establish pre-4.0 Wildcard Query Behavior
wildcard_receiver resolver_query_minimum_interval 0
```

## 2.5. Unicast Resolver

To use the unicast resolver, the following configuration file may be used.

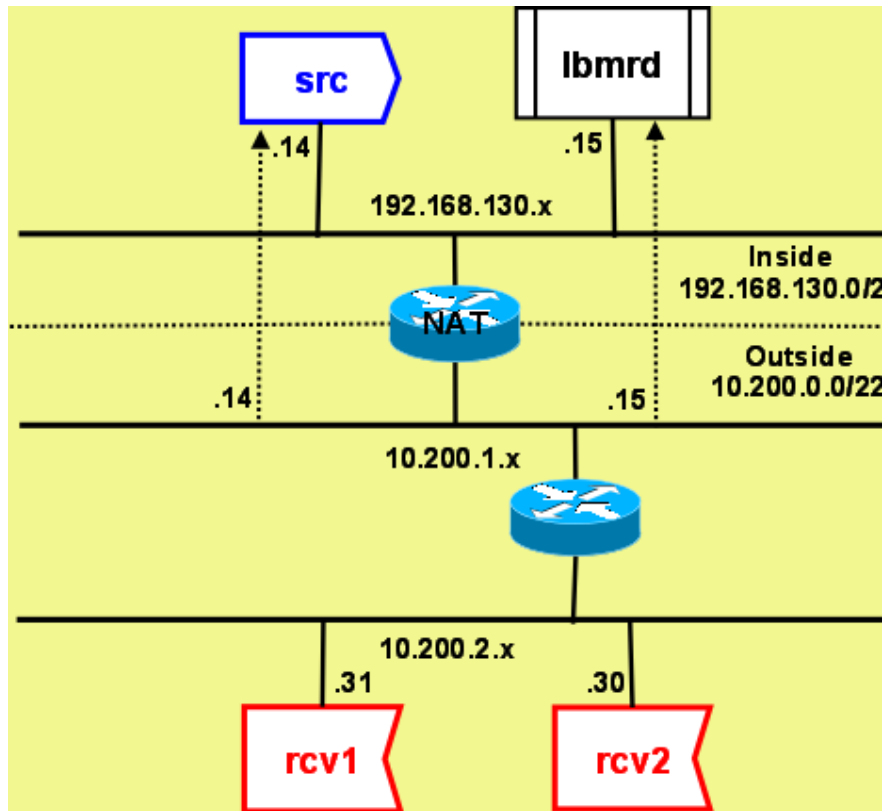
```
#
# Topic resolution can be configured to use unicast traffic with an
# LBM resolver daemon (lbmr) instead of the default which uses multicast.
# Be sure to insert the IP address of your lbmr below.
#
context resolver_unicast_address 127.0.0.1
```

You may download the file ([Examples/UnicastResolver.cfg](#)). Most browsers let you right-click on the link and use the *save link target* function, or some variation.

### 2.5.1. Unicast Resolution Across Administrative Domains

The following network architecture shows a source or publisher (**src**) on a *private* LAN.

Figure 2-1. Unicast Topic Resolution Across Domains



The following **lbmrdr** configuration file allows **rcv1** and **rcv2** to connect to **src** for receipt of topic messages.

```
<?xml version="1.0" encoding="UTF-8" ?>
<lbmrdr version="1.0">
  <domains>
    <domain name="NAT">
      <network>192.168.130.0/24</network>
    </domain>
    <domain name="OPEN">
      <network>10.200.2.31</network>
      <network>10.200.2.30</network>
    </domain>
  </domains>
  <transformations>
    <transform source="NAT" destination="OPEN">
      <rule>
        <match address="192.168.130.14" port="*" />
        <replace address="10.200.1.14" port="*" />
      </rule>
    </transform>
  </transformations>
</lbmrdr>
```

## 2.6. Configure Previous Port Defaults

To use the previous default ports (prior to **LBM 3.3** and **UME 2.0**), the following configuration file may be used.

```
context mim_destination_port          4401
context mim_incoming_destination_port 4401
context mim_outgoing_destination_port 4401
context resolver_multicast_incoming_port 2965
context resolver_multicast_outgoing_port 2965
context resolver_multicast_port       2965
context resolver_unicast_destination_port 5380
context resolver_unicast_port_high    4406
context resolver_unicast_port_low     4402
source transport_lbtrm_destination_port 4400
context transport_lbtrm_source_port_high 4399
context transport_lbtrm_source_port_low  4390
context transport_lbtru_port_high       4389
receiver transport_lbtru_port_high      4379
context transport_lbtru_port_low        4380
receiver transport_lbtru_port_low       4360
context request_tcp_port_high          4395
context request_tcp_port_low           4391
context transport_tcp_port_high        4390
context transport_tcp_port_low         4371
source ume_primary_store_port         4567
source ume_secondary_store_port       4567
source ume_tertiary_store_port         4567
```

You may download the file ([Examples/Configure.Previous.Ports.cfg](#)). Most browsers let you right-click on the link and use the *save link target* function, or some variation.

**Note:** Alternatively, **UMS/UMP** will use the original port settings with the definition of the "LBM\_USE\_ORIG\_DEFAULT\_PORTS" environment variable (value not pertinent).

## 2.7. Configure New Port Defaults

In the unusual case that you must run older versions of **Ultra Messaging** (less than **LBM 3.3** / **UME 2.0**) on certain machine(s) and need these older version to work with the machines running the current versions of **UMS** and **UMP**, you can use the following configuration file for the older versions to synchronize port usage between old and current versions.

```
context mim_destination_port          14401
context mim_incoming_destination_port 14401
context mim_outgoing_destination_port 14401
context resolver_multicast_incoming_port 12965
context resolver_multicast_outgoing_port 12965
context resolver_multicast_port       12965
context resolver_unicast_destination_port 15380
```

```

context resolver_unicast_port_high      14406
context resolver_unicast_port_low       14402
source transport_lbtrm_destination_port 14400
context transport_lbtrm_source_port_high 14399
context transport_lbtrm_source_port_low  14390
context transport_lbtru_port_high        14389
receiver transport_lbtru_port_high       14379
context transport_lbtru_port_low         14380
receiver transport_lbtru_port_low        14360
context request_tcp_port_high            14395
context request_tcp_port_low             14391
context transport_tcp_port_high          14390
context transport_tcp_port_low           14371
source ume_primary_store_port            14567
source ume_secondary_store_port          14567
source ume_tertiary_store_port           14567

```

You may download the file ([Examples/Configure.Previous.Ports.cfg](#)). Most browsers let you right-click on the link and use the *save link target* function, or some variation.

## 2.8. Interrelated Configuration Options

Some **Ultra Messaging** configuration options are related in ways that might not be immediately apparent. Changing the value for one option without adjusting its related option can cause problems such as NAK storms, tail loss, etc. This section identifies these relationships and recommends a best practice for setting the interrelated options.

The following sections discuss configuration option relationships.

- *Preventing NAK Storms with NAK Interval Options*
- *Preventing Tail Loss With TSNI and NAK Interval Options*
- *Preventing IPC Receiver Deafness With Keepalive Options*
- *Preventing Erroneous LBT-RM/LBT-RU Session Timeouts*
- *Preventing Errors Due to Bad Multicast Address Ranges*
- *Preventing Store or Queue Timeouts*
- *Preventing ULB Timeouts*
- *Preventing Unicast Resolver Daemon Timeouts*
- *Preventing Undetected Late Join Loss*
- *Preventing Undetected Loss*

### 2.8.1. Preventing NAK Storms with NAK Interval Options

The NAK generation interval should be sufficiently longer than the NAK backoff interval so that the source, after receiving the first NAK from a receiver, has time to retransmit the missing datagram and prevent a NAK storm from

all receivers. LBTRM, LBTRU, and MIM all use NAK generation and backoff intervals. The NAK behavior for all transports is the same.

Interrelated Options:

- `transport_lbtrm_nak_backoff_interval`  
(`../Config/transportlbt-rmreliabilityoptions.html#RECEIVERTRANSPORTLBTRMNAKBACKOFFINTERVAL`)
- `transport_lbtrm_nak_generation_interval` (`../Config/transportlbt-rmreliabilityoptions.html#RECEIVERTRANSPORTLBTRMNAKGENERATIONINTERVAL`)
- `transport_lbtru_nak_backoff_interval`  
(`../Config/transportlbt-rureliabilityoptions.html#RECEIVERTRANSPORTLBTRUNAKBACKOFFINTERVAL`)
- `transport_lbtru_nak_generation_interval` (`../Config/transportlbt-rureliabilityoptions.html#RECEIVERTRANSPORTLBTRUNAKGENERATIONINTERVAL`)
- `mim_nak_backoff_interval`  
(`../Config/multicastimmediatemessagingreliabilityoptions.html#CONTEXTMIMNAKBACKOFFINTERVAL`)
- `mim_nak_generation_interval` (`../Config/multicastimmediatemessagingreliabilityoptions.html#CONTEXTMIMNAKGENERATIONINTERVAL`)

Recommendation:

- Set the NAK generation interval to at least 2x the NAK backoff interval.

For more, see also Transport LBT-RM Reliability Options (`../Config/transportlbt-rmreliabilityoptions.html`), Transport LBT-RU Reliability Options (`../Config/transportlbt-rureliabilityoptions.html`), or Multicast Immediate Messaging Reliability Options (`../Config/multicastimmediatemessagingreliabilityoptions.html`).

Example:

```
#
# +-----+
# | To avoid NAK storms, set NAK generation interval to at least 2x the      |
# | NAK backoff interval.                                                    |
# +-----+
#
receiver transport_lbtrm_nak_backoff_interval      200
receiver transport_lbtrm_nak_generation_interval  10000
```

## 2.8.2. Preventing Tail Loss With TSNI and NAK Interval Options

Tail loss describes a situation where the last few messages sent by a publisher before it exits are not received by a subscriber. A TSNI active threshold that is too small relative to the TSNI and/or NAK generation interval may cause tail loss, especially with ordered delivery.

Interrelated Options:

- `transport_topic_sequence_number_info_active_threshold` (`../Config/majoroptions.html#SOURCETRANSPORTTOPICSEQUENCENUMBERINFOACTIVETHRESHOLD`)

- `transport_topic_sequence_number_info_interval`  
(`../Config/majoroptions.html#SOURCETRANSPORTTOPICSEQUENCENUMBERINFOINTERVAL`)
- `transport_lbtrm_nak_generation_interval` (`../Config/transportlbt-rmreliabilityoptions.html#RECEIVERTRANSPORTLBTRMNAKGENERATIONINTERVAL`)
- `transport_lbtru_nak_generation_interval` (`../Config/transportlbt-rureliabilityoptions.html#RECEIVERTRANSPORTLBTRUNAKGENERATIONINTERVAL`)

Recommendation:

- set the TSNI active threshold to at least 4x the topic sequence number info interval (TSNI) plus the NAK generation interval.

For more, see Transport LBT-RM Reliability Options

(`../Config/transportlbt-rmreliabilityoptions.html#TRANSPORTLBT-RMRELIABILITYOPTIONS`) or Transport

LBT-RU Reliability Options

(`../Config/transportlbt-rureliabilityoptions.html#TRANSPORTLBT-RURELIABILITYOPTIONS`).

Example:

```
#
# +-----+
# | To avoid tail loss, set transport_topic_sequence_number_info_active_threshold |
# | to at least the sum of 4x the topic sequence number interval plus the NAK      |
# | generation interval.                                                         |
# | NOTE: resolver_active_threshold is in seconds.                             |
# +-----+
#
source    transport_topic_sequence_number_info_interval      2000
receiver  transport_lbtrm_nak_generation_interval            10000
source    transport_topic_sequence_number_info_active_threshold  60
```

### 2.8.3. Preventing IPC Receiver Deafness With Keepalive Options

With an LBT-IPC transport, an activity timeout that is too small relative to the session message interval may cause receiver deafness. If a timeout is too short, the keepalive messages might not be received in time to prevent the receiver from being deleted or disconnecting because the source appears to be gone.

Interrelated Options:

- `transport_lbtipc_activity_timeout`  
(`../Config/transportlbt-ipcoperationoptions.html#RECEIVERTRANSPORTLBTIPCACTIVITYTIMEOUT`)
- `transport_lbtipc_sm_interval`  
(`../Config/transportlbt-ipcoperationoptions.html#SOURCETRANSPORTLBTIPCSMINTERVAL`)

Recommendations:

- set the activity timeout to at least 2x the session message interval

For more, see Transport LBT-IPC Operation Options (`../Config/transportlbt-ipcoperationoptions.html`).

Example:

```
#
# +-----+
# | To avoid receiver deafness: |
# | - set client activity timeout to at least 2x the acknowledgement interval. |
# | - set activity timeout to at least 2x the session message interval. |
# +-----+
#
receiver transport_lbtipc_activity_timeout      60000
source    transport_lbtipc_sm_interval          10000
```

## 2.8.4. Preventing Erroneous LBT-RM/LBT-RU Session Timeouts

An LBT-RM or LBT-RU receiver-side quiescent timeout may delete a transport session that a source is still active on. This can happen if the timeout is too short relative to the source's interval between session messages (which serve as a session keepalive).

Interrelated Options:

- `transport_lbtrm_activity_timeout`  
(`../Config/transportlbt-rmoperationoptions.html#RECEIVERTRANSPORTLBTRMACTIVITYTIMEOUT`)
- `transport_lbtrm_sm_maximum_interval`  
(`../Config/transportlbt-rmoperationoptions.html#SOURCETRANSPORTLBTRMSMMAXIMUMINTERVAL`)
- `transport_lbtru_activity_timeout`  
(`../Config/transportlbt-ruoperationoptions.html#RECEIVERTRANSPORTLBTRUACTIVITYTIMEOUT`)
- `transport_lbtru_sm_maximum_interval`  
(`../Config/transportlbt-ruoperationoptions.html#SOURCETRANSPORTLBTRUSMMAXIMUMINTERVAL`)

Recommendations:

- set the receiver LBT-RM or LBT-RU activity timeout to at least 3x the source session message maximum interval

For more, see Transport LBT-RM Operation Options (`../Config/transportlbt-rmoperationoptions.html`) or Transport LBT-RU Operation Options (`../Config/transportlbt-ruoperationoptions.html`).

Example:

```
#
# +-----+
# | To avoid erroneous session timeouts, set receiver transport activity |
# | timeout to at least 3x the source session message maximum interval. |
# +-----+
#
receiver transport_lbtrm_activity_timeout      60000
source    transport_lbtrm_sm_maximum_interval  10000
receiver transport_lbtru_activity_timeout      60000
source    transport_lbtru_sm_maximum_interval  10000
```

## 2.8.5. Preventing Errors Due to Bad Multicast Address Ranges

Sometimes it is easy to accidentally reverse the low and high values for LBT-RM multicast addresses, which actually creates a very large range. Aside from excluding intended addresses, this can cause error conditions.

Interrelated Options:

- `transport_lbtrm_multicast_address_low`  
(`../Config/transportlbt-rmnetworkoptions.html#CONTEXTTRANSPORTLBTRMMULTICASTADDRESSLOW`)
- `transport_lbtrm_multicast_address_high`  
(`../Config/transportlbt-rmnetworkoptions.html#CONTEXTTRANSPORTLBTRMMULTICASTADDRESSHIGH`)

Recommendations:

- ensure that the intended low and high values for LBT-RM multicast addresses are not reversed

For more, see Transport LBT-RM Network Options (`../Config/transportlbt-rmnetworkoptions.html`).

Example:

```
#
# +-----+
# | To avoid incorrect LBT-RM multicast address ranges, ensure that you have not |
# | reversed the low and high values. |
# +-----+
#
context transport_lbtrm_multicast_address_low 224.10.10.10
context transport_lbtrm_multicast_address_high 224.10.10.14
```

## 2.8.6. Preventing Store or Queue Timeouts

**Note:** These interrelations apply only to the Ultra Messaging Persistence or Ultra Messaging Queuing Edition.

A store or queue may be erroneously declared unresponsive if its activity timeout expires before it has had adequate opportunity to verify it is still active via activity check intervals.

Interrelated Options:

- `ume_store_activity_timeout`
- `ume_store_check_interval`
- `umq_queue_activity_timeout`
- `umq_queue_check_interval`



**Recommendations:**

- set the store or queue activity timeout to at least 5x the activity check interval

For more, see the **UM** Configuration Guide, 4.29. Ultra Messaging Persistence Options and/or (if using **UM** Queuing Edition), the **UM** Configuration Guide, 4.30. Ultra Messaging Queuing Options.

**Example:**

```
#
# +-----+
# | To avoid erroneous store or queue activity timeouts, set the activity      |
# | timeout to at least 5x the activity check interval.                      |
# +-----+
#
source ume_store_activity_timeout          3000
source ume_store_check_interval            500
context umq_queue_activity_timeout        3000
context umq_queue_check_interval          500
```

## 2.8.7. Preventing ULB Timeouts

**Note:** These interrelations apply only to the Ultra Messaging Queuing Edition.

A ULB source or receiver may be erroneously declared unresponsive if its activity timeout expires before it has had adequate opportunities to attempt to re-register via activity check intervals if the source appears to be inactive. It is also possible for sources to attempt to reassign messages that have already been processed.

**Interrelated Options:**

- `umq_ulb_source_activity_timeout`
- `umq_ulb_source_check_interval`
- `umq_ulb_application_set_message_reassignment_timeout`
- `umq_ulb_application_set_receiver_activity_timeout`
- `umq_ulb_check_interval`

**Recommendations:**

- set the ULB source activity timeout to at least 5x the ULB source activity check interval
- set the ULB application set message reassignment timeout to at least 5x the ULB check interval
- set the ULB receiver activity timeout to at least 5x the ULB check interval

For more (if using **UM** Queuing Edition), see the **UM** Configuration Guide, 4.30. Ultra Messaging Queuing Options.

**Example:**

```
#
# +-----+
# | To avoid erroneous ULB source, receiver or application set message activity |
# | timeouts, set the activity timeout to at least 5x the activity check      |
# | interval.                                                                |
# +-----+
#
receiver umq_ulb_source_activity_timeout      10000
receiver umq_ulb_source_check_interval        1000
source   umq_ulb_application_set_message_reassignment_timeout 50000
source   umq_ulb_application_set_receiver_activity_timeout   10000
source   umq_ulb_check_interval                 1000
```

### 2.8.8. Preventing Unicast Resolver Daemon Timeouts

A unicast resolver daemon may be erroneously declared inactive if its activity timeout expires before it has had adequate opportunity to verify that it is still alive.

Interrelated Options:

- `resolver_unicast_activity_timeout`  
(`../Config/resolveroperationoptions.html#CONTEXTRESOLVERUNICASTACTIVITYTIMEOUT`)
- `resolver_unicast_check_interval`  
(`../Config/resolveroperationoptions.html#CONTEXTRESOLVERUNICASTCHECKINTERVAL`)

Recommendations:

- Set the unicast resolver daemon activity timeout to at least 5x the activity check interval. Or, if activity notification is not desired, set both options to 0.

For more, see `Resolver Operation Options (../Config/resolveroperationoptions.html)`.

Example:

```
#
# +-----+
# | To avoid erroneous unicast resolver daemon timeouts, set the activity      |
# | timeout to at least 5x the activity check interval.                      |
# +-----+
#
context  resolver_unicast_activity_timeout      1000
context  resolver_unicast_check_interval        200
```

## 2.8.9. Preventing Undetected Late Join Loss

If during a Late Join operation, a transport times out while a receiver is requesting retransmission of missing messages, this can cause lost messages to go undetected and likely become unrecoverable.

Interrelated Options:

- `retransmit_request_generation_interval`  
(`../Config/latejoinoptions.html#RECEIVERRETRANSMITREQUESTGENERATIONINTERVAL`)
- `transport_tcp_activity_timeout`  
(`../Config/transporttcpoperationoptions.html#RECEIVERTRANSPORTTCPACTIVITYMETHOD`)
- `transport_lbtrm_activity_timeout`  
(`../Config/transportlbt-rmoperationoptions.html#RECEIVERTRANSPORTLBTRMACTIVITYTIMEOUT`)
- `transport_lbtru_activity_timeout`  
(`../Config/transportlbt-ruoperationoptions.html#RECEIVERTRANSPORTLBTRUACTIVITYTIMEOUT`)
- `transport_lbtipc_activity_timeout`  
(`../Config/transportlbt-ipcooperationoptions.html#RECEIVERTRANSPORTLBTIPTACTIVITYTIMEOUT`)

Recommendations:

- set the Late Join retransmit request interval to a value less than its transport's activity timeout value

For more, see [Late join Options \(../Config/latejoinoptions.html\)](#) and also the applicable [Transport LBT-RU Operation Options](#) section.

Example:

```
#
# +-----+
# | To avoid a transport inactivity timeout while requesting Late Join      |
# | retransmissions, set the Late Join retransmit request interval to a value |
# | less than its transport's activity timeout.                            |
# +-----+
#
receiver retransmit_request_generation_interval 10000
receiver transport_lbtrm_activity_timeout      60000
```

## 2.8.10. Preventing Undetected Loss

It is possible that an unrecoverable loss due to unsatisfied NAKs or a transport activity timeout may go unreported if the delivery controller loss check is disabled or has too long an interval. For UMP stores, the loss check interval must be enabled. Two options (three, if using LBT-RM) are interrelated and must be set according to the guidelines below.

Interrelated Options:

- `delivery_control_loss_check_interval`  
(`../Config/deliverycontroloptions.html#RECEIVERDELIVERYCONTROLLOSSCHECKINTERVAL`)

- `transport_lbtrm_activity_timeout`  
(../Config/transportlbt-rmoperationoptions.html#RECEIVERTRANSPORTLBTRMACTIVITYTIMEOUT)
- `transport_lbtrm_nak_generation_interval` (../Config/transportlbt-rmreliabilityoptions.html#RECEIVERTRANSPORTLBTRMNAKGENERATIONINTERVAL)
- `transport_lbtru_activity_timeout`  
(../Config/transportlbt-ruoperationoptions.html#RECEIVERTRANSPORTLBTRUACTIVITYTIMEOUT)

Recommendations:

- For LBT-RM, set the transport activity timeout to value greater than the sum of the delivery control loss check interval and the NAK generation interval. Also, set the NAK generation interval to at least 4x the delivery control loss check interval.
- for LBT-RU, set the transport activity timeout to value greater than the delivery control loss check interval
- for UMP, always enable and set accordingly the delivery control loss check interval when configuring a store

For more, see [Delivery Control Options](#) (../Config/deliverycontroloptions.html).

Example:

```
#
# +-----+
# | To avoid undetected or unreported loss, set NAK generation to 4x the delivery|
# | control check interval, and ensure that these two combined are less than the |
# | transport activity timeout |
# +-----+
#
receiver delivery_control_loss_check_interval      2500
receiver transport_lbtrm_activity_timeout          60000
receiver transport_lbtrm_nak_generation_interval  10000
```

# Chapter 3. Common Tasks

This chapter describes some common tasks.

## 3.1. Configuring Multi-Homed Hosts

By default, **UM** will select the first multicast-capable, non-loopback interface for multicast topic resolution. If you are fortunate, on a multi-homed host, the correct interface will be selected. However, this fortuitous selection should *not* be relied upon. Moving the interface card to a different slot, a change in the operating system kernel, and numerous other factors can lead to a different ordering of interfaces as reported by the operating system. This in turn can lead **UM** to select a different interface after the change.

It is strongly recommended that the actual interface be specified. The *resolver\_multicast\_interface* option allows you to explicitly specify the multicast interface. Note that this also changes the interface for LBT-RM and multicast immediate messaging.

Other options of interest include *resolver\_unicast\_interface* when using the unicast resolver, *request\_tcp\_interface* when using the request/response messaging, *transport\_lbtru\_interface* and *transport\_tcp\_interface* for receivers, and *transport\_lbtru\_interface* and *transport\_tcp\_interface* for sources, depending on the transport being used.

## 3.2. Traversing a Firewall

To use **UM** across a firewall, several port options may need to be changed. The options of interest include:

- Multicast resolver: *resolver\_multicast\_port*.
- Unicast resolver: *resolver\_unicast\_port*, *resolver\_unicast\_port\_low*, *resolver\_unicast\_port\_high*, and *resolver\_unicast\_destination\_port*.
- TCP transport: *transport\_tcp\_port\_low* and *transport\_tcp\_port\_high* for contexts, and *transport\_tcp\_port* for sources.
- LBT-RM transport: *transport\_lbtrm\_source\_port\_low*, *transport\_lbtrm\_source\_port\_high* for contexts, and *transport\_lbtrm\_destination\_port* for sources.
- LBT-RU transport: *transport\_lbtru\_port\_low* and *transport\_lbtru\_port\_high* for contexts, *transport\_lbtru\_port* for sources, *transport\_lbtru\_port\_low* and *transport\_lbtru\_port\_high* for receivers.
- Multicast immediate messaging: *mim\_destination\_port*, *mim\_incoming\_destination\_port*, and *mim\_outgoing\_destination\_port*.
- Requests: *request\_tcp\_port*, *request\_tcp\_port\_low*, and *request\_tcp\_port\_high*.

In addition, since machines acting as a firewall are often multi-homed as well, consult the section on Configuring Multi-Homed Hosts for additional considerations.

### 3.3. Running Multiple Applications

If you are running multiple **UM** applications on the same machine, using the same (or the default) configuration, you may encounter problems due to the way **UM** allocates and uses ports. The **UM Knowledgebase** (<https://communities.informatica.com/infakb/kbexternal/default.aspx>) contains an article on **Address and Port Usage** which explains how to handle this situation.

# Chapter 4. Reference

## 4.1. Introduction

### 4.1.1. Case Sensitivity

All **Ultra Messaging** scope, option, and value strings are case-insensitive. Thus, any of `context`, `CONTEXT`, and `Context` are recognized as specifying the "context" scope.

### 4.1.2. Specifying Interfaces

The `*_interface` options require a network interface, usually supplied as a string (from a config file via `lbm_config()` or in source code via `*_attr_str_setopt()`), the syntax used for network interface specifications is:

```
a.b.c.d/num
```

where `num` is the number of leading 1 bits in the netmask. If the `/num` is omitted, it defaults to 32 (netmask 255.255.255.255), which means that it must be an exact match for the interface's IP address. However, if `/num` is supplied, it tells **Ultra Messaging** to find an interface within that network. This makes it easier to share a configuration file between many (possibly multi-homed) machines on the same network. For example:

```
context resolver_unicast_interface 192.168.0.0/24
```

specifies a netmask of 255.255.255.0 and would match the interface `192.168.0.3` on one host, and `192.168.0.251` on another host. You can also set network interfaces by name. When setting a configuration option's interface by name, you must use quotes, as illustrated below.

```
context resolver_unicast_interface "interfacename"
```

### 4.1.3. Socket Buffer Sizes

When specifying send or receive socket buffer sizes, keep the following platform-specific information in mind.

- Linux

The kernel value `net.core.rmem_max` dictates the highest value allowed for a receive socket. The kernel value `net.core.wmem_max` dictates the highest value allowed for a sending socket. Increase these values to increase the amount of buffering allowed.

- Windows

Windows should allow socket buffer sizes to be set very high if needed without requiring registry changes.

See our whitepaper [Topics in High Performance Messaging](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www) ([http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM\\_Topics\\_in\\_High\\_Performance\\_WP\\_www](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www)) for background and guidelines on UDP buffer sizing.

### 4.1.4. Reference Entry Format

This section describes the format of each option reference entry.

Each entry begins with a brief description of the option. Following the description is a series of items which define the permissible usage and describes the values for the option.

- Scope  
Defines the scope to which the option applies.
- Type  
Defines the data type of the option. The type is required for calls to the `*_setopt()` and `*_getopt()` API functions.
- Units  
Defines the units in which the option value is expressed. This item is optional.
- Default value  
For range-valued options, indicates the base default value for the option.
- Byte order  
For options whose value is an IP address or port, defines the byte ordering (Host or Network) expected by the API for `*_setopt()` calls, and returned by the API for `*_getopt()` calls.
- May be set during operation  
If an option may be set during operation, it is so indicated here.

Next, for enumerated-valued options, a table is provided which details the permissible string and integer values, a description of each value, and optional notes for each value. The default value is noted within the description.

Alternately, for switch-valued options, a table is provided which describes the meaning of each of the two possible values. The default value is noted within the description.

### 4.1.5. Network Compatibility Mode

This section lists the values for the `network_compatibility_mode` (`../Config/majoroptions.html#CONTEXTNETWORKCOMPATIBILITYMODE`) option, that attempts to maintain wire-level backwards compatibility with older releases by blocking the sending of some (though possibly not all) newer message header types. An application using an older UM release typically logs a warning message when receiving an unknown message header type that did not yet exist in that older release. In a mixed UM version environment, Informatica recommends that your applications filter these unknown message header warning log messages. This option should only be used if such filtering is undesired or not possible.

Note that this option **does not** change any internal behaviors. It merely prevents the sending new message header types which disables any new functionality that relies on the new message header types for both old and new



applications. Other than the warning log message for an application using an earlier release, new message header types do not cause any harm.

String Value	Integer Value	Description
default	LBM_CTX_ATTR_NET_COMPAT_MODE_DEFAULT	Network compatibility mode is disabled. UM sends all new message header types.
LBM_3.6	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_3_6	Block any message headers that only an <b>LBM</b> 3.6 or newer application would understand.
LBM_3.6.1	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_3_6_1	Block any message headers that only an <b>LBM</b> 3.6.1 or newer application would understand.
LBM_3.6.2	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_3_6_2	Block any message headers that only an <b>LBM</b> 3.6.2 or newer application would understand.
LBM_3.6.5	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_3_6_5	Block any message headers that only an <b>LBM</b> 3.6.5 or newer application would understand.
LBM_4.0	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_0	Block any message headers that only an <b>LBM</b> 4.0 or newer application would understand.
LBM_4.0.1	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_0_1	Block any message headers that only an <b>LBM</b> 4.0.1 or newer application would understand.
LBM_4.1	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_1	Block any message headers that only an <b>LBM</b> 4.1 or newer application would understand.
LBM_4.1.1	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_1_1	Block any message headers that only an <b>LBM</b> 4.1.1 or newer application would understand.
LBM_4.1.2	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_1_2	Block any message headers that only an <b>LBM</b> 4.1.1 or newer application would understand.
LBM_4.1.3	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_1_3	Block any message headers that only an <b>LBM</b> 4.1.1 or newer application would understand.
LBM_4.2.1	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_1	Block any message headers that only an <b>LBM</b> 4.2.1 or newer application would understand.
LBM_4.2.2	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_2	Block any message headers that only an <b>LBM</b> 4.2.2 or newer application would understand.

String Value	Integer Value	Description
LBM_4.2.3	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_3	Block any message headers that only an <b>LBM</b> 4.2.3 or newer application would understand.
LBM_4.2.4	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_4	Block any message headers that only an <b>LBM</b> 4.2.4 or newer application would understand.
LBM_4.2.5	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_5	Block any message headers that only an <b>LBM</b> 4.2.5 or newer application would understand.
LBM_4.2.6	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_6	Block any message headers that only an <b>LBM</b> 4.2.6 or newer application would understand.
LBM_4.2.7	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_7	Block any message headers that only an <b>LBM</b> 4.2.7 or newer application would understand.
LBM_4.2.8	LBM_CTX_ATTR_NET_COMPAT_MODE_LBM_4_2_8	Block any message headers that only an <b>LBM</b> 4.2.8 or newer application would understand.
UME_3.0	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_0	Block any message headers that only an <b>UME</b> 3.0 or newer application would understand.
UME_3.0.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_0_1	Block any message headers that only an <b>UME</b> 3.0.1 or newer application would understand.
UME_3.0.2	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_0_2	Block any message headers that only an <b>UME</b> 3.0.2 or newer application would understand.
UME_3.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_1	Block any message headers that only an <b>UME</b> 3.1 or newer application would understand.
UME_3.1.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_1_1	Block any message headers that only an <b>UME</b> 3.1.1 or newer application would understand.
UME_3.1.2	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_1_2	Block any message headers that only an <b>UME</b> 3.1.2 or newer application would understand.
UME_3.1.3	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_1_3	Block any message headers that only an <b>UME</b> 3.1.3 or newer application would understand.
UME_3.2.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_1	Block any message headers that only an <b>UME</b> 3.2.1 or newer application would understand.

String Value	Integer Value	Description
UME_3.2.2	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_2	Block any message headers that only an <b>UME</b> 3.2.2 or newer application would understand.
UME_3.2.3	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_3	Block any message headers that only an <b>UME</b> 3.2.3 or newer application would understand.
UME_3.2.4	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_4	Block any message headers that only an <b>UME</b> 3.2.4 or newer application would understand.
UME_3.2.5	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_5	Block any message headers that only an <b>UME</b> 3.2.5 or newer application would understand.
UME_3.2.6	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_6	Block any message headers that only an <b>UME</b> 3.2.6 or newer application would understand.
UME_3.2.7	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_7	Block any message headers that only an <b>UME</b> 3.2.7 or newer application would understand.
UME_3.2.8	LBM_CTX_ATTR_NET_COMPAT_MODE_UME_3_2_8	Block any message headers that only an <b>UME</b> 3.2.8 or newer application would understand.
UMQ_1.0	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_1_0	Block any message headers that only an <b>UMQ</b> 1.0 or newer application would understand.
UMQ_1.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_1_1	Block any message headers that only an <b>UMQ</b> 1.1 or newer application would understand.
UMQ_1.1.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_1_1_1	Block any message headers that only an <b>UMQ</b> 1.1.1 or newer application would understand.
UMQ_2.0	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_0	Block any message headers that only an <b>UMQ</b> 2.0 or newer application would understand.
UMQ_2.0.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_0_1	Block any message headers that only an <b>UMQ</b> 2.0.1 or newer application would understand.
UMQ_2.1.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_1	Block any message headers that only an <b>UMQ</b> 2.1.1 or newer application would understand.
UMQ_2.1.3	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_3	Block any message headers that only an <b>UMQ</b> 2.1.3 or newer application would understand.

String Value	Integer Value	Description
UMQ_2.1.4	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_4	Block any message headers that only an <b>UMQ</b> 2.1.4 or newer application would understand.
UMQ_2.1.5	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_5	Block any message headers that only an <b>UMQ</b> 2.1.5 or newer application would understand.
UMQ_2.1.6	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_6	Block any message headers that only an <b>UMQ</b> 2.1.6 or newer application would understand.
UMQ_2.1.7	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_7	Block any message headers that only an <b>UMQ</b> 2.1.7 or newer application would understand.
UMQ_2.1.8	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_8	Block any message headers that only an <b>UMQ</b> 2.1.8 or newer application would understand.
UMQ_2.1.9	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_9	Block any message headers that only an <b>UMQ</b> 2.1.9 or newer application would understand.
UMQ_2.1.10	LBM_CTX_ATTR_NET_COMPAT_MODE_UMQ_2_1_10	Block any message headers that only an <b>UMQ</b> 2.1.10 or newer application would understand.
UM_5.0	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_0	Block any message headers that only an <b>UM</b> 5.0 or newer application would understand.
UM_5.0.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_0_1	Block any message headers that only an <b>UM</b> 5.0.1 or newer application would understand.
UM_5.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_1	Block any message headers that only an <b>UM</b> 5.1 or newer application would understand.
UM_5.1.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_1_1	Block any message headers that only an <b>UM</b> 5.1.1 or newer application would understand.
UM_5.1.2	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_1_2	Block any message headers that only an <b>UM</b> 5.1.2 or newer application would understand.
UM_5.2	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_2	Block any message headers that only an <b>UM</b> 5.2 or newer application would understand.
UM_5.2.1	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_2_1	Block any message headers that only an <b>UM</b> 5.2.1 or newer application would understand.

String Value	Integer Value	Description
UM_5.2.2	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_2_2	Block any message headers that only an <b>UM 5.2.2</b> or newer application would understand.
UM_5.3	LBM_CTX_ATTR_NET_COMPAT_MODE_UM_5_3	Block any message headers that only an <b>UM 5.3</b> or newer application would understand.

## 4.2. Major Options

Options in this group have a major impact on the operation of **Ultra Messaging**. Most **UM** application developers will need to be aware of the default values of these options or perhaps override them.

### 4.2.1. context\_event\_function (context)

Callback function (and associated client data pointer) that is called when a context event occurs. This callback may be called inline or from an event queue, if one is given. If called inline, the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

*Scope:* context  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 1.0.

### 4.2.2. context\_name (context)

The name of the context, limited to 128 alphanumeric characters, hyphens or underscores.

*Scope:* context  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.3/UMP 3.3/UMQ 2.3.

### 4.2.3. fd\_management\_type (context)

Define the mechanism **UM** uses for socket file descriptor (FD) management. For more information, search on "file descriptors" in the Informatica Knowledge Base (<https://communities.informatica.com/infakb/kbexternal/default.aspx/>).

**Scope:** context  
**Type:** int  
**When to Set:** Can only be set during object initialization.

String value	Integer value	Description
poll	LBM_CTX_ATTR_FDTYPE_POLL	FD management uses <code>poll()</code> . <i>Unix only.</i>
select	LBM_CTX_ATTR_FDTYPE_SELECT	FD management uses <code>select()</code> . <b>Default for Unix.</b> <i>Unix only.</i>
epoll	LBM_CTX_ATTR_FDTYPE_EPOLL	FD management uses <code>epoll()</code> . <i>Linux kernel 2.6 or later only.</i>
devpoll	LBM_CTX_ATTR_FDTYPE_DEVPOLL	FD management uses the <code>/dev/poll</code> driver. <i>Solaris 8 or later only.</i>
kqueue	LBM_CTX_ATTR_FDTYPE_KQUEUE	FD management uses the BSD <code>kqueue</code> notification system. <i>Mac OS X only.</i>
wsaeventselect	LBM_CTX_ATTR_FDTYPE_WSAEV	FD management uses <code>WSAEventSelect()</code> and <code>WaitForMultipleObjects()</code> . Creates a limit of 64 file descriptors. <b>Default for Windows.</b> <i>Windows only.</i>
wincomport	LBM_CTX_ATTR_FDTYPE_WINCPORT	FD management uses Windows completion ports and completion routines. Disables the 64 file descriptor limit set by <code>WSAEventSelect()</code> . <i>Windows XP or later only.</i>

#### 4.2.4. message\_selector (receiver)

Enables **UM** to pass a message selector string to any receiver. The value must be an expression that conforms to JMS message selector syntax as defined in the Oracle JMS specification. For a **UM** receiver used with **UMP**, please see the Native Applications section in UM JMS Guide.

**Scope:** receiver  
**Type:** string  
**Default value:** NULL  
**When to Set:** Can only be set during object initialization.  
**Version:** This option was implemented in UMQ 5.3.

#### 4.2.5. network\_compatibility\_mode (context)

This option attempts to maintain wire-level backwards compatibility with older releases by blocking the sending of some (though possibly not all) newer message header types. See Network Compatibility Mode ([../Config/reference.html#NETWORKCOMPATIBILITY](#)) for more information and option values.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2/UMQ 2.1.

### 4.2.6. operational\_mode (context)

The mode in which **UM** operates to process events. Refer to Embedded and Sequential Mode ([../Design/architecture.html#EMBEDDED-MODE](#)) for additional information.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
embedded	LBM_CTX_ATTR_OP_EMBEDDED	A thread is spawned within <b>UM</b> to handle processing of events (timers and socket events). <b>Default for all.</b>
sequential	LBM_CTX_ATTR_OP_SEQUENTIAL	The application is responsible for calling <code>lbm_context_process_events()</code> to process events. Sequential mode does not support Multi-Transport Threads ( <a href="#">../Design/fundamental-concepts.html#MULTITHREADED-TRANSPORTS</a> ).

### 4.2.7. ordered\_delivery (receiver)

For LBT-RM, LBT-RU, TCP-LB or LBT-IPC transport sessions only. (This option also applies to TCP when using Late Join because the Late Join messages are not part of the TCP message stream.) Indicates whether or not the topic should have its data delivered in order and reassembled. The default value guarantees ordering and reassembly of large messages. Reassembly of large messages is optional. Changing this option from the default value to a value of 0 (zero) results in messages being delivered as soon as they arrive. Value -1 allows arrival order delivery after the reassembly of large messages. See also Ordered Delivery ([../Design/architecture.html#ORDERED-DELIVERY](#)) for more information about large message fragmentation and reassembly.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
"1" (Integer value as a string.)	1	<b>UM</b> delivers topic messages to a receiver in-order and reassembles large messages. <b>Default for all.</b>
"0" (Integer value as a string.)	0	<b>UM</b> delivers topic messages to a receiver as they arrive and may be out of order. Duplicate delivery is possible. <b>UM</b> delivers large messages as individual fragments of less than the maximum datagram size for the transport in use.
"-1" (Integer value as a string.)	-1	<b>UM</b> delivers topic messages to a receiver as they arrive and may be out of order. Duplicate delivery is possible. However, <b>UM</b> reassembles large messages. Your application can use the <code>sequence_number</code> field of <code>lbm_msg_t</code> objects to order or discard messages.

#### 4.2.8. rcv\_sync\_cache (receiver)

**Ultra Messaging Cache** only - a valid cache address (such as TCP:192.168.5.11:4567) in the standard form of `TCP:address:port` enables a **UM** receiver to use **UMCache** to receive a snapshot of larger, multiple-field messages stored by **UMCache**. Receiving applications can then become synchronized with the live stream of messages sent on the receiver's topic. `address` is the IP address of the machine where the **UMCache** runs and `port` is the configured port where the cache request handler listens.

*Scope:* receiver  
*Type:* umcache\_reqlib\_request\_info\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0/UMP 5.0/UMQ 5.0

#### 4.2.9. rcv\_sync\_cache\_timeout (receiver)

**Ultra Messaging Cache** only - The maximum time period that a **UM** receiver waits for a snapshot message from the **UMCache**.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 2000 (2 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0/UMP 5.0/UMQ 5.0



### 4.2.10. receive\_thread\_pool\_size (context)

For LBT-RM, LBT-RU, or TCP-LB transport sessions only. Defines the maximum number of threads available for transports (excluding the context thread). See [Multi-Transport Threads](#) ([../Design/fundamental-concepts.html#MULTITHREADED-TRANSPORTS](#)).

<i>Scope:</i>	context
<i>Type:</i>	int
<i>Default value:</i>	4
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1/UME 3.1.

### 4.2.11. resolver\_context\_advertisement\_interval (context)

Interval between context advertisements. Setting this option to 0 disables context advertisements, though gateway and other functionality depends upon context advertisements, so a value of 0 is not generally recommended.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	10000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1

### 4.2.12. resolver\_source\_notification\_function (context)

Callback function (and associated client data pointer) that is called when a new source is seen for any topic. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

<i>Scope:</i>	context
<i>Type:</i>	lbm_src_notify_func_t
<i>Default value:</i>	NULL
<i>When to Set:</i>	Can only be set during object initialization.
<i>Config File:</i>	Cannot be set from an UM configuration file.

### 4.2.13. source\_cost\_evaluation\_function (context)

Callback function that you can use in the `lbm_src_cost_function_cb()` to evaluate or determine the cost of a message path. The UM Gateway evaluates the cost of any new topic it detects. The callback supplied with this option

can affect the cost of topics to bias the UM Gateway toward certain message paths. A value of NULL for the callback turns off the callback being called. See also [Applications Can Also Set the Topic Cost](#) ([../Gateway/concepts.html#APPS-SET-TOPIC-COST](#)).

*Scope:* context  
*Type:* lbm\_src\_cost\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.  
*Version:* This option was implemented in UMS 5.0/UMP 5.0/UMQ 5.0

#### 4.2.14. source\_event\_function (context)

Callback function (and associated client data pointer) that is called when a context source event (such as a multicast immediate mode source wakeup event) occurs. This callback may be called inline or from an event queue, if one is given. If called inline, the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

*Scope:* context  
*Type:* lbm\_context\_src\_event\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

#### 4.2.15. source\_includes\_topic\_index (context)

Determines whether the topic index is included in the source string generated for messages and new source notifications.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0.

Value	Description
1	Indicates the topic index should be included in the source string. <b>Default for all.</b>
0	Indicates the topic index should not be included.

## 4.2.16. transport (source)

The transport type to be used for created sources.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
tcp	LBM_SRC_TOPIC_ATTR_TRANSPORT_TCP	TCP over IPv4 <b>Default for all.</b>
lbtrm, lbt-rm	LBM_SRC_TOPIC_ATTR_TRANSPORT_LBTRM	UDP-based reliable multicast with unicast NAKs
lbtru, lbt-ru	LBM_SRC_TOPIC_ATTR_TRANSPORT_LBTRU	UDP-based reliable unicast with unicast NAKs
lbtipc, lbt-ipc	LBM_SRC_TOPIC_ATTR_TRANSPORT_LBTIPC	InterProcess Communication between processes on the same host using a shared memory area.
lbtrdma, lbt-rdma	LBM_SRC_TOPIC_ATTR_TRANSPORT_LBTRDMA	Mellanox® InfiniBand Remote Direct Memory Access transport between hosts using a shared memory area.

## 4.2.17. transport\_demux\_tablesz (receiver)

Specifies the size of the table used for storing receiver delivery controllers used by **UM** for message delivery. Must be a power of two (1, 2, 4, 8, 16, etc.). If not a power of two, **UM** generates a log warning and uses the next highest power of two. For most use cases with low to moderate numbers of topics per transport session, the default suffices. For large numbers of topics and in cases where the lowest latency is desired, set the option to the next highest power of two for the number of topics expected on the transport session.

*Scope:* receiver  
*Type:* size\_t  
*Default value:* 1  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2.

## 4.2.18. transport\_session\_multiple\_sending\_threads (context)

Flag to indicate the application intends to use multiple sending threads per transport session.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Indicates the application does intend to use multiple sending threads per transport session and that <b>UM</b> should make that assumption. <b>Default for all.</b>
0	Indicates the application does not intend to use multiple sending threads per transport session and that <b>UM</b> should make that assumption.

#### 4.2.19. transport\_source\_side\_filtering\_behavior (source)

The filtering behavior desired when TCP and LBT-RU clients are connected. Any other value besides none requires that the clients send unicast messages to the source. These control messages are sent to the TCP request port of the senders context and processed internally. This option affects the transport session underlying the source rather than the source itself. Refer to [Source Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
none	LBM_SRC_TOPIC_ATTR_SSF_NONE	The source sends all data to all clients regardless of the topics they are listening to. <b>Default for all.</b>
inclusion	LBM_SRC_TOPIC_ATTR_SSF_INCLUSION	The source sends only that data to a client that the client specifically requests.

#### 4.2.20. transport\_topic\_sequence\_number\_info\_active\_threshold (source)

Duration in seconds that an inactive source sends contiguous Topic Sequence Number Info (TSNI) messages. (Inactive sources send TSNI messages according to the `transport_topic_sequence_number_info_interval`.) A value of 0 indicates that sources continue sending TSNI until data messages resume, with no timeout. See also [Interrelated Configuration Options](#) ([../Config/grouprelated.html](#)).

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* seconds  
*Default value:* 60  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.2.21. transport\_topic\_sequence\_number\_info\_interval (source)

The interval between Topic Sequence Number Info (TSNI) messages that a source sends. TSNI messages are enabled on all transports and they carry the topic sequence number of the latest message sent by the source. The interval is also a source inactivity threshold. In other words, a source does not send TSNI messages during normal data transmission, but once the source is inactive for as long as this interval, it starts sending TSNI messages. A value of 0 turns off TSNI messages for the source. See also Interrelated Configuration Options ([../Config/grouprelated.html](#)).

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 5000 (5 second)  
*When to Set:* Can only be set during object initialization.

### 4.2.22. use\_extended\_reclaim\_notifications (source)

Specifies which reclaim notification your application receives. The expanded notification, LBM\_SRC\_EVENT\_UME\_MESSAGE\_RECLAIMED\_EX, contains a flag, LBM\_SRC\_EVENT\_UME\_MESSAGE\_RECLAIMED\_EX\_FLAG\_FORCED that **UMP** sets if the reclamation is a forced reclaim.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2.

Value	Description
1	Indicates your application receives the expanded reclaim notification. <b>Default for all.</b>
0	Indicates your application receives the standard reclaim notification that is identical to the expanded notification but without the "Forced" flag.

### 4.2.23. use\_transport\_thread (receiver)

For LBT-RM, LBT-RU, or TCP-LB transport sessions only. Determines whether **UM** uses a thread from the receiver thread pool to process message data or if it uses the context thread, which is the default. See Multi-Transport Threads ([../Design/fundamental-concepts.html#MULTITHREADED-TRANSPORTS](#)).

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

**Version:** This option was implemented in LBM 4.1/UME 3.1.

String value	Integer value	Description
"1" (Integer value as a string.)	1	<b>UM</b> uses a thread from the receiver thread pool.
"0" (Integer value as a string.)	0	<b>UM</b> uses the context thread to process message data. <b>Default for all.</b>

## 4.3. Resolver Operation Options

See Topic Resolution ([../Design/architecture.html#TOPIC-RESOLUTION](#)) for more information.

The following topic resolution options have been deprecated in **LBM** Version 4.0.

- `resolver_active_source_interval`
- `resolver_active_threshold`
- `resolver_maximum_advertisements`
- `resolver_maximum_queries`
- `resolver_query_interval`

See Re-establish Pre-4.0 Topic Resolution ([../Config/disable-topic-res.html#PRE40-TR](#)) for option values that configure the topic resolution used in **LBM** Version 3.6 and prior versions. You should also comment out or remove from your **Ultra Messaging** Configuration file the deprecated configuration options shown above.

### 4.3.1. Minimum Values for Advertisement and Query Intervals

These intervals have the following effective minimal values.

- 10 ms for Initial Phase Advertisements
- 20 ms for Initial Phase Queries
- 30 ms Wildcard Queries
- 100 ms for Sustaining Phase Advertisements and Queries

These effective minimums exist because the internal timer that schedules advertisements and queries fires at the stated interval, i.e., every 10 ms for Initial Phase Advertisements, every 20 ms for Initial Phase Queries, etc. If you set the option's value below the minimum, after the the initial advertisement or query at 0 ms, the resolver schedules the second advertisement or query at the first timer "tick", which is the minimum. Subsequent advertisements or

queries can only be issued at the next timer "tick". If you increase this option from the default to a value that is not a multiple of the minimum, the resolver maintains the rate you establish as an average over subsequent "ticks".

As an example, If you set `resolver_advertisement_sustain_interval` (`../Config/resolveroperationoptions.html#SOURCERESOLVERADVERTISEMENTSUSTAININTERVAL`) or `resolver_query_sustain_interval` (`../Config/resolveroperationoptions.html#RECEIVERRESOLVERQUERYUSUSTAININTERVAL`) at 10 ms, the resolver schedules the second advertisement or query after the initial (0 ms) at the first timer "tick", which is 100 ms. Subsequent advertisements or queries can only be issued at the next timer "tick" (every 100 ms). If you increase either option from the default to 1.25 seconds, for example and not a multiple of 100 ms, the resolver maintains the rate you establish as an average over subsequent "ticks". That is, the second advertisement or query goes out at the 1300 ms "tick". The resolver tracks the tardiness of this advertisement (50 ms) and adjusts the next advertisement or query, which goes out at 2500 ms, giving an average of 1250 ms or 1.25 seconds.

### 4.3.2. `disable_extended_topic_resolution_message_options` (context)

This is a topic resolution compatibility option that, when set to 1, lets LBM 4.0 (or later) installations work with LBM 3.5.3 / UME 2.2.4 (or earlier) installations. If you do not have early-version installations in the network, leave this option at 0.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

Value	Description
1	Enable compatibility with earlier-version installations (and disable some message structure features).
0	Normal current-version compatibility. Strongly recommended. <b>Default for all.</b>

### 4.3.3. `resolution_no_source_notification_threshold` (receiver)

The threshold for the number of unanswered topic resolution queries before **UM** delivers a `LBM_MSG_NO_SOURCE_NOTIFICATION` for the topic. The receiver does not stop querying after the delivery of this notification. A value of 0 indicates no notifications will be sent.

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* Number of queries  
*Default value:* 0 (do not notify)

*When to Set:* May be set during operation.

#### 4.3.4. `resolution_number_of_sources_query_threshold` (receiver)

The threshold for the number of sources a topic must have before topic resolution queries are not sent. A value of zero results in no topic resolution queries being generated. See also Disabling Aspects of Topic Resolution ([disable-topic-res.html](#)).

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* Number of sources  
*Default value:* 10000000 (10 million)  
*When to Set:* May be set during operation.

#### 4.3.5. `resolver_advertisement_maximum_initial_interval` (source)

The longest - and last - interval in the initial phase of topic advertisement. A value of 0 disables the initial phase of advertisement. See also Disabling Aspects of Topic Resolution ([disable-topic-res.html](#)).

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

#### 4.3.6. `resolver_advertisement_minimum_initial_duration` (source)

The duration of the initial phase of topic advertisement. A value of 0 guarantees that the initial phase of advertisement never completes.

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 5000 (5 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0



### 4.3.7. resolver\_advertisement\_minimum\_initial\_interval (source)

Interval between the first topic advertisement sent upon creation of the source and the second advertisement sent by the source. A value of 0 disables the initial phase of advertisement. See also Disabling Aspects of Topic Resolution (disable-topic-res.html). This option has an effective minimum of 10 ms. See Minimum Values for Advertisement and Query Intervals (resolveroperationoptions.html).

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10 (0.01 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.8. resolver\_advertisement\_minimum\_sustain\_duration (source)

The duration of the sustaining phase of topic advertisement. A value of 0 guarantees that the sustaining phase of advertising never completes.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* seconds  
*Default value:* 60 (1 minute)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.9. resolver\_advertisement\_send\_immediate\_response (source)

Allows you to disable the normal immediate response to queries and wildcard queries. Sources normally send topic advertisements (TIR) immediately in response to topic queries (TQR) for a local topic or wildcard queries (WC-TQR) with a pattern that matches a local topic. If you configure sources to delay sending advertisements, **UM** delays advertisements by the limits defined by the advertisement rate limiter options, `resolver_*_bps` and `resolver_*_per_second`.

*Scope:* source  
*Type:* lbm\_uint\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2/UMQ 2.1

Value	Description
-------	-------------

Value	Description
1	Sources immediately send advertisements (TIR) in response to topic queries (TQR) or wildcard queries (WC-TQR). <b>Default for all.</b>
0	Sources delay sending advertisements (TIR) in response to topic queries (TQR) or wildcard queries (WC-TQR).

#### 4.3.10. resolver\_advertisement\_sustain\_interval (source)

Interval between sending topic advertisements in the sustaining phase of topic advertisement. A value of 0 disables the sustaining phase of advertisement. See also Disabling Aspects of Topic Resolution (disable-topic-res.html). This option has an effective minimum of 100 ms. See Minimum Values for Advertisement and Query Intervals (resolveroperationoptions.html).

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

#### 4.3.11. resolver\_cache (context)

Whether or not to cache topic resolution information. When topic resolution information is not cached, it takes up less memory. However, wildcard receivers will only see topics that have other **UM** receivers created. And source notification only occurs for topics that have **UM** receivers created.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Topic resolution information will be cached. <b>Default for all.</b>
0	Do not cache topic resolution information.

#### 4.3.12. resolver\_datagram\_max\_size (context)

The maximum datagram size that can be generated for topic resolution advertisements and queries. The default value

is 8192, the minimum is 500 bytes, and the maximum is 65535.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint_t
<i>Units:</i>	bytes
<i>Default value:</i>	8192
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.6/UME 3.0.

### 4.3.13. resolver\_initial\_advertisement\_bps (context)

Maximum advertisement rate during the initial phase of topic advertisement. A value of 0 sets no rate limit on advertisements in the initial phase of topic advertisement.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint64_t
<i>Units:</i>	bits per second
<i>Default value:</i>	1000000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

### 4.3.14. resolver\_initial\_advertisements\_per\_second (context)

Maximum number of advertisements sent within a one second period during the initial phase of topic advertisement. A value of 0 sets no rate limit on advertisements in the initial phase of topic advertisement.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	advertisements
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

### 4.3.15. resolver\_initial\_queries\_per\_second (context)

Maximum number of queries sent within a one second period during the initial phase of topic querying. A value of 0 sets no rate limit on queries in the initial phase of topic querying.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t

<i>Units:</i>	advertisements
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

#### 4.3.16. resolver\_initial\_query\_bps (context)

Maximum query rate during the initial phase of topic querying. A value of 0 sets no rate limit on queries in the initial phase of topic querying.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint64_t
<i>Units:</i>	bits per second
<i>Default value:</i>	1000000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

#### 4.3.17. resolver\_query\_maximum\_initial\_interval (receiver)

The longest - and last - interval in the initial phase of topic querying. A value of 0 disables the initial phase of querying. See also Disabling Aspects of Topic Resolution ([disable-topic-res.html](#)).

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	200 (0.2 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

#### 4.3.18. resolver\_query\_minimum\_initial\_duration (receiver)

The duration of the initial phase of topic querying. A value of 0 guarantees that the initial phase of querying never completes.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	5000 (5 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

*Version:* This option was implemented in LBM 4.0

### 4.3.19. resolver\_query\_minimum\_initial\_interval (receiver)

Interval between the first topic query sent upon creation of the receiver and the second query sent by the receiver. A value of 0 disables the initial phase of querying. See also Disabling Aspects of Topic Resolution ([disable-topic-res.html](#)). This option has an effective minimum of 20 ms. See Minimum Values for Advertisement and Query Intervals ([resolveroperationoptions.html](#)).

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 20 (0.02 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.20. resolver\_query\_minimum\_sustain\_duration (receiver)

The duration of the sustaining phase of topic querying. A value of 0 guarantees that the sustaining phase of querying never completes.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* seconds  
*Default value:* 60 (1 minute)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.21. resolver\_query\_sustain\_interval (receiver)

Interval between sending topic queries in the sustaining phase of topic querying. A value of 0 disables the sustaining phase of querying. See also Disabling Aspects of Topic Resolution ([disable-topic-res.html](#)). This option has an effective minimum of 100 ms. See Minimum Values for Advertisement and Query Intervals ([resolveroperationoptions.html](#)).

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)

*When to Set:* Can only be set during object initialization.

*Version:* This option was implemented in LBM 4.0

### 4.3.22. resolver\_receiver\_map\_tablesz (context)

The size of the hash table used for storing receiver topic information used for topic resolution. This value should be a prime number.

*Scope:* context

*Type:* size\_t

*Units:* map entries

*Default value:* 131111

*When to Set:* Can only be set during object initialization.

### 4.3.23. resolver\_send\_initial\_advertisement (source)

Controls whether or not a source sends an advertisement upon creation. Turning off this advertisement speeds source creation and reduces the number of messages on your network through application initialization.

*Scope:* source

*Type:* lbm\_uint\_t

*When to Set:* Can only be set during object initialization.

*Version:* This option was implemented in LBM 4.0

Value	Description
1	Source sends a topic advertisement immediately upon creation. <b>Default for all.</b>
0	Source does not send an advertisement upon creation. This setting does not affect the topic resolution phases you have configured, which execute as expected. See Disabling Aspects of Topic Resolution ( <a href="#">disable-topic-res.html</a> ) for information about altering topic resolution phase advertisements.

### 4.3.24. resolver\_source\_map\_tablesz (context)

The size of the hash table used for storing source topic information used by topic resolution. This value should be a prime number.

*Scope:* context  
*Type:* size\_t  
*Units:* map entries  
*Default value:* 131111  
*When to Set:* Can only be set during object initialization.

### 4.3.25. resolver\_string\_hash\_function (context)

The hash function to use for hashing topic name strings for source and receiver topics. The application may choose from a list of defined hash functions or it may define its own hash function, as identified by the string value of this option. When setting a hash function, note that:

- If set through a configuration file or a call to `lbm_context_attr_str_setopt()`, only the string values `classic`, `djb2`, `sdbm`, or `murmur2` are valid. (If retrieved by a call to `lbm_context_attr_str_getopt()`, one of these string values is returned.)
- If set through a call to `lbm_context_attr_setopt()`, you must pass a pointer to a hash function. Use this method for hash functions other than the four pre-defined functions.

*Scope:* context  
*Type:* lbm\_str\_hash\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
classic		A "classic" good string hash function. Works best when topic names have a constant prefix with a changing suffix.
djb2		The Dan Bernstein algorithm from <code>comp.lang.c</code> . Works best when topic names have a changing prefix with a constant suffix.
sdbm		sdbm database library (used in Berkeley DB). A useful alternative to djb2.
murmur2		Good all-around hash function by Austin Appleby. Best for medium to long topic strings. <b>Default for all.</b>

### 4.3.26. resolver\_string\_hash\_function\_ex (context)

This option is similar to the `resolver_string_hash_function` above, except for the following differences:

- This option can be set via only `lbm_context_attr_setopt()` (not from a configuration file or `lbm_context_attr_str_setopt()`). Hence, this also means you cannot use the string options (`classic`, etc).
- You can pass a string length to the hash function, allowing it to then possibly run faster by operating on multiple-character strings at a time. Note that if -1 is passed in, you must use a `strlen` to calculate the length.
- The hash function accepts a `clientid` pointer, which you can set as needed, and which is passed back in each time the function is called.

This option is the better choice when setting your own custom hash function. Note that both the `resolver_string_hash_function` and `resolver_string_hash_function_ex` options set the same attributes, hence, if you use both (not recommended) one will override the other.

*Scope:* context  
*Type:* `lbm_str_hash_func_ex_t`  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.3.27. resolver\_sustain\_advertisement\_bps (context)

Maximum advertisement rate during the sustaining phase of topic advertisement. A value of 0 sets no rate limit on advertisements in the sustaining phase of topic advertisement.

*Scope:* context  
*Type:* `lbm_uint64_t`  
*Units:* bits per second  
*Default value:* 1000000  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.28. resolver\_sustain\_advertisements\_per\_second (context)

Maximum number of advertisements sent within a one second period during the sustaining phase of topic advertisement. A value of 0 sets no rate limit on advertisements in the sustaining phase of topic advertisement.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* advertisements  
*Default value:* 0



*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.29. resolver\_sustain\_queries\_per\_second (context)

Maximum number of queries sent within a one second period during the sustaining phase of topic querying. A value of 0 sets no rate limit on queries in the sustaining phase of topic querying.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* advertisements  
*Default value:* 0  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.30. resolver\_sustain\_query\_bps (context)

Maximum query rate during the sustaining phase of topic querying. A value of 0 sets no rate limit on queries in the sustaining phase of topic querying.

*Scope:* context  
*Type:* lbm\_uint64\_t  
*Units:* bits per second  
*Default value:* 1000000  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.3.31. resolver\_unicast\_activity\_timeout (context)

Indicates the maximum time between messages from a unicast resolver daemon before **UM** declares it inactive and stops sending normal topic resolution traffic via that daemon. **UM** will still send keepalives to the daemon. A value of 0 will force all resolver daemons to be treated as permanently active.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0

### 4.3.32. resolver\_unicast\_change\_interval (context)

Indicates how often **UM** will change to the next available resolver daemon specified using the. resolver\_unicast\_daemon configuration option. The actual value used is random, and is selected from the range  $(1/2 * \text{change\_interval}, 3/2 * \text{change\_interval})$ . If all resolver daemons have been marked inactive, **UM** enters a quick-change mode where it uses a random value from the range  $(1/4 * \text{change\_interval}, 3/4 * \text{change\_interval})$  in order to more quickly locate an active daemon.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0

### 4.3.33. resolver\_unicast\_check\_interval (context)

Indicates how often a **UM** checks for resolver activity in order to determine liveness. A value of 0 will disable activity checks. This setting only applies to the unicast resolver.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0

### 4.3.34. resolver\_unicast\_force\_alive (context)

Indicates whether sources or receivers in this context should send keepalive messages to a configured Unicast Topic Resolver so they can receive topic resolution traffic.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Send keepalive messages to the Unicast Topic Resolver every 5000ms, if this context has sent no topic resolution traffic during the interval.

Value	Description
0	Send keepalive messages to the Unicast Topic Resolver every 5000ms, regardless of whether this context has sent any other topic resolution traffic during the interval. Default for all. <b>Default for all.</b>

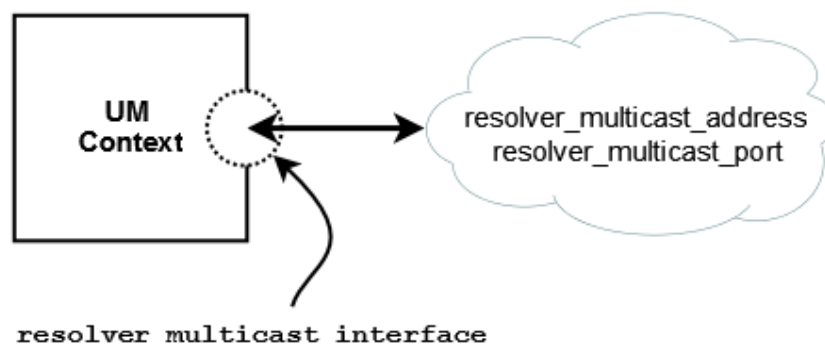
### 4.3.35. resolver\_unicast\_keepalive\_interval (context)

Indicates how often keepalive messages should be sent to a resolver daemon. Keepalives are only sent if no other traffic has been sent since the last keepalive interval expired.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 500  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0

## 4.4. Multicast Resolver Network Options

Figure 4-1. Multicast resolver network options



See also Topic Resolution ([../Design/architecture.html#TOPIC-RESOLUTION](#)) for more information.

#### 4.4.1. resolver\_multicast\_address (context)

Multicast address used for Topic Resolution. This option automatically sets the values for `resolver_multicast_incoming_address` and `resolver_multicast_outgoing_address` as evidenced by the default values for all three options, which are the same.

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.9.10.11  
*When to Set:* Can only be set during object initialization.

#### 4.4.2. resolver\_multicast\_incoming\_address (context)

Incoming multicast address used for finer control of Topic Resolution. For example, if you want the context to listen on a different address/port than the `resolver_multicast_address`, set this option and `resolver_multicast_incoming_port` to different values. This value may be set to 0.0.0.0 (`INADDR_ANY`), to switch off listening to topic resolution messages. This means that queries from receivers or advertisements from sources will not be handled. See also `resolver_multicast_outgoing_address`.

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.9.10.11  
*When to Set:* Can only be set during object initialization.

#### 4.4.3. resolver\_multicast\_incoming\_port (context)

Incoming multicast port used for finer control of Topic Resolution. For example, if you want the context to listen on a different address/port than the `resolver_multicast_port`, set this option and `resolver_multicast_incoming_address` to different values. See also `resolver_multicast_outgoing_port`.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 12965  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

#### 4.4.4. resolver\_multicast\_interface (context)

Specifies which network interface **UM** sends/receives all multicast traffic (Topic Resolution, LBT-RM, Multicast Immediate Messaging). Can specify full IP address of interface, or just network part (see Specifying Interfaces for details). Default is set to default multicast interface as determined by **UM** (the first multicast-capable, non-loopback

interface).

*Scope:* context  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0  
*When to Set:* Can only be set during object initialization.

#### 4.4.5. resolver\_multicast\_outgoing\_address (context)

Outgoing multicast address used for finer control of Topic Resolution. For example, if you want the context to send on a different address/port than the `resolver_multicast_address`, set this option and `resolver_multicast_outgoing_port` to different values. See also `resolver_multicast_incoming_address`.

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.9.10.11  
*When to Set:* Can only be set during object initialization.

#### 4.4.6. resolver\_multicast\_outgoing\_port (context)

Outgoing multicast port used for finer control of Topic Resolution. For example, if you want the context to send on a different address/port than the `resolver_multicast_port`, set this option and `resolver_multicast_outgoing_address` to different values. See also `resolver_multicast_incoming_port`.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 12965  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

#### 4.4.7. resolver\_multicast\_port (context)

Multicast port used for Topic Resolution. This option automatically sets the values for `resolver_multicast_incoming_port` and `resolver_multicast_outgoing_port` as evidenced by the default values for all three options, which are the same.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 12965  
*Byte order:* Network

*When to Set:* Can only be set during object initialization.

#### 4.4.8. resolver\_multicast\_receiver\_socket\_buffer (context)

Value used to set `SO_RCVBUF` value of the resolver receivers. In some cases the OS will not allow all of this value to be used. A value of 0 instructs **UM** to use the default OS values. See the section on socket buffer sizes for platform-dependent information. See also our white paper *Topics in High Performance Messaging* ([http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM\\_Topics\\_in\\_High\\_Performance\\_WP\\_www](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www)) for background and guidelines on UDP buffer sizing.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* bytes  
*Default value:* 0 (use OS default)  
*When to Set:* Can only be set during object initialization.

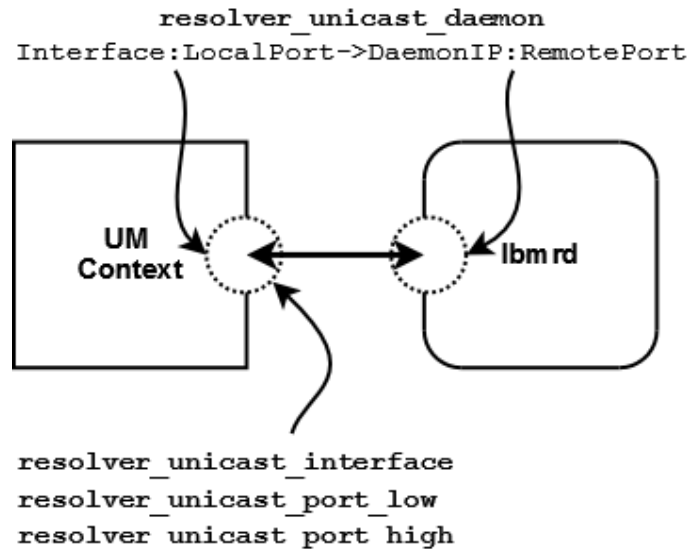
#### 4.4.9. resolver\_multicast\_ttl (context)

The IP TTL (hop count) to use for a Topic Resolution packet. A value of 1 confines the packet to the local network (but may also cause high CPU usage on some routers). Also controls TTL on LBT-RM packets.

*Scope:* context  
*Type:* `lbm_uint8_t`  
*Default value:* 16  
*When to Set:* May be set during operation.

## 4.5. Unicast Resolver Network Options

Figure 4-2. Unicast resolver network options



This diagram shows a single unicast resolver daemon configured with `resolver_unicast_daemon`.

If using multiple `lbmrd` instances with a single context, you can configure `resolver_unicast_interface` and `resolver_unicast_port_low/high` and omit the `Interface:LocalPort` section of `resolver_unicast_daemon`.

See also Unicast Topic Resolution ([../Design/architecture.html#UNICAST-TOPIC-RESOLUTION](#)) for more information.

### 4.5.1. resolver\_unicast\_daemon (context)

Add a unicast resolver daemon specification to the list of unicast resolver daemons. Unlike most other **UM** settings, every time this setting is called, it adds another daemon specification to the list and does NOT overwrite previous specifications. Each entry contains the interface, source port, resolver IP, and destination port for a single daemon. For the configuration file as well as string versions of setting this option, the string value is formatted as `[Iface[:Src_Port]->] IP:Dest_Port`. `Iface` is the interface to use (previously set via `resolver_unicast_interface`). `Src_Port` is the source port to use (previously `resolver_unicast_port`). `IP` is the resolver daemon's IP address (previously `resolver_unicast_address`), `Dest_Port` is the resolver daemon's UDP port (previously `resolver_unicast_destination_port`). Either the `Src_Port` or both the `Iface` and `Src_Port` may be omitted, in which case the default `resolver_unicast_interface` and `resolver_unicast_port` settings are used. Because each entry adds a new daemon specification and does not overwrite previous values, an entry or string with the IP address of 0.0.0.0 and TCP port of 0 removes all previous daemon specifications. At least one daemon specification means the context does not use multicast topic resolution. Possible formats of this option are:

`Interface:LocalPort->DaemonIP:RemotePort`, `Interface->DaemonIP:RemotePort` or `DaemonIP:RemotePort`. `Interface` may be specified in any of the ways described in [Specifying Interfaces](#).

*Scope:* context  
*Type:* lbm\_ucast\_resolver\_entry\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0

### 4.5.2. resolver\_unicast\_interface (context)

Specifies the network interface over which **UM** receives unicast Topic Resolution messages. Can specify full IP address of interface, or just network part (see Specifying Interfaces for details). Default is set to INADDR\_ANY, meaning that it will accept unicast Topic Resolution messages on any interface.

*Scope:* context  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.5.3. resolver\_unicast\_port\_high (context)

The highest local UDP port in a range of ports used for unicast topic resolution messages. The **UM** resolution daemon (lbmr) sends unicast topic resolution messages to the UDP port range defined by this option and resolver\_unicast\_port\_low.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14406  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

### 4.5.4. resolver\_unicast\_port\_low (context)

The lowest local UDP port in a range of ports used for unicast topic resolution messages. The **UM** resolution daemon (lbmr) sends unicast topic resolution messages to the UDP port range defined by this option and resolver\_unicast\_port\_high.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14402  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.



### 4.5.5. resolver\_unicast\_receiver\_socket\_buffer (context)

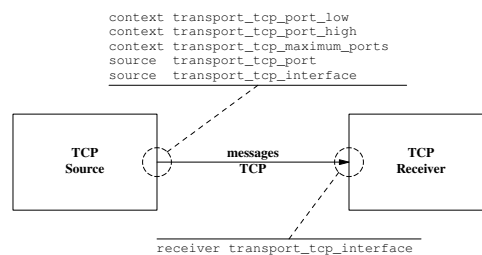
Value used to set `SO_RCVBUF` value of the UDP receivers for unicast topic resolution messages. In some cases the OS will not allow all of this value to be used. A value of 0 instructs **UM** to use the default OS values. See the section on socket buffer sizes for platform-dependent information.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	bytes
<i>Default value:</i>	0 (use OS default)
<i>When to Set:</i>	Can only be set during object initialization.

## 4.6. Transport TCP Network Options

TCP receivers initiate connections toward TCP sources. Messages flow from sources to receivers.

**Figure 4-3. TCP network options**



`transport_tcp_port_low` is the lowest port that **UMS** will allocate for TCP sources in a context; `transport_tcp_port_high` is the highest. No more than `transport_tcp_maximum_ports` ports will be assigned to TCP sources within a single context.

Creation of a **UMS** source on a TCP transport will allocate an unused port from the range if less than `transport_tcp_maximum_ports` ports have already been allocated. Setting `transport_tcp_maximum_ports` to a fraction of the range allows the corresponding multiple number of **UMS** processes to share a common configuration.

If a particular TCP port is desired by a source, it may be given with `transport_tcp_port`. If the desired port is already in use, then an unused port will be sought as described above. A value of 0 (the default) expresses no preference and results in the default open port seeking behavior described above.

`transport_tcp_interface` may be used on TCP sources to choose particular interface, overriding the default `INADDR_ANY` which accepts connections on all interfaces. Similarly, `transport_tcp_interface` may be used on receivers to choose a particular interface for outgoing connections.

### 4.6.1. transport\_tcp\_interface (receiver)

Specifies the network interface to which **UM** receivers bind before connecting to sources. You can specify the full IP address of interface, or just the network part (see Specifying Interfaces for details).

*Scope:* receiver  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.6.2. transport\_tcp\_interface (source)

Specifies the network interface over which **UM** accepts connection requests (from topic receivers). You can specify the full IP address of interface, or just the network part (see Specifying Interfaces for details). Be aware that this option is applied to the transport session when the first topic is created on that session. Thus, setting a different interface for a subsequent topic that maps onto the same transport session will have no effect. Default is set to INADDR\_ANY, meaning that it will not bind to a specific interface. You can also modify the default by setting the option to 0.0.0.0/0 which produces the same result.

*Scope:* source  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.6.3. transport\_tcp\_maximum\_ports (context)

Maximum number of TCP sessions to allocate.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Units:* number of ports  
*Default value:* 10  
*When to Set:* Can only be set during object initialization.

### 4.6.4. transport\_tcp\_port (source)

The preferred TCP port number for this Topic. If 0, the context will attempt to find one in the given TCP port range.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Default value:* 0 (pick open port)

*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

### 4.6.5. transport\_tcp\_port\_high (context)

High port number to assign TCP sessions to.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14390  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

### 4.6.6. transport\_tcp\_port\_low (context)

Low port number to assign TCP sessions to.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14371  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

## 4.7. Transport TCP Operation Options

### 4.7.1. transport\_session\_maximum\_buffer (source)

Value used to control the maximum amount of data buffered in **UM** for the transport session used for the topic. For the normal multiple receiver behavior, this value represents the total buffered by all TCP receivers. For the bounded\_latency and source\_paced multiple receiver behavior, this value represents the individual receiver buffered amount. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* bytes

*Default value:* 65536  
*When to Set:* Can only be set during object initialization.

### 4.7.2. transport\_tcp\_activity\_method (receiver)

For TCP sessions only. The type of timeout method to use for TCP receivers.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.3.8/UME 2.0.6.

String value	Integer value	Description
timer	LBM_RCV_TOPIC_ATTR_TCP_ACTIVITY_T	<b>Timer method</b> that requires new TCP session data to be sent to determine if the connection is alive. <b>Default for all.</b>
SO_KEEPALIVE	LBM_RCV_TOPIC_ATTR_TCP_ACTIVITY_T	<b>Set SO_KEEPALIVE</b> on the TCP connection which uses the TCP keepalive support in the operating system to determine if the connection is alive. When you use the SO_KEEPALIVE method, <b>UM</b> uses transport_tcp_activity_timeout value to set the idle and probe times for SO_KEEPALIVE. The idle time is 90% of the timeout value at most. The probe time is 10% with 10 seconds as the minimum.

### 4.7.3. transport\_tcp\_activity\_timeout (receiver)

For TCP sessions only. The maximum time that a TCP session may be quiescent before it is deleted and an EOS event is delivered for all topics using this transport session. A value greater than zero turns the timer on.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 0  
*When to Set:* Can only be set during object initialization.

### 4.7.4. transport\_tcp\_coalesce\_threshold (source)

The threshold of the maximum number of individual messages that are copied into a single buffer before being sent

to the TCP source internals. When used with small messages, this allows TCP to use less memory at the expense of an additional copy operation of the data before being sent. The default values are also the maximum allowable values.

*Scope:* source  
*Type:* int  
*Units:* number of individual messages  
*Default value:* 1024 for Linux, Microsoft® Windows®; 16 for Solaris, AIX  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 2.3.

### 4.7.5. transport\_tcp\_datagram\_max\_size (context)

The maximum datagram size that can be generated for a TCP transport session. The default value is 65535, the minimum is 500 bytes, and the maximum is 65535.

*Scope:* context  
*Type:* lbm\_uint\_t  
*Units:* bytes  
*Default value:* 65535  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1

### 4.7.6. transport\_tcp\_exclusiveaddr (source)

Applicable only to Windows. Indicate whether the TCP session should set `SO_EXCLUSIVEADDRUSE` or not before it binds. The default setting in Windows allows multiple binds to the same port. By default, **UM** will set `SO_EXCLUSIVEADDRUSE` to minimize port sharing. Refer to Microsoft's web site for more information on `SO_EXCLUSIVEADDRUSE`.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Set <code>SO_EXCLUSIVEADDRUSE</code> . <b>Default for Windows.</b>
0	Do not set <code>SO_EXCLUSIVEADDRUSE</code> .

### 4.7.7. transport\_tcp\_listen\_backlog (source)

The backlog used in the TCP `listen()` call to set the queue length for incoming connections.

*Scope:* source  
*Type:* int  
*Units:* number of queued connections  
*Default value:* 5  
*When to Set:* Can only be set during object initialization.

### 4.7.8. transport\_tcp\_multiple\_receiver\_behavior (source)

The flow control behavior desired when multiple TCP clients are receiving for a TCP session. If an application is only allowed to send as fast as all receivers can consume data, markedly slower receivers will lose data (have unrecoverably lost UM messages) if they can not keep up with the other faster receivers for the TCP session. Note that at high rates and with receivers that can consume data at fairly similar rates, all receivers may experience some loss at times. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to Source Configuration and Transport Sessions ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
normal	LBM_SRC_TOPIC_ATTR_TCP_MULTI_RECV	The application sends as fast as the slowest receiver consumes data. This slows down all receivers on that TCP session. <b>Default for all.</b>
bounded_latency	LBM_SRC_TOPIC_ATTR_TCP_MULTI_RECV	The application sends as fast as the fastest receiver can consume data even if recent data headed for slower receivers must be discarded.
source_paced	LBM_SRC_TOPIC_ATTR_TCP_MULTI_RECV	The application sends as fast as it can even if recent data headed for any or all receivers must be discarded.

### 4.7.9. transport\_tcp\_multiple\_receiver\_send\_order (source)

In the case of multiple receivers, this option determines whether datagrams are sent to each receiver in the established order of receivers, or if receivers are selected in random order for each datagram transmission.

*Scope:* source

*Type:* `lbm_src_topic_attr_t`  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
serial	LBM_SRC_TOPIC_ATTR_TCP_MULTI_RECV	Select receivers to receive a datagram based on current established order. <b>Default for all.</b>
random	LBM_SRC_TOPIC_ATTR_TCP_MULTI_RECV	For each datagram sent, select receivers in random order, for the sake of "fairness". Note that this option adds a small amount of CPU overhead.

#### 4.7.10. transport\_tcp\_nodelay (source)

Whether the TCP sockets used for the transport session should set `TCP_NODELAY` or not. (Setting `TCP_NODELAY` disables Nagle's algorithm.)

*Scope:* `source`  
*Type:* `int`  
*When to Set:* Can only be set during object initialization.

Value	Description
1	TCP transport sockets should set <code>TCP_NODELAY</code> (disable Nagle). <b>Default for all.</b>
0	TCP transport sockets should not set <code>TCP_NODELAY</code> (leave Nagle enabled).

#### 4.7.11. transport\_tcp\_receiver\_socket\_buffer (context)

Value used to set `SO_RCVBUF` value of the TCP receivers for topics. In some cases the OS will not allow all of this value to be used. A value of 0 instructs **UM** to use the default OS values. See the section on socket buffer sizes for platform-dependent information.

*Scope:* `context`  
*Type:* `lbm_ulong_t`  
*Units:* bytes  
*Default value:* 0 (use OS default)  
*When to Set:* Can only be set during object initialization.

### 4.7.12. transport\_tcp\_reuseaddr (source)

Whether the TCP session should set `SO_REUSEADDR` or not before it binds. **WARNING:** This option is not recommended for Microsoft Windows users because the `SO_REUSEADDR` socket option in Windows allows a socket to forcibly bind to a port in use by another socket. Multiple sockets using the same port results in indeterminate behavior.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Set <code>SO_REUSEADDR</code> .
0	Do not set <code>SO_REUSEADDR</code> . <b>Default for all.</b>

### 4.7.13. transport\_tcp\_sender\_socket\_buffer (source)

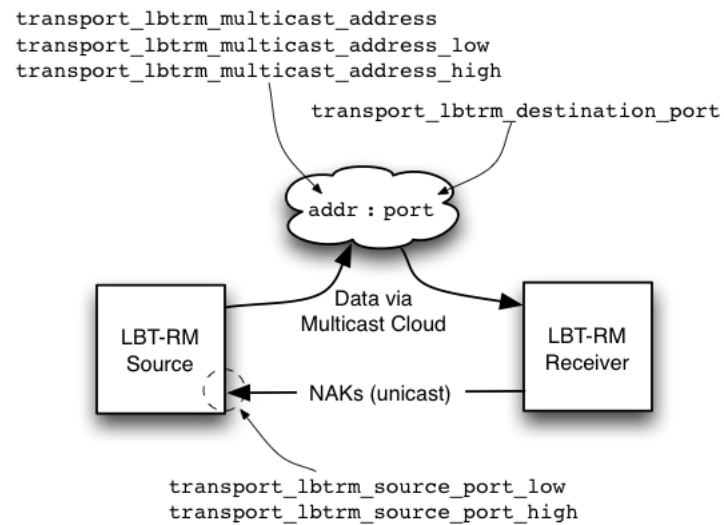
Value used to set the `SO_SNDBUF` value of the TCP session. In some cases the OS will not allow all of this value to be used. A value of 0 instructs **UM** to use the OS defaults. See the section on socket buffer sizes for platform-dependent information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 0 (use OS default)  
*When to Set:* Can only be set during object initialization.

## 4.8. Transport LBT-RM Network Options

The following illustration shows where the various options are applied. Note that for a multi-homed LBT-RM source, the interface LBT-RM multicast resolver interface specified with `resolver_multicast_interface` will be used as the source for LBT-RM.



**Figure 4-4. LBT-RM network options**

### 4.8.1. transport\_lbtrm\_destination\_port (source)

The UDP destination port used for this Topic when LBT-RM is used.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Default value:* 14400  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

### 4.8.2. transport\_lbtrm\_multicast\_address (source)

The preferred multicast address for this Topic when LBT-RM is used. If 0.0.0.0 (INADDR\_ANY), the context will attempt to find one in the given multicast address range.

*Scope:* source  
*Type:* struct in\_addr  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.8.3. transport\_lbtrm\_multicast\_address\_high (context)

Multicast address used as the highest value to assign LBT-RM sessions to.

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.10.10.14  
*When to Set:* Can only be set during object initialization.

### 4.8.4. transport\_lbtrm\_multicast\_address\_low (context)

Multicast address used as the lowest value to assign LBT-RM session to.

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.10.10.10  
*When to Set:* Can only be set during object initialization.

### 4.8.5. transport\_lbtrm\_source\_port\_high (context)

Highest port number value used for LBT-RM source sessions used for unicast NAK processing. NAKs are sent back to this port number for processing and retransmission generation. Each LBT-RM session must use a unique port value. Note that this does not control the UDP source port on the outbound LBT-RM stream.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14399  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

### 4.8.6. transport\_lbtrm\_source\_port\_low (context)

Lowest port number value used for LBT-RM source sessions used for unicast NAK processing. NAKs are sent back to this port number for processing and retransmission generation. Each LBT-RM session must use a unique port value. Note that this does not control the UDP source port on the outbound LBT-RM stream.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14390  
*Byte order:* Host

*When to Set:* Can only be set during object initialization.

## 4.9. Transport LBT-RM Reliability Options

In addition to LBT-RM reliability options, this section discusses the following topics.

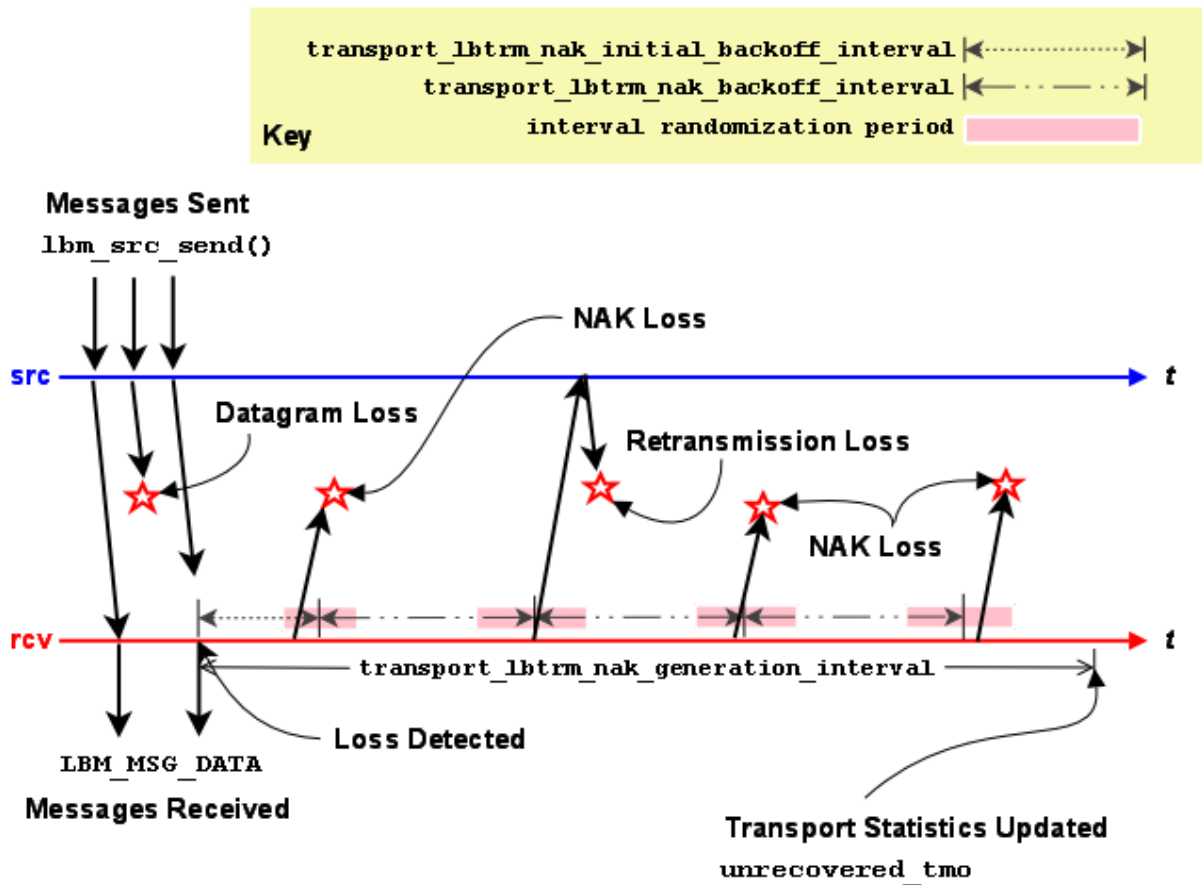
- *LBT-RM Datagram Loss Resulting in Unrecovered Message Loss*
- *LBT-RM Source Ignoring NAKs for Efficiency*
- *LBT-RM Receiver Suppressing NAK Generation*

### 4.9.1. LBT-RM Datagram Loss Resulting in Unrecovered Message Loss

The key options that control the effort that an LBT-RM receiver will make to recover from datagram loss are *transport\_lbtrm\_nak\_backoff\_interval* and *transport\_lbtrm\_nak\_generation\_interval*. Timers for both start when loss is detected. The following timeline illustrates a case where a receiver is notified of unrecoverable message loss following repeated datagram loss.

**Note:** The actual length of the interval randomization periods are between 1/2 and 3/2 of the configured interval value. In Figure 4-5, these periods appear shorter to simplify the diagram.

Figure 4-5. Scenario Timeline: LBT-RM Datagram Loss Resulting in Unrecovered Message Loss



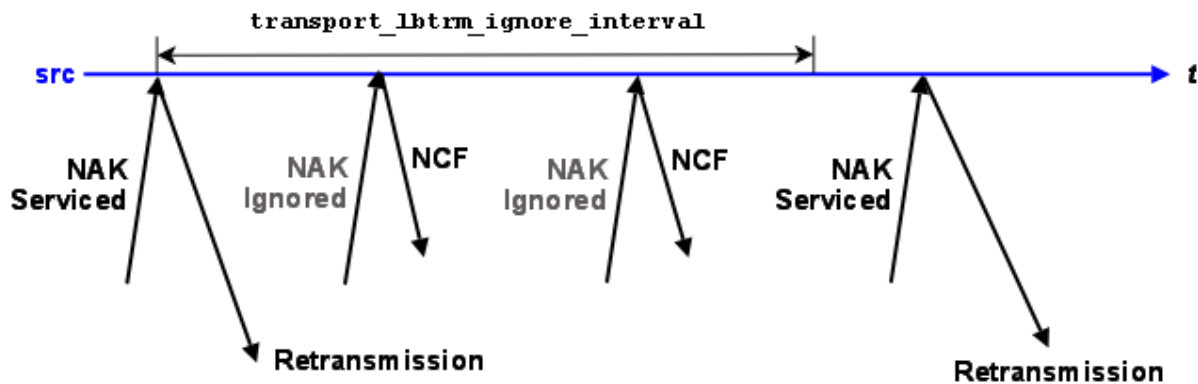
Set `transport_lbtrm_nak_backoff_interval` to the NAK service time that could be reasonably expected from the receiver's location in the network plus some cushion for network congestion. Set `transport_lbtrm_nak_generation_interval` to the latency budget established for the transport layer. See our whitepaper [Topics in High Performance Messaging](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www) ([http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM\\_Topics\\_in\\_High\\_Performance\\_WP\\_www](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www)) for background on latency budgets. See also [Reducing Loss Recovery Latencies](https://communities.informatica.com/infakb/faq/5/Pages/80070.aspx) (<https://communities.informatica.com/infakb/faq/5/Pages/80070.aspx>)

**Note:** Figure 4-5 depicts loss occurring over a LBT-RM transport session. Many topics may be sent across a given transport session. For information about how topic level loss is reported, see [Delivery Control Options](http://Config/deliverycontroloptions.html) ([../Config/deliverycontroloptions.html](http://Config/deliverycontroloptions.html)).

### 4.9.2. LBT-RM Source Ignoring NAKs for Efficiency

Bandwidth efficiency of an LBT-RM source may be improved by avoiding useless retransmissions. Consider the case of an LBT-RM source that has received a NAK for a datagram that it has just retransmitted. It's likely that the NAK and the retransmission "crossed in the mail." Hence it's likely that the receiver generating the NAK will have already received the retransmission just sent. If so, there's no need for the source to send another retransmission so the NAK can be safely ignored. Consider the timeline illustrated in Figure 4-6.

**Figure 4-6. Scenario Timeline: LBT-RM Source Ignoring NAKs for Efficiency**

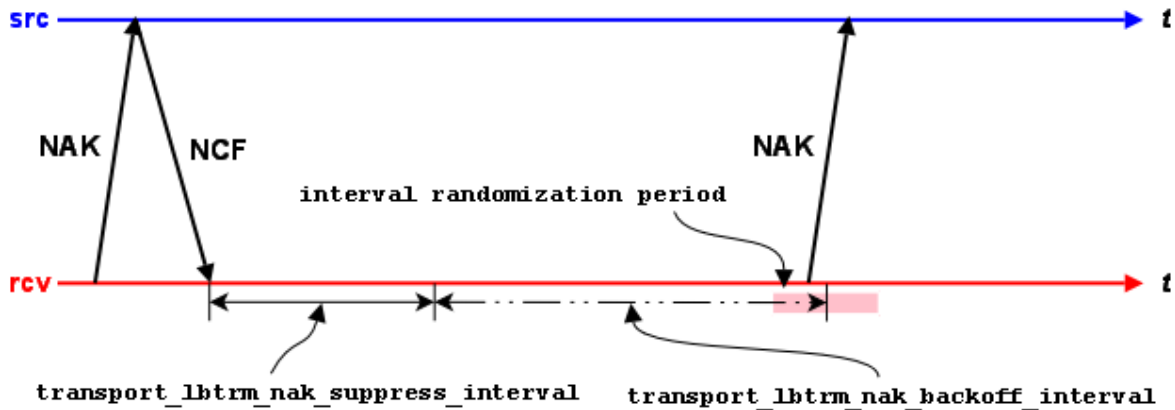


This shows NAKs for a given datagram being ignored for *transport\_lbtrm\_ignore\_interval* following the retransmission of that datagram. (The successive NAKs received by the source in Figure 4-6 indicate that more than one receiver is subscribed to the source's topic.) NAKs will be serviced as normal following the passage of the interval. When ignoring a NAK, the source sends a NCF (NAK ConFirmation) instead of a retransmission, which starts a NAK suppression interval at the receiver. (See Figure 4-7.)

### 4.9.3. LBT-RM Receiver Suppressing NAK Generation

LBT-RM sources want receivers to be notified that their NAKs have been heard. Prompt notification via a retransmission or NCF can suppress useless NAK generation. There are a variety of circumstances where the source does not send a retransmission in response to a receiver's NAK. For example, as shown in Figure 4-6, NAKs received during the ignore interval do not generate retransmissions. Another example would be if previous retransmissions have used up all the retransmission bandwidth for the current rate limiter interval. See Figure 4-7 for a depiction of how a receiver responds to a NCF.

Figure 4-7. Scenario Timeline: An LBT-RM Receiver Suppressing NAK Generation



Following the receipt of an NCF, a receiver suppresses all NAK generation until `transport_lbtrm_nak_suppress_interval` passes. NAK generation resumes with the usual `transport_lbtrm_nak_backoff_interval` if repair was not received during the suppression interval.

**Note:** The actual length of the interval randomization period is between 1/2 and 3/2 of the configured interval value. In Figure 4-7, this period appears shorter to simplify the diagram.

#### 4.9.4. `transport_lbtrm_ignore_interval` (source)

The interval to ignore NAKs after a retransmission is sent. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.9.5. `transport_lbtrm_nak_backoff_interval` (receiver)

For LBT-RM sessions only. The maximum interval between transmissions of a single NAK. The actual time the receiver will wait to NAK again is random. The algorithm used to determine the time range is  $(1/2 * \text{backoff\_interval})$

-  $3/2 * \text{backoff\_interval}$ ). This can result in a wait interval longer than the specified value. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions \(../Design/lbm-objects.html#RECEIVER-OBJECT\)](#) and [Interrelated Configuration Options \(../Config/grouprelated.html\)](#) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.9.6. transport\_lbtrm\_nak\_generation\_interval (receiver)

For LBT-RM sessions only. The maximum time that a piece of data may be outstanding before the data is unrecoverably lost. Although the minimum valid value is 5 milliseconds, larger values are advisable. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions \(../Design/lbm-objects.html#RECEIVER-OBJECT\)](#) and [Interrelated Configuration Options \(../Config/grouprelated.html\)](#) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.9.7. transport\_lbtrm\_nak\_initial\_backoff\_interval (receiver)

For LBT-RM sessions only. The interval between loss detection and transmission of the first NAK. The actual time the receiver will wait to NAK is random. The algorithm used to determine the time range is  $(1/2 * \text{initial\_backoff\_interval} - 3/2 * \text{initial\_backoff\_interval})$ . This can result in a wait interval longer than the specified value. A value of 0 indicates that the receiver should immediately send a NAK. Users should be fully aware of the implications of this before using a value of 0.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 50 (0.05 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

### 4.9.8. transport\_lbtrm\_nak\_suppress\_interval (receiver)

For LBT-RM sessions only. The maximum interval to suppress sending NAKs after an NCF or a NAK from another receiver. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to Receiver Configuration and Transport Sessions (../Design/lbm-objects.html#RECEIVER-OBJECT) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.

### 4.9.9. transport\_lbtrm\_receiver\_socket\_buffer (context)

Value used to set `SO_RCVBUF` value of the LBT-RM receiver multicast socket. In some cases the OS will not allow all of this value to be used. See the section on socket buffer sizes for platform-dependent information. See also our white paper Topics in High Performance Messaging ([http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM\\_Topics\\_in\\_High\\_Performance\\_WP\\_www](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www)) for background and guidelines on UDP buffer sizing.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 524288 (512KB)  
*When to Set:* Can only be set during object initialization.

### 4.9.10. transport\_lbtrm\_send\_naks (receiver)

For LBT-RM sessions only. This flag indicates whether LBT-RM should send negative acknowledgements (NAKs) for missing packets or not. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to Receiver Configuration and Transport Sessions (../Design/lbm-objects.html#RECEIVER-OBJECT) for additional information.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
-------	-------------



Value	Description
1	NAKs are sent for missing packets to request retransmission. <b>Default for all.</b>
0	Do not send NAKs for missing packets.

#### 4.9.11. transport\_lbtrm\_source\_socket\_buffer (context)

Value used to set `SO_SNDBUF` value of the LBT-RM send multicast socket. In some cases the OS will not allow all of this value to be used. See the section on socket buffer sizes for platform-dependent information. A value of 0 instructs **UM** to use the OS default.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 0 (use OS default or 131072, whichever is larger)  
*When to Set:* Can only be set during object initialization.

#### 4.9.12. transport\_lbtrm\_transmission\_window\_limit (source)

Caps the total amount of memory that a transmission window uses, which includes data and overhead. For example, if the `transport_lbtrm_transmission_window_size` is 24 MB (default) and the source sends 20 byte messages with the "flush" flag, the actual amount of memory used can approximate 300 MB. The default value of this option does not limit the transmission window.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.

#### 4.9.13. transport\_lbtrm\_transmission\_window\_size (source)

The maximum amount of buffered data that the LBT-RM source is allowed to retain for retransmissions. The minimum valid value is 65,536 bytes. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes

*Default value:* 25165824 (24 MB)

*When to Set:* Can only be set during object initialization.

## 4.10. Transport LBT-RM Operation Options

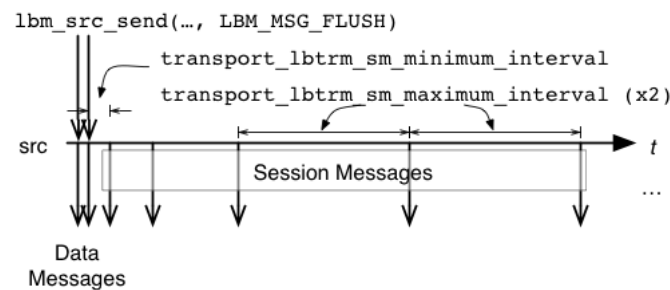
### Scenario Timeline: LBT-RM Source Stops Sending

Reliable multicast protocols like LBT-RM rely on sequence numbers and the arrival of data after a loss as evidence that the loss happened. What would happen if the last packet sent by a source was lost? How would receivers learn of the loss if no further messages were sent?

LBT-RM generates session messages when the sources on a transport session stop sending. These messages contain the expected last sequence number for the session so that receivers can detect loss even when sources aren't sending. Session messages also help to maintain state in multicast routers and switches that require regular traffic to prevent the reclamation of unused forwarding entries.

The following timeline illustrates the case where an LBT-RM source stops sending.

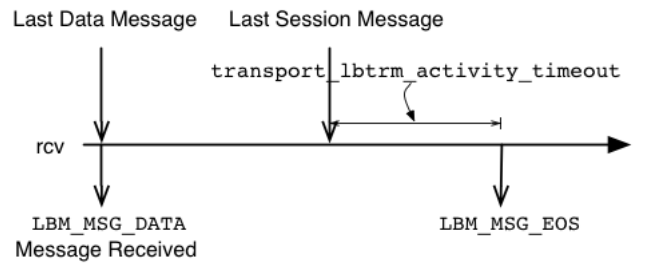
**Figure 4-8. An LBT-RM source stops sending**



No session messages are generated as long as the interval between `lbm_src_send()` calls that generate writes to LBT-RM is less than `transport_lbtrm_sm_minimum_interval`. The interval between session messages starts at `transport_lbtrm_sm_minimum_interval` and doubles till it reaches `transport_lbtrm_sm_maximum_interval`.

### Scenario Timeline: Receiver Detects End of LBT-RM Session

The absence of activity on a transport session is the only indication receivers get that a source is gone or no longer available through any network path. LBT-RM receivers reset a session activity timer for each data message or session message that arrives. If the activity timer ever expires, all receivers on the transport session receive an `LBM_MSG_EOS` event. This is illustrated in the following timeline:

**Figure 4-9. A receiver detects the end of an LBT-RM session**

The activity timer is controlled with the `transport_lbtrm_activity_timeout` option.

#### 4.10.1. `transport_lbtrm_activity_timeout` (receiver)

For LBT-RM sessions only. The maximum time that an LBT-RM session may be quiescent before it is deleted and an EOS event is delivered for all topics using this transport session. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#RECEIVER-OBJECT](#)) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 60000 (60 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.10.2. `transport_lbtrm_coalesce_threshold` (source)

The threshold of the maximum number of individual messages that are copied into a single buffer before being sent to the LBT-RM source internals. When used with small messages, this allows LBT-RM to use less memory at the expense of an additional copy operation of the data before being sent. The default value is also the maximum allowable value for Solaris and AIX. For Linux and Microsoft Windows, the maximum allowable value is 1023. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* int  
*Units:* number of individual messages  
*Default value:* 15  
*When to Set:* Can only be set during object initialization.

### 4.10.3. transport\_lbtrm\_data\_rate\_limit (context)

Maximum aggregate transmission rate of all LBT-RM sessions' original data plus retransmissions for this particular context.

<i>Scope:</i>	context
<i>Type:</i>	unsigned long int
<i>Units:</i>	bits per second
<i>Default value:</i>	10000000 (10 Mbps)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.10.4. transport\_lbtrm\_datagram\_max\_size (context)

The maximum datagram size that can be generated for a LBT-RM transport session. The default value is 8192, the minimum is 500 bytes, and the maximum is 65535.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint_t
<i>Units:</i>	bytes
<i>Default value:</i>	8192
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1

### 4.10.5. transport\_lbtrm\_preactivity\_timeout (receiver)

Use this option only if the receiver subscribes to a pre-LBM 3.3 source or if you have turned off TSNI messages in your post-LBM 3.3 implementation. Set it high enough so the source starts sending data messages before the timeout expires. This timeout begins when the receiver receives a topic advertisement. Pre-LBM 3.3 sources do not send TSNI messages which in effect inform receivers that the source is alive even though it has not started sending data. Session messages provide the same information but do not begin until after the source has started sending data. This option provides an additional activity timeout for the receiver that does not rely on TSNI or sessions messages. The default value of 0 (zero) essentially disables this option, giving precedence to the receiver's standard activity timeout. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to Receiver Configuration and Transport Sessions ([../Design/lbm-objects.html#RECEIVER-OBJECT](#)) for additional information.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	0 (zero)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.4.1/UME 2.1.1.

### 4.10.6. transport\_lbtrm\_rate\_interval (context)

Period that LBT-RM rate limiter runs. Reducing period reduces burst intensity, but also increases CPU load.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 100  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
"10" (Integer value as a string.)	10	LBT-RM rate limiter runs every 10 milliseconds.
"20" (Integer value as a string.)	20	LBT-RM rate limiter runs every 20 milliseconds.
"50" (Integer value as a string.)	50	LBT-RM rate limiter runs every 50 milliseconds.
"100" (Integer value as a string.)	100	LBT-RM rate limiter runs every 100 milliseconds. <b>Default for all.</b>

### 4.10.7. transport\_lbtrm\_retransmit\_rate\_limit (context)

Maximum aggregate transmission rate of all LBT-RM sessions' retransmissions for this particular context. This should always be less than the value used for original data.

*Scope:* context  
*Type:* unsigned long int  
*Units:* bits per second  
*Default value:* 5000000 (5 Mbps)  
*When to Set:* Can only be set during object initialization.

### 4.10.8. transport\_lbtrm\_sm\_maximum\_interval (source)

The maximum interval between LBT-RM session messages. In lieu of data being sent, LBT-RM sends session

messages to inform receivers of sequence numbers and to let receivers know that the sender is still transmitting. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.10.9. transport\_lbtrm\_sm\_minimum\_interval (source)

The minimum interval between LBT-RM session messages. In lieu of data being sent, LBT-RM sends session messages to inform receivers of sequence numbers and to let receivers know that the sender is still transmitting. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.10.10. transport\_lbtrm\_tgsz (source)

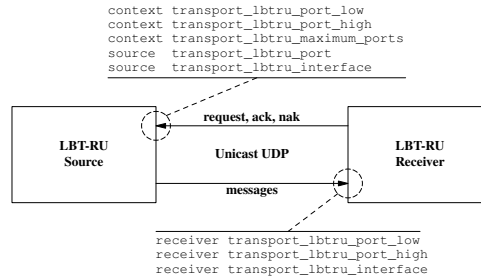
The transmission group size used for this Topic when LBT-RM is used. This value must be greater than 0 and must be a power of 2 no greater than 32K. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Units:* packets  
*Default value:* 8  
*When to Set:* Can only be set during object initialization.

## 4.11. Transport LBT-RU Network Options

LBT-RU receivers initiate UDP connections toward LBT-RU sources for delivering NAKs and ACKs. LBT-RU sources then initiate connections toward LBT-RU receivers for delivery of UMS messages.

**Figure 4-10. LBT-RU network options**



`transport_lbtru_port_low` is the lowest port that UMS will allocate for LBT-RU sources in a context; `transport_lbtru_port_high` is the highest. No more than `transport_lbtru_maximum_ports` ports will be assigned to LBT-RU sources within a single context.

Creation of an UMS source on an LBT-RU transport will allocate an unused UDP port from the range if less than `transport_lbtru_maximum_ports` ports have already been allocated. Setting `transport_lbtru_maximum_ports` to a fraction of the range allows the corresponding multiple number of UMS processes to share a common configuration.

If a particular UDP port is desired by a source, it may be given with `transport_lbtru_port`. If the desired port is already in use, then an unused port will be sought as described above. A value of 0 (the default) expresses no preference and results in the default open port seeking behavior described above.

`transport_lbtru_interface` may be used on LBT-RU sources to choose particular interface, overriding the default `INADDR_ANY` which accepts connections on all interfaces. Similarly, `transport_lbtru_interface` may be used on receivers to choose a particular interface for outgoing connections.

### 4.11.1. transport\_lbtru\_interface (receiver)

Specifies the network interface over which UM LBT-RU receivers read application data messages. Can specify full IP address of interface, or just network part (see Specifying Interfaces for details).

**Scope:** receiver  
**Type:** lbm\_ipv4\_address\_mask\_t  
**Default value:** 0.0.0.0 (INADDR\_ANY)  
**When to Set:** Can only be set during object initialization.

### 4.11.2. transport\_lbtru\_interface (source)

Specifies the network interface over which UM LBT-RU sources receive connection requests from topic receivers. Can specify full IP address of interface, or just network part (see Specifying Interfaces for details). Be aware that this option is applied to the transport session when the first topic is created on that session. Thus, setting a different interface for a subsequent topic that maps onto the same transport session will have no effect. Default is set to INADDR\_ANY, meaning that it will accept incoming connection requests from any interface.

*Scope:* source  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.11.3. transport\_lbtru\_maximum\_ports (context)

Maximum number of unicast port numbers that LBT-RU will use for all implicitly allocated sessions.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Units:* number of ports  
*Default value:* 5  
*When to Set:* Can only be set during object initialization.

### 4.11.4. transport\_lbtru\_port (source)

The preferred unicast port number for this Topic. If 0, the context will attempt to find one in the given LBT-RU source port range.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Default value:* 0 (use open port)  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

### 4.11.5. transport\_lbtru\_port\_high (context)

High unicast port number to assign LBT-RU sources to. Clients will send connection requests, ACKs, and NAKs to a port number in this range.

*Scope:* context  
*Type:* lbm\_uint16\_t



*Default value:* 14389  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

#### 4.11.6. transport\_lbtru\_port\_high (receiver)

High port number to use for receiving LBT-RU data. All LBT-RU data for the topic will arrive on this range.

*Scope:* receiver  
*Type:* lbm\_uint16\_t  
*Default value:* 14379  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

#### 4.11.7. transport\_lbtru\_port\_low (context)

Low unicast port number to assign LBT-RU sources to. Clients will send connection requests, ACKs, and NAKs to a port number in this range.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14380  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

#### 4.11.8. transport\_lbtru\_port\_low (receiver)

Low port number to use for receiving LBT-RU data. All LBT-RU data for the topic will arrive on this range.

*Scope:* receiver  
*Type:* lbm\_uint16\_t  
*Default value:* 14360  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

## 4.12. Transport LBT-RU Reliability Options

For every LBT-RU reliability option, there is a corresponding LBT-RM reliability option. For more information on how LBT-RU reliability options interact and for illustrations, please see the introduction to the transport LBT-RM reliability options section.

### 4.12.1. transport\_lbtru\_ignore\_interval (source)

The interval to ignore NAKs after a retransmission is sent. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.12.2. transport\_lbtru\_nak\_backoff\_interval (receiver)

For LBT-RU sessions only. The maximum interval between transmissions of a single NAK. The actual value is random (to reduce self-similar behaviors) and is uniform on the range  $[0.5 * \text{interval}, 1.5 * \text{interval}]$ . This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#RECEIVER-OBJECT](#)) and [Interrelated Configuration Options](#) ([../Config/grouprelated.html](#)) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200 [100,300] (0.2 [0.1,0.3] seconds)  
*When to Set:* Can only be set during object initialization.

### 4.12.3. transport\_lbtru\_nak\_generation\_interval (receiver)

For LBT-RU sessions only. The maximum time that a piece of data may be outstanding before the data is unrecoverably lost. Although the minimum valid value is 5 milliseconds, larger values are advisable. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#RECEIVER-OBJECT](#)) and [Interrelated Configuration Options](#) ([../Config/grouprelated.html](#)) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.12.4. transport\_lbtru\_nak\_suppress\_interval (receiver)

For LBT-RU sessions only. The maximum interval to suppress sending NAKs after an NCF is received. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to Receiver Configuration and Transport Sessions (../Design/lbm-objects.html#RECEIVER-OBJECT) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.

#### 4.12.5. transport\_lbtru\_receiver\_socket\_buffer (context)

Value used to set `SO_RCVBUF` value of the LBT-RU receiver unicast socket (both sender and receiver sides). In some cases the OS will not allow all of this value to be used. See the section on socket buffer sizes for platform-dependent information. See also our white paper Topics in High Performance Messaging ([http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM\\_Topics\\_in\\_High\\_Performance\\_WP\\_www](http://vip.informatica.com/content/Downloads?docid=1568&=NA-Ongoing-2011Q1-JP-UM_Topics_in_High_Performance_WP_www)) for background and guidelines on UDP buffer sizing.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 524288 (512KB)  
*When to Set:* Can only be set during object initialization.

#### 4.12.6. transport\_lbtru\_source\_socket\_buffer (context)

Value used to set `SO_SNDBUF` value of the LBT-RU send multicast socket. In some cases the OS will not allow all of this value to be used. See the section on socket buffer sizes for platform-dependent information. A value of 0 instructs **UM** to use the OS default.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 0 (use OS default or 131072, whichever is larger)  
*When to Set:* Can only be set during object initialization.

#### 4.12.7. transport\_lbtru\_transmission\_window\_limit (source)

Caps the total amount of memory that a transmission window uses, which includes data and overhead. For example, if the `transport_lbtru_transmission_window_size` is 24 MB (default) and the source sends 20 byte messages with the "flush" flag, the actual amount of memory used can approximate 300 MB. The default value of this option does not limit the transmission window.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.

#### 4.12.8. transport\_lbtru\_transmission\_window\_size (source)

The maximum amount of buffered data that the LBT-RU source is allowed to retain for retransmissions. The minimum valid value is 65536 bytes. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 25165824 (24 MB)  
*When to Set:* Can only be set during object initialization.

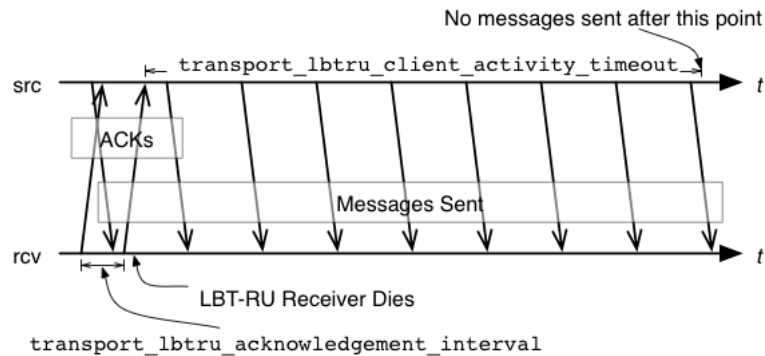
### 4.13. Transport LBT-RU Operation Options

For most LBT-RU operation options, there is a corresponding LBT-RM operation option. For more information on how LBT-RU operation options interact and for illustrations, please see the introduction to the transport LBT-RM operation options section.

Two options unique to LBT-RU are `transport_lbtru_client_map_size` and `transport_lbtru_connect_interval`.

The illustration below shows the interaction of two more options unique to LBT-RU: `transport_lbtru_acknowledgement_interval` and `transport_lbtru_client_activity_timeout`.

**Figure 4-11. An LBT-RU receiver goes away**



### 4.13.1. `transport_lbtru_acknowledgement_interval` (receiver)

For LBT-RU session only. The interval between sending acknowledgements. Each client continually sends acknowledgements to let the source know that the client is still alive. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#RECEIVER-OBJECT](#)) for additional information.

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.13.2. `transport_lbtru_activity_timeout` (receiver)

For LBT-RU sessions only. The maximum time that an LBT-RU session may be quiescent before it is deleted and an EOS event is delivered for all topics using this transport session. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions](#) ([../Design/lbm-objects.html#RECEIVER-OBJECT](#)) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 60000 (60 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.13.3. transport\_lbtru\_client\_activity\_timeout (source)

The maximum time that an LBT-RU client may be quiescent, i.e. not sending ACKs, before the sender assumes that it is dead and stops sending to it. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.13.4. transport\_lbtru\_client\_map\_size (source)

The size of the hash table used to store client information and state. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* size\_t  
*Units:* table entries  
*Default value:* 7  
*When to Set:* Can only be set during object initialization.

### 4.13.5. transport\_lbtru\_coalesce\_threshold (source)

The threshold of the maximum number of individual messages that are copied into a single buffer before being sent to the LBT-RU source internals. When used with small messages, this allows LBT-RU to use less memory at the expense of an additional copy operation of the data before being sent. The default value is also the maximum allowable value for Solaris and AIX. For Linux and Microsoft Windows, the maximum allowable value is 1023. This option affects the transport session underlying the source rather than the source itself. The transport session uses the

value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* int  
*Units:* number of messages  
*Default value:* 15  
*When to Set:* Can only be set during object initialization.

#### 4.13.6. transport\_lbtru\_connect\_interval (receiver)

For LBT-RU session only. The interval between sending connection requests. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions \(../Design/lbm-objects.html#RECEIVER-OBJECT\)](#) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 100 (0.1 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.13.7. transport\_lbtru\_data\_rate\_limit (context)

Maximum aggregate transmission rate of all LBT-RU sessions original data for this particular context.

*Scope:* context  
*Type:* unsigned long int  
*Units:* bits per second  
*Default value:* 10000000 (10 Mbps)  
*When to Set:* Can only be set during object initialization.

#### 4.13.8. transport\_lbtru\_datagram\_max\_size (context)

The maximum datagram size that can be generated for a LBT-RU transport session. The default value is 8192, the minimum is 500 bytes, and the maximum is 65535.

*Scope:* context  
*Type:* lbm\_uint\_t  
*Units:* bytes

*Default value:* 8192  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1

#### 4.13.9. transport\_lbtru\_maximum\_connect\_attempts (receiver)

The maximum number of connect attempts to make before this transport session is deleted and an EOS event is delivered for all topics using this transport session. This option affects the transport session underlying the receiver rather than the receiver itself. The transport session uses the value from the first receiver created on the session and ignores subsequent receivers. Refer to [Receiver Configuration and Transport Sessions](#) ([./Design/lbm-objects.html#RECEIVER-OBJECT](#)) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Default value:* 600  
*When to Set:* Can only be set during object initialization.

#### 4.13.10. transport\_lbtru\_rate\_interval (context)

Period that LBT-RU rate limiter runs. Reducing period reduces burst intensity, but also increases CPU load.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 100  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
"10" (Integer value as a string.)	10	LBT-RU rate limiter runs every 10 milliseconds.
"20" (Integer value as a string.)	20	LBT-RU rate limiter runs every 20 milliseconds.
"50" (Integer value as a string.)	50	LBT-RU rate limiter runs every 50 milliseconds.



String value	Integer value	Description
"100" (Integer value as a string.)	100	LBT-RU rate limiter runs every 100 milliseconds. <b>Default for all.</b>

#### 4.13.11. transport\_lbtru\_retransmit\_rate\_limit (context)

Maximum aggregate transmission rate of all LBT-RU sessions retransmissions for this particular context. This should always be less than the value used for original data.

*Scope:* context  
*Type:* unsigned long int  
*Units:* bits per second  
*Default value:* 5000000 (5 Mbps)  
*When to Set:* Can only be set during object initialization.

#### 4.13.12. transport\_lbtru\_sm\_maximum\_interval (source)

The maximum interval between LBT-RU session messages. In lieu of data being sent, LBT-RU sends session messages to each client to inform them of sequence numbers and to let receivers know that the sender is still transmitting. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.13.13. transport\_lbtru\_sm\_minimum\_interval (source)

The minimum interval between LBT-RU session messages. In lieu of data being sent, LBT-RU sends session messages to each client to inform them of sequence numbers and to let receivers know that the sender is still transmitting. This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to [Source Configuration and Transport Sessions \(../Design/lbm-objects.html#SOURCE-OBJECT\)](#) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t

*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.13.14. transport\_lbtru\_use\_session\_id (source)

Flag to indicate whether the application desires LBT-RU to use a session ID or not. Older versions of **UM** may not understand session IDs with LBT-RU and may not be able to receive LBT-RU transport sessions that include session IDs.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Indicates the application desires LBT-RU to use a session ID. <b>Default for all.</b>
0	Indicates the application does not desire LBT-RU to use a session ID.

### 4.14. Transport LBT-IPC Operation Options

The following option descriptions and diagrams describe the **Ultra Messaging** Configuration Options available for the LBT-IPC transport.

**Figure 4-12. An LBT-IPC source goes away**

The Source Session Message mechanism enables the receiver to detect when a source goes away and works similarly to LBT-RU. It operates independently of message writes and reads in the Shared Memory Area.

#### 4.14.1. transport\_lbtipc\_activity\_timeout (receiver)

The maximum period of inactivity (lack of session messages) from an IPC source before the **UM** delivers an EOS event for all topics using the transport session. Refer to Receiver Configuration and Transport Sessions (../Design/lbm-objects.html#RECEIVER-OBJECT) and Interrelated Configuration Options (../Config/grouprelated.html) for additional information.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 60,000 (60 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

#### 4.14.2. transport\_lbtipc\_behavior (source)

Desired flow control behavior when multiple receivers have joined the same LBT-IPC transport session. See also Transport LBT-IPC ([../Design/lbm-objects.html#TRANSPORT-LBT-IPC](#)). This option affects the transport session underlying the source rather than the source itself. The transport session uses the value from the first source created on the session and ignores subsequent sources. Refer to Source Configuration and Transport Sessions ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* lbm\_ushort\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

String value	Integer value	Description
source_paced	LBM_SRC_TOPIC_ATTR_LBTIPC_BEHAVIOR_SOURCE_PACED	Your application writes as fast as it can to the LBT-IPC shared memory area. Slower receivers can experience loss. A source does not consider if any receivers have successfully read a message before it reclaims it. <b>Default for all.</b>
receiver_paced	LBM_SRC_TOPIC_ATTR_LBTIPC_BEHAVIOR_RECEIVER_PACED	Your application writes to the LBT-IPC shared memory area only as fast as the slowest receiver consumes data. A source will not reclaim a message until all receivers have successfully read the message. This slows down all receiver on the LBT-IPC transport session.

#### 4.14.3. transport\_lbtipc\_datagram\_max\_size (context)

The maximum datagram size that can be generated for a LBT-IPC transport session. The default value is 65535, the minimum is 500 bytes, and the maximum is 65535.

*Scope:* context  
*Type:* lbm\_uint\_t  
*Units:* bytes

*Default value:* 65535  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1

#### 4.14.4. transport\_lbtipc\_id (source)

The preferred Transport ID for a specific source's LBT-IPC session. If 0, the UM context attempts to find one in the given Transport ID range of transport\_lbtipc\_id\_low and transport\_lbtipc\_id\_high. Refer to Source Configuration and Transport Sessions ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Units:* bytes  
*Default value:* 0 (use open port)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

#### 4.14.5. transport\_lbtipc\_id\_high (context)

Highest transport ID of the range of available LBT-IPC Transport IDs.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 20,005  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

#### 4.14.6. transport\_lbtipc\_id\_low (context)

Lowest transport ID of the range of available LBT-IPC Transport IDs.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 20,001  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

#### 4.14.7. transport\_lbtipc\_maximum\_receivers\_per\_transport (source)

The maximum number of receiving contexts that can join an IPC transport session. Once a receiving context joins an IPC transport session, it can receive messages on multiple topics. Increasing this value increases the amount of shared memory allocated per transport session by a negligible amount.

*Scope:* source  
*Type:* lbm\_ushort\_t  
*Default value:* 20  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

#### 4.14.8. transport\_lbtipc\_receiver\_operational\_mode (context)

The mode in which **UM** operates to process LBT-IPC messages. See also [Embedded and Sequential Mode](#) ([../Design/architecture.html#EMBEDDED-MODE](#)) for additional information.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
embedded	LBM_CTX_ATTR_OP_EMBEDDED	<b>UM</b> spawns a thread to process received LBT-IPC messages. <b>Default for all.</b>
sequential	LBM_CTX_ATTR_OP_SEQUENTIAL	Your application must call <code>lbm_context_process_lbtipc_messages()</code> to process received LBT-IPC messages. If you also set the context's <code>operational_mode</code> option to sequential, your application must donate an additional thread to service <code>lbm_context_process_events()</code> calls. You can use sequential mode with the C API, but not with the Java API or .NET API. The Java and .NET APIs do not provide an equivalent <code>lbm_context_process_lbtipc_messages()</code> API for LBT-IPC.

#### 4.14.9. transport\_lbtipc\_receiver\_thread\_behavior (context)

Receiver behavior for monitoring the signaling semaphore set by the IPC source when it writes new data to the shared memory area.

*Scope:* context

*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

String value	Integer value	Description
pend	LBM_CTX_ATTR_IPC_RCV_THREAD_PEND	Receiver waits (sleep) for notification from OS that IPC source has updated the signaling semaphore. This option is best when the IPC source frequently writes new data to the shared area. <b>Default for all.</b>
busy_wait	LBM_CTX_ATTR_IPC_RCV_THREAD_BUSY_WAIT	Provides the lowest latency as the receiver monopolizes the CPU core looking for an incremented semaphore. This option works best for infrequent or sporadic message delivery from the IPC source, but involves a CPU cost.

#### 4.14.10. transport\_lbtipc\_sm\_interval (source)

Time period between sessions message sent from source to receivers. Refer to Source Configuration and Transport Sessions (../Design/lbm-objects.html#SOURCE-OBJECT) and Interrelated Configuration Options (../Config/grouprelated.html) for additional information.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10,000 (10 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

#### 4.14.11. transport\_lbtipc\_transmission\_window\_size (source)

Size of an LBT-IPC transport's shared memory area. This value may vary across platforms. The actual size of the shared memory area equals the value you specify for this option plus about 64 KB for header information. The minimum value for this option is 65,536. Refer to Source Configuration and Transport Sessions (../Design/lbm-objects.html#SOURCE-OBJECT) for additional information.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 25165824 (24 MB)  
*When to Set:* Can only be set during object initialization.

*Version:* This option was implemented in LBM 3.5ea2/UME 2.2ea1

## 4.15. Transport LBT-RDMA Operation Options

Use of the LBT-RDMA transport requires the purchase and installation of the **Ultra Messaging RDMA Transport Module**. See your **Ultra Messaging** representative for licensing specifics.

See also Transport LBT-RDMA ([../Design/lbm-objects.html#TRANSPORT-LBT-RDMA](#)).

### 4.15.1. transport\_lbtrdma\_datagram\_max\_size (context)

The maximum datagram size that can be generated for a LBT-RDMA transport session. The default value is 4096, the minimum is 500 bytes, and the maximum is 4096.

*Scope:* context  
*Type:* lbm\_uint\_t  
*Units:* bytes  
*Default value:* 4096  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1

### 4.15.2. transport\_lbtrdma\_interface (source)

Specifies the network interface over which UM LBT-RDMA sources receive connection requests from topic receivers. You can specify the full IP address of the interface, or just the network part (see Specifying Interfaces for details). Be aware that the first source joining a transport session sets the interface with this option. Thus, setting a different interface for a subsequent topic that maps onto the same transport session will have no effect. Default is set to INADDR\_ANY, meaning that it accepts incoming connection requests from any interface.

*Scope:* source  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1

### 4.15.3. transport\_lbtrdma\_maximum\_ports (context)

Maximum number of LBT-RDMA sessions to allocate.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Units:* number of ports  
*Default value:* 5  
*When to Set:* Can only be set during object initialization.

#### 4.15.4. transport\_lbtrdma\_port (source)

Port number for a specific source's LBT-RDMA session that is outside the transport\_lbtrdma\_port\_low and transport\_lbtrdma\_port\_high range. Refer to Source Configuration and Transport Sessions ([../Design/lbm-objects.html#SOURCE-OBJECT](#)) for additional information.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Default value:* 0 (zero)  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1

#### 4.15.5. transport\_lbtrdma\_port\_high (context)

Highest port number that can be assigned to a LBT-RDMA session.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 20,020  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1

#### 4.15.6. transport\_lbtrdma\_port\_low (context)

Lowest port number that can be assigned to a LBT-RDMA session.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 20,001  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1



### 4.15.7. transport\_lbtrdma\_receiver\_thread\_behavior (context)

Receiver behavior for monitoring a LBT-RDMA source's shared memory area for new data.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1

String value	Integer value	Description
pend	LBM_CTX_ATTR_RDMA_RCV_THREAD_PEND	Receiver waits (sleep) for notification from RDMA that the source has updated the shared memory area with new data. Default. <b>Default for all.</b>
busy_wait	LBM_CTX_ATTR_RDMA_RCV_THREAD_BUSY_WAIT	UDM polls the shared memory area for new data.

### 4.15.8. transport\_lbtrdma\_transmission\_window\_size (source)

Size of an LBT-RDMA transport's shared memory area. This value may vary across platforms. The actual size of the shared memory area equals the value you specify for this option plus about 64 KB for header information. The minimum value for this option is 65,536. Refer to [Source Configuration and Transport Sessions](#) (../Design/lbm-objects.html#SOURCE-OBJECT) for additional information.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 25165824 (24 MB)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1

## 4.16. Transport Acceleration Options

Transport Acceleration Options enable Datagram Bypass Layer (DBL) acceleration in conjunction with DBL-enabled Myricom® (<http://www.myri.com>) 10-Gigabit Ethernet NICs for Linux and Microsoft Windows. DBL is a kernel-bypass technology that accelerates sending and receiving UDP traffic. DBL does not support fragmentation and reassembly, so you must avoid sending messages larger than the MTU configured on the DBL interface.

DBL acceleration is available for the following **Ultra Messaging** transport types.

- LBT-RM (UDP-based reliable multicast)

- LBT-RU (UDP-based reliable unicast)
- Multicast Immediate Messaging
- Multicast Topic Resolution

**Note:** Transport Acceleration is not supported on the HP NonStop® platform.

To enable DBL Transport Acceleration,

1. Install the Myricom 10-Gigabit Ethernet NIC.
2. Install the DBL shared library.
3. Update your search path to include the location of the DBL shared library.
4. Include the appropriate Transport Acceleration Options in your **Ultra Messaging** Configuration File.

**Note:** We recommend setting `transport_datagram_max_size` (`../Config/majoroptions.html#CONTEXTTRANSPORTDATAGRAMMAXSIZE`) to a value at least 28 bytes smaller than the configured MTU.

#### 4.16.1. `dbl_lbtrm_acceleration` (context)

Flag indicating if DBL acceleration is enabled for LBT-RM transports.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

Value	Description
1	DBL acceleration is enabled for LBT-RM.
0	DBL acceleration is not enabled for LBT-RM. <b>Default for all.</b>

#### 4.16.2. `dbl_lbtru_acceleration` (context)

Flag indicating if DBL acceleration is enabled for LBT-RU transports.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

Value	Description
1	DBL acceleration is enabled for LBT-RU.
0	DBL acceleration is not enabled for LBT-RU. <b>Default for all.</b>

### 4.16.3. dbl\_mim\_acceleration (context)

Flag indicating if DBL acceleration is enabled for multicast immediate messaging (MIM).

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

Value	Description
1	DBL acceleration is enabled for MIM.
0	DBL acceleration is not enabled for MIM. <b>Default for all.</b>

### 4.16.4. dbl\_resolver\_acceleration (context)

Flag indicating if DBL acceleration is enabled for topic resolution.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

Value	Description
1	DBL acceleration is enabled for topic resolution.
0	DBL acceleration is not enabled for topic resolution. <b>Default for all.</b>

### 4.16.5. resolver\_ud\_acceleration (context)

Flag indicating if Accelerated Multicast is enabled for Topic Resolution. Accelerated Multicast requires InfiniBand or 10 Gigabit Ethernet hardware and the purchase and installation of the **Ultra Messaging** Accelerated Multicast Module. See your **Ultra Messaging** representative for licensing specifics. UD Acceleration relies on the network hardware supporting loopback which InfiniBand does, but current ConnectX and ConnectX2 10 Gigabit Ethernet

hardware does not. Future 10gbe hardware should support loopback.

- InfiniBand users should enable this option.
- 10 Gigabit Ethernet users should have this option disabled (default) for current generation hardware.

Check your hardware configuration to determine if this option needs to be changed.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 5.2.

Value	Description
1	Accelerated Topic Resolution is enabled.
0	Accelerated Topic Resolution is not enabled. <b>Default for all.</b>

#### 4.16.6. ud\_acceleration (context)

Flag indicating if Accelerated Multicast is enabled for LBT-RM. Accelerated Multicast requires InfiniBand or 10 Gigabit Ethernet hardware and the purchase and installation of the **Ultra Messaging** Accelerated Multicast Module. See your **Ultra Messaging** representative for licensing specifics.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1.

Value	Description
1	Accelerated Multicast is enabled.
0	Accelerated Multicast is not enabled. <b>Default for all.</b>

### 4.17. Multicast Immediate Messaging Network Options

The multicast address and port used for incoming and outgoing multicast immediate messages can be set with *mim\_address* and *mim\_destination\_port*. A context may use different multicast addresses and/or ports for incoming and outgoing messages by setting *mim\_incoming\_address*, *mim\_outgoing\_address*, *mim\_incoming\_destination\_port*, and/or *mim\_outgoing\_destination\_port*. In case of conflict, **UM** uses the most recently set option.

As with LBT-RM on multi-homed hosts, the interface **UM** uses for MIM follows the interface used for multicast topic resolution. See *resolver\_multicast\_interface*.

**Warning** The addresses and ports you configure for MIM traffic should not overlap with any addresses or ports - or address and port ranges - configured for LBT-RM transports or Topic Resolution traffic. For example, do not use the same multicast address for both Topic Resolution (*resolver\_multicast\_address*) and MIM (*mim\_address*). Use different addresses and ports for all multicast address options and port options.

See also Multicast Immediate Messaging ([../Design/lbm-features.html#MULTICAST-IMMEDIATE-MESSAGING](http://Design/lbm-features.html#MULTICAST-IMMEDIATE-MESSAGING)) for more information about this feature.

### 4.17.1. *mim\_address* (context)

The IP multicast address that multicast immediate messages are sent to and received from.

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.10.10.21  
*When to Set:* Can only be set during object initialization.

### 4.17.2. *mim\_destination\_port* (context)

The UDP destination port that multicast immediate messages are sent to and received from.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14401  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

### 4.17.3. *mim\_incoming\_address* (context)

The IP multicast address that multicast immediate messages are received from. Setting this option to 0.0.0.0 turns off multicast immediate messaging (MIM).

*Scope:* context  
*Type:* struct in\_addr  
*Default value:* 224.10.10.21  
*When to Set:* Can only be set during object initialization.

#### 4.17.4. `mim_incoming_destination_port` (context)

The UDP destination port that multicast immediate messages are received from.

<i>Scope:</i>	context
<i>Type:</i>	<code>lbm_uint16_t</code>
<i>Default value:</i>	14401
<i>Byte order:</i>	Network
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.17.5. `mim_outgoing_address` (context)

The IP multicast address that multicast immediate messages are sent to.

<i>Scope:</i>	context
<i>Type:</i>	<code>struct in_addr</code>
<i>Default value:</i>	224.10.10.21
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.17.6. `mim_outgoing_destination_port` (context)

The UDP destination port that multicast immediate messages are sent to.

<i>Scope:</i>	context
<i>Type:</i>	<code>lbm_uint16_t</code>
<i>Default value:</i>	14401
<i>Byte order:</i>	Network
<i>When to Set:</i>	Can only be set during object initialization.

### 4.18. Multicast Immediate Messaging Reliability Options

For every MIM reliability option, there is a corresponding LBT-RM reliability option. For more information on how MIM reliability options interact and for illustrations, please see the introduction to the transport LBT-RM reliability options section.

See also Multicast Immediate Messaging ([../Design/lbm-features.html#MULTICAST-IMMEDIATE-MESSAGING](#)) for more information about this feature.

### 4.18.1. mim\_ignore\_interval (context)

For multicast immediate message senders only. See `transport_lbtrm_ignore_interval` for description.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	500 (0.5 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.18.2. mim\_nak\_backoff\_interval (context)

For multicast immediate message receivers only. See `transport_lbtrm_nak_backoff_interval` for description.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	200 (0.2 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.18.3. mim\_nak\_generation\_interval (context)

For multicast immediate message receivers only. See `transport_lbtrm_nak_generation_interval` for description.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	10000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.18.4. mim\_nak\_initial\_backoff\_interval (context)

For multicast immediate message receivers only. See `transport_lbtrm_nak_initial_backoff_interval` for description.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	50 (0.05 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.4/UME 2.1.

### 4.18.5. mim\_nak\_suppress\_interval (context)

For multicast immediate message receivers only. See `transport_lbtrm_nak_suppress_interval` for description.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.

### 4.18.6. mim\_send\_naks (context)

For multicast immediate message receivers only. See `transport_lbtrm_send_naks` for description.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	NAKs are sent for missing packets to request retransmission. <b>Default for all.</b>
0	Do not send NAKs for missing packets.

### 4.18.7. mim\_transmission\_window\_limit (context)

For multicast immediate message senders only. See `transport_lbtrm_transmission_window_limit` for description.

*Scope:* context  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.

### 4.18.8. mim\_transmission\_window\_size (context)

For multicast immediate message senders only. See `transport_lbtrm_transmission_window_size` for description.

*Scope:* context  
*Type:* size\_t  
*Units:* bytes



*Default value:* 25165824 (24 MB)  
*When to Set:* Can only be set during object initialization.

## 4.19. Multicast Immediate Messaging Operation Options

For many MIM operation options, there is a corresponding LBT-RM operation option. For more information on how MIM operation options interact and for illustrations, please see the introduction to the transport LBT-RM operation options section.

Note that the LBT-RM rate controller also governs MIM transmission rates. Hence there is no separate option for setting MIM transmission rate.

See also Multicast Immediate Messaging ([../Design/lbm-features.html#MULTICAST-IMMEDIATE-MESSAGING](http://Design/lbm-features.html#MULTICAST-IMMEDIATE-MESSAGING)) for more information about this feature.

### 4.19.1. immediate\_message\_receiver\_function (context)

Callback function (and associated event queue and client data pointer) called when a topic-less immediate message is received for which there is no receiver. A value of NULL for the callback prevents the callback from being called.

*Scope:* context  
*Type:* lbm\_context\_rcv\_immediate\_msgs\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.19.2. immediate\_message\_topic\_receiver\_function (context)

Callback function (and associated event queue and client data pointer) that is called when an immediate message is received for a topic for which there is no receiver. A value of NULL for the callback prevents the callback from being called.

*Scope:* context  
*Type:* lbm\_context\_rcv\_immediate\_msgs\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.19.3. mim\_activity\_timeout (context)

For multicast immediate message receivers only. See `transport_lbtrm_activity_timeout` for description. However, multicast immediate message channels do not deliver an EOS indication.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 60000 (60 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.19.4. mim\_delivery\_control\_activity\_check\_interval (context)

The interval between activity checks of a Multicast Immediate Messaging delivery controller. Multiple MIM delivery controllers may exist to accommodate multiple messages from a single MIM sender received across more than one UM Gateway. These multiple delivery controllers allow for duplicate message detection.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 5000 (5 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

### 4.19.5. mim\_delivery\_control\_activity\_timeout (context)

The maximum time that a Multicast Immediate Messaging delivery controller may be quiescent before it is deleted. MIM delivery controllers may be created to accommodate multiple messages from a single MIM sender received across more than one UM Gateway. These multiple delivery controllers allow for duplicate message detection.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 60000 (60 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0.

### 4.19.6. mim\_delivery\_control\_order\_tablesz (context)

For multicast immediate messages with ordered delivery, this controls the size of the hash table used to hold data.

*Scope:* context  
*Type:* size\_t  
*Units:* table entries  
*Default value:* 1031  
*When to Set:* Can only be set during object initialization.

#### 4.19.7. mim\_implicit\_batching\_interval (context)

The maximum timeout between when the first message of an implicitly batched immediate message is queued until the batch is sent. A message will not stay in the queue longer than this value before being sent in the worse case.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.19.8. mim\_implicit\_batching\_minimum\_length (context)

The minimum length of an implicitly batched multicast immediate message. When the total length of the implicitly batched messages reaches or exceeds this value, the batch is sent.

*Scope:* context  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 2048 (8192 for Microsoft Windows)  
*When to Set:* Can only be set during object initialization.

#### 4.19.9. mim\_ordered\_delivery (context)

For multicast immediate messages only. Indicates whether or not the MIM source should have its data delivered in order. The default value also guarantees fragmentation and reassembly of large messages. Changing this option from the default value results in large messages being delivered as individual fragments of less than 8K each, requiring the application to reassemble them. See also [Ordered Delivery](#) (../Design/architecture.html#ORDERED-DELIVERY) for more information about large message fragmentation and reassembly.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Indicates the source should have its data delivered in order. <b>Default for all.</b>
0	The source should have its data delivered as soon as possible and may come in out of order.

#### 4.19.10. mim\_sm\_maximum\_interval (context)

For multicast immediate message senders only. See `transport_lbtrm_sm_maximum_interval` for description.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.19.11. mim\_sm\_minimum\_interval (context)

For multicast immediate message senders only. See `transport_lbtrm_sm_minimum_interval` for description.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.19.12. mim\_sqn\_window\_increment (context)

For multicast immediate message receivers only. Determines the increment by which the sequence number window is moved when detecting the receipt of duplicate multicast immediate messages. Must be a multiple of 8 and an even divisor of `mim_sqn_window_size`.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* messages  
*Default value:* 8192  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2.8/UME 3.2.8/UMQ 2.1.8

### 4.19.13. mim\_sqn\_window\_size (context)

For multicast immediate message receivers only. Determines the window size used to detect the receipt of duplicate multicast immediate messages. Must be a multiple of 8.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	messages
<i>Default value:</i>	16384
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.2.8/UME 3.2.8/UMQ 2.1.8

### 4.19.14. mim\_src\_deletion\_timeout (context)

The timeout after a multicast immediate message is sent before the internal source is deleted and cleaned up.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	30000 (30 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.19.15. mim\_tgsz (context)

For multicast immediate message senders only. See transport\_lbtrm\_tgsz for description.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint16_t
<i>Units:</i>	packets
<i>Default value:</i>	8
<i>When to Set:</i>	Can only be set during object initialization.

### 4.19.16. mim\_unrecoverable\_loss\_function (context)

Callback function (and associated client data pointer) that is called when a MIM receiver has unrecoverable loss. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

<i>Scope:</i>	context
---------------	---------

**Type:** `lbm_mim_unrecloss_func_t`  
**Default value:** `NULL`  
**When to Set:** Can only be set during object initialization.  
**Config File:** Cannot be set from an UM configuration file.

## 4.20. Late Join Options

### 4.20.1. Late Join Recovery

#### 4.20.1.1. Overview

Late Join allows sources to save a predefined amount of their messaging traffic for late-joining receivers. Sources set the configuration options that determine whether they use Late Join or not, and receivers set options that determine whether they will participate in Late Join recovery if sources use Late Join.

UMP's persistent store is built on Late Join technology. In the Estimating Recovery Time ([../Config/latejoinoptions.html#ESTIMATINGRECOVERYTIME](#)) discussion below, the terms "Late Join buffers" and "UMP store" are roughly equivalent.

For more, review the Late Join ([../Design/lbm-features.html#USING-LATE-JOIN](#)) section in the Concepts Guide, especially Configuring Late Join for Large Numbers of Messages ([../Design/lbm-features.html#USING-LATE-JOIN-LARGENUMBERS](#)).

#### 4.20.1.2. Estimating Recovery Time

To estimate Late Join recovery time  $R$  in minutes, use the formula:  $R = D / (1 - (txrate / rxrate))$  where:

- $D$  is the downtime (in minutes) across all receivers
- $txrate$  is the average transmission rate of normal messages from sources during recovery (in kmsgs/sec)
- $rxrate$  is the average recovery rate from source-side Late Join buffers during recovery (in kmsgs/sec)

For example, consider the following scenario:

- $D = 10$  minutes
- $txrate = 10k$  messages / second
- $rxrate = 25k$  messages / second

Plugging these values into the formula gives an estimated recovery time in minutes:  $R = 10 / (1 - (10 / 25))$  or 16.67 minutes. You can use this estimated recovery time to set Late Join option `retransmit_request_generation_interval`

([../Config/latejoinoptions.html#RECEIVERRETRANSMITREQUESTGENERATIONINTERVAL](#)). Set it at least as

high as the longest expected recovery time (don't forget to convert to milliseconds). Note that if this interval is too short, you may experience burst loss during recovery.

Note that this formula assumes the following:

- Recovery rate is as linear as possible with use of option `response_tcp_nodelay 1`
- Transmit rate (`txrate`) from \*all\* relevant sources is fairly constant and equal
- Recovery rate (`rxrate`) from Late Join buffers is fairly constant and equal, and should be measured in a live test, if possible. You can adjust the recovery rate with two Late Join configuration options:
  - `retransmit_request_outstanding_maximum`  
(../Config/latejoinoptions.html#RECEIVERRETRANSMITREQUESTOUTSTANDINGMAXIMUM)
  - `retransmit_request_interval` (../Config/latejoinoptions.html#RECEIVERRETRANSMITREQUESTINTERVAL)

### 4.20.2. late\_join (source)

Configure the source to enable both Late Join and Off-Transport Recovery (OTR) (../Design/lbm-features.html#OTR) operation for receivers.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Enable source for Late Join and OTR.
0	Disable source for Late Join and OTR. <b>Default for all.</b>

### 4.20.3. retransmit\_initial\_sequence\_number\_request (receiver)

When a late-joining receiver detects (from the topic advertisement) that a source is enabled for Late Join but has sent no messages, this flag option lets the receiver request an initial sequence number from a source. Sources respond with a TSNI.

*Scope:* receiver  
*Type:* int  
*Default value:* 1  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2.

Value	Description
-------	-------------

Value	Description
1	The receiver requests an initial sequence number from Late Join enabled sources that have not sent any messages. <b>Default for all.</b>
0	The receiver does not request an initial sequence number.

#### 4.20.4. retransmit\_message\_caching\_proximity (receiver)

This option enables receiver caching of new messages during a recovery. The option value determines how close or proximate the current new sequence number must be to the latest retransmitted sequence number for the receiver to start caching. The receiver recovers uncached data later in the recovery process by the retransmit request mechanism. An option value greater than or equal to the default turns on caching of new data immediately. A smaller value means that caching does not begin until recovery has caught up somewhat with the source. A larger value means that caching can begin earlier during recovery. This value has meaning for only receivers using ordered delivery of data. See [Configuring Late Join for Large Numbers of Messages](#)

([../Design/lbm-features.html#USING-LATE-JOIN-LARGENUMBERS](#)) for additional information about this option.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* messages  
*Default value:* 2147483647  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.3.2/UME 2.0.

#### 4.20.5. retransmit\_request\_generation\_interval (receiver)

The maximum interval between when a receiver first sends a retransmission request and when the receiver stops and reports loss on the remaining RXs not received. See [Configuring Late Join for Large Numbers of Messages](#) ([../Design/lbm-features.html#USING-LATE-JOIN-LARGENUMBERS](#)) for additional information about this option.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.20.6. retransmit\_request\_interval (receiver)

The interval between retransmission request messages to the source. See [Configuring Late Join for Large Numbers of Messages](#) ([../Design/lbm-features.html#USING-LATE-JOIN-LARGENUMBERS](#)) for additional information



about this option.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	500 (0.5 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.20.7. retransmit\_request\_maximum (receiver)

The maximum number of messages to request, counting backward from the current latest message, when late-joining a topic. Due to network timing factors, **UM** may transmit an additional message. For example, a value of 5 sends 5 or possibly 6 retransmit messages to the new receiver. (Hence, you cannot request and be guaranteed to receive only 1 last message--you may get 2.) A value of 0 indicates no maximum.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	messages
<i>Default value:</i>	0
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.20.8. retransmit\_request\_outstanding\_maximum (receiver)

The maximum number of messages to request at a single time from a persistent store or a source. A value of 0 indicates no maximum. See [Configuring Late Join for Large Numbers of Messages](#) ([../Design/lbm-features.html#USING-LATE-JOIN-LARGENUMBERS](#)) for additional information about this option.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	messages
<i>Default value:</i>	200
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.20.9. retransmit\_retention\_age\_threshold (source)

Specifies the minimum age of messages in the retained message buffer before **UM** can delete them. **UM** cannot delete any messages younger than this value. For **UMS** Late Joins, this and `retransmit_retention_size_threshold` are the only options that affect the retention buffer size. For **UMP**, these two options combined with `retransmit_retention_size_limit` affect the retention buffer size. **UM** deletes a message when it meets all configured threshold criteria, i.e., the message is older than this option (if set), and the size of the retention buffer exceeds the

`retransmit_retention_size_threshold` (if set). A value of 0 sets the age threshold to be always triggered, in which case deletion is determined by other threshold criteria.

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* seconds  
*Default value:* 0 (threshold always triggered)  
*When to Set:* Can only be set during object initialization.

#### 4.20.10. `retransmit_retention_size_limit` (source)

Sets a maximum limit on the size of the source's retransmit retention buffer when using a **UMP** store. With **UMP**, stability and delivery confirmation events can delay the deletion of retained messages, which can increase the size of the buffer above the `retransmit_retention_size_threshold`. Hence, this option provides a hard size limit. **UM** sets a minimum value for this option of 8K for UDP and 64K for TCP, and issues a log warning if you set a value less than the minimum.

*Scope:* source  
*Type:* `size_t`  
*Units:* bytes  
*Default value:* 25165824 (24 MB)  
*When to Set:* Can only be set during object initialization.

#### 4.20.11. `retransmit_retention_size_threshold` (source)

Specifies the minimum size of the retained message buffer before **UM** can delete messages. The buffer must reach this size before **UM** can delete any messages older than `retransmit_retention_age_threshold`. For **UMP**, these options combined with `retransmit_retention_size_limit` affect the retention buffer size. A value of 0 sets the size threshold to be always triggered, in which case deletion is determined by other threshold criteria.

*Scope:* source  
*Type:* `size_t`  
*Units:* bytes  
*Default value:* 0 (threshold always triggered)  
*When to Set:* Can only be set during object initialization.

#### 4.20.12. `use_late_join` (receiver)

Flag indicating if the receiver should participate in a late join operation or not.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	The receiver will participate in using late join if requested to by the source. <b>Default for all.</b>
0	The receiver will not participate in using late join even if requested to by the source.

## 4.21. Off-Transport Recovery Options

See also Off-Transport Recovery (OTR) ([../Design/lbm-features.html#OTR](#)) for more information about this feature.

### 4.21.1. otr\_request\_duration (receiver)

The length of time a receiver continues to send OTR lost-message requests before giving up.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 20000 (20 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UM 5.2

### 4.21.2. otr\_request\_initial\_delay (receiver)

The length of time a receiver waits before initiating OTR lost-message requests.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UM 5.3

### 4.21.3. otr\_request\_log\_alert\_cooldown (receiver)

Each OTR request generates a log message. The first request's log message is a WARNING-level log message, and subsequent requests that quickly follow generate INFO-level log messages. After a time interval defined by this option, the next request leading a new burst of requests again generates a WARNING-level log message.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	seconds
<i>Default value:</i>	300 (5 minutes)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UM 5.3

### 4.21.4. otr\_request\_maximum\_interval (receiver)

The maximum time interval between a receiver's OTR lost-message requests. After the receiver initiates OTR and is waiting to receive the retransmission, the initial interval (set by `otr_request_minimum_interval`) doubles in length for each request until it reaches this option's value, then continues at this interval (until timeout or **UM** recovers messages). NOTE: When using TCP Request/Response, this value must be shorter than `response_tcp_deletion_timeout`.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	10000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UM 5.3

### 4.21.5. otr\_request\_minimum\_interval (receiver)

The initial time interval between a receiver's OTR lost-message requests. While the receiver is waiting to receive the retransmission, the interval doubles in length for each request until it reaches the maximum interval set by `otr_request_maximum_interval`.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	1000 (1 second)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UM 5.2

### 4.21.6. otr\_request\_outstanding\_maximum (receiver)

The maximum number of OTR lost-message requests outstanding at any given time. Each message specifies an individual lost message. A value of 0 indicates no maximum.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* messages  
*Default value:* 200  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UM 5.2

### 4.21.7. use\_otr (receiver)

Flag indicating if the receiver can use OTR or not.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UM 5.2

Value	Description
1	The receiver is enabled to use OTR to recover lost messages.
0	The receiver is not enabled to use OTR to recover lost messages. <b>Default for all.</b>

## 4.22. Request Network Options

See also Request/Response ([../Design/fundamental-concepts.html#REQUEST-RESPONSE](#)) for more information about this feature.

### 4.22.1. request\_tcp\_bind\_request\_port (context)

Allows you to turn off request port binding. Setting this option to 0 prevents sockets from being bound to the request port. Turning off request port binding also turns off the **UM** features: Request/Response ([../Design/lbm-features.html#REQUEST-RESPONSE-MESSAGES](#)), Late Join ([../Design/lbm-features.html#USING-LATE-JOIN](#)), the reception of Unicast Immediate Messages and **UMP**.

*Scope:* context

*Type:* int  
*Default value:* 1  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.3.7/UME 2.0.5.

Value	Description
1	Set request port binding. <b>Default for all.</b>
0	Turn off request port binding.

### 4.22.2. request\_tcp\_interface (context)

Specifies the network interface over which **UM** accepts TCP connections in response to requests it has sent out. You can specify a full IP address of interface, or just the network part (see Specifying Interfaces for details). Default is set to INADDR\_ANY, meaning that it will not bind to a specific interface. You can also modify the default by setting the option to 0.0.0.0/0 which produces the same result.

*Scope:* context  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.22.3. request\_tcp\_port (context)

Port number used for listening for responses from requests. If 0, use a random open port within the range of [request\_tcp\_port\_low, request\_tcp\_port\_high]. If nonzero, the specific port number is used instead. Each **UM** context will bind to a TCP port for requests when it is initialized.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 0 (use open port)  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.

### 4.22.4. request\_tcp\_port\_high (context)

High port number to use for listening for responses from requests.

*Scope:* context  
*Type:* lbm\_uint16\_t

*Default value:* 14395  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

### 4.22.5. request\_tcp\_port\_low (context)

Low port number to use for listening for responses from requests.

*Scope:* context  
*Type:* lbm\_uint16\_t  
*Default value:* 14391  
*Byte order:* Host  
*When to Set:* Can only be set during object initialization.

## 4.23. Request Operation Options

See also Request/Response ([../Design/fundamental-concepts.html#REQUEST-RESPONSE](#)) for more information about this feature.

### 4.23.1. request\_tcp\_exclusiveaddr (context)

Applicable only to Windows. Indicate whether the TCP accepting socket should set `SO_EXCLUSIVEADDRUSE` or not before it binds. The default setting in Windows allows multiple binds to the same port. By default, **UM** will set `SO_EXCLUSIVEADDRUSE` to minimize port sharing. Refer to Microsoft's web site for more information on `SO_EXCLUSIVEADDRUSE`.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Set <code>SO_EXCLUSIVEADDRUSE</code> . <b>Default for Windows.</b>
0	Do not set <code>SO_EXCLUSIVEADDRUSE</code> .

### 4.23.2. request\_tcp\_listen\_backlog (context)

The backlog used in the TCP `listen()` call to set the queue length for incoming connections.

*Scope:* context  
*Type:* int  
*Default value:* 5  
*When to Set:* Can only be set during object initialization.

### 4.23.3. request\_tcp\_reuseaddr (context)

Whether the TCP accepting socket should set `SO_REUSEADDR` or not before it binds. NOTE: For Microsoft Windows, **UM** always forces this value to "0" regardless of the value set in any configuration files.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Set <code>SO_REUSEADDR</code> .
0	Do not set <code>SO_REUSEADDR</code> . <b>Default for all.</b>

## 4.24. Response Operation Options

See also Request/Response ([../Design/fundamental-concepts.html#REQUEST-RESPONSE](#)) for more information about this feature.

### 4.24.1. response\_session\_maximum\_buffer (context)

Value used to control the maximum amount of data buffered in **UM** for each response session (unicast connection to a requester).

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 65536  
*When to Set:* Can only be set during object initialization.

### 4.24.2. response\_session\_sender\_socket\_buffer (context)

Value used to set the `SO_SNDBUF` value of the response session (unicast connection to a requester). In some cases the



OS will not allow all of this value to be used. A value of 0 instructs **UM** to use the OS defaults. See the section on socket buffer sizes for platform-dependent information.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* bytes  
*Default value:* 0 (use OS defaults)  
*When to Set:* Can only be set during object initialization.

### 4.24.3. response\_tcp\_deletion\_timeout (context)

After **UM** deletes a TCP response, this is the timeout period after which **UM** closes the connection and reclaims its memory. NOTE: When using Off-Transport Recovery, this value must be longer than otr\_request\_maximum\_interval.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 2000 (2 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.24.4. response\_tcp\_interface (context)

Specifies the network interface over which **UM** initiates TCP connections for responses. You can specify the full IP address of interface, or just the network part (see Specifying Interfaces for details). Default is set to INADDR\_ANY, meaning that it will not bind to a specific interface. You can also modify the default by setting the option to 0.0.0.0/0 which produces the same result.

*Scope:* context  
*Type:* lbm\_ipv4\_address\_mask\_t  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.

### 4.24.5. response\_tcp\_nodelay (context)

Whether the TCP sockets used for sending responses should set TCP\_NODELAY or not. (Setting TCP\_NODELAY disables Nagle's algorithm.)

*Scope:* context  
*Type:* int

*When to Set:* Can only be set during object initialization.

Value	Description
1	TCP response sockets should set <code>TCP_NODELAY</code> (disable Nagle).
0	TCP response sockets should not set <code>TCP_NODELAY</code> (leave Nagle enabled). <b>Default for all.</b>

## 4.25. Implicit Batching Options

### 4.25.1. `implicit_batching_interval` (source)

The maximum timeout between when the first message of an implicit batch is queued until the batch is sent. A message will not stay in the queue longer than this value before being sent in the worse case. Refer to `Message Batching` ([../Design/architecture.html#MESSAGE-BATCHING](#)) for additional information.

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* May be set during operation.

### 4.25.2. `implicit_batching_minimum_length` (source)

The minimum length of an implicitly batched message. When the total length of the implicitly batched messages reaches or exceeds this value, the batch is sent. Refer to `Message Batching` ([../Design/architecture.html#MESSAGE-BATCHING](#)) for additional information.

*Scope:* source  
*Type:* `size_t`  
*Units:* bytes  
*Default value:* 2048 (8192 for Microsoft Windows)  
*When to Set:* May be set during operation.

### 4.25.3. `implicit_batching_type` (source)

The implicit batching algorithm to use which controls when messages sent on a transport session are flushed or

batched, if batching is in use.

*Scope:* source

*Type:* int

*When to Set:* May be set during operation.

String value	Integer value	Description
default	LBM_SRC_TOPIC_ATTR_IMPLICIT_BATCH	<b>Implicit batching</b> is controlled entirely by the <code>implicit_batching_minimum_length</code> and <code>implicit_batching_interval</code> options. Refer to Message Batching ( <a href="#">../Design/architecture.html#MESSAGE-BATCHING</a> ) for additional information. <b>Default for all.</b>
adaptive	LBM_SRC_TOPIC_ATTR_IMPLICIT_BATCH	<b>Source-paced batching</b> method that attempts to adjust the amount of messages sent in each batch automatically. The options, <code>implicit_batching_minimum_length</code> and <code>implicit_batching_interval</code> , limit batch sizes and intervals but sizes and intervals will usually be much smaller. Setting this option may have a negative impact on maximum throughput.

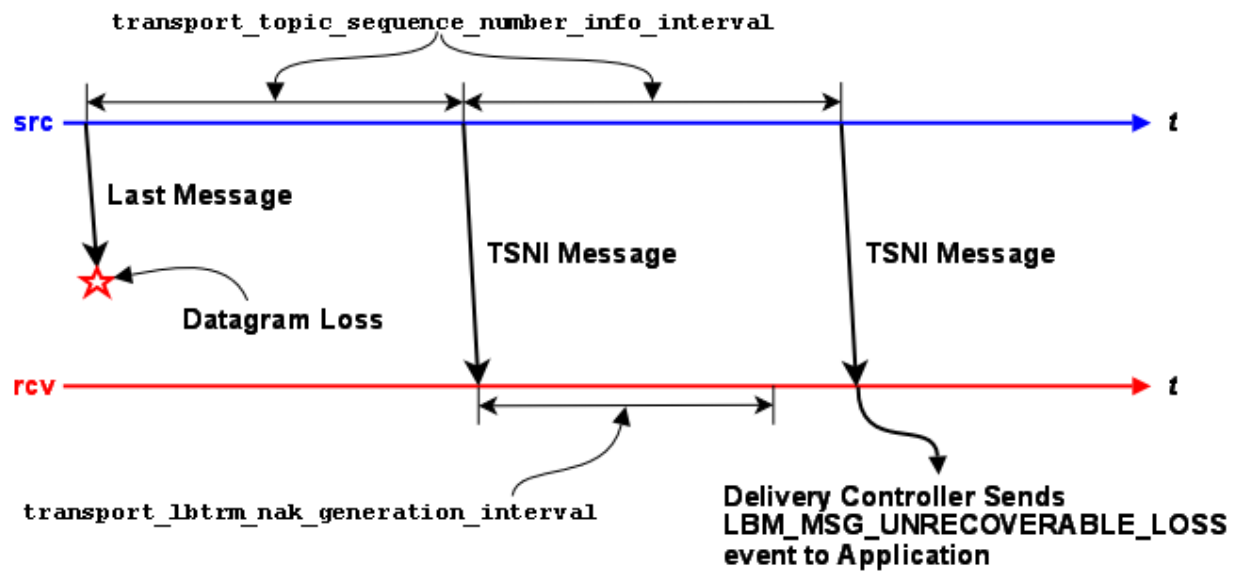
## 4.26. Delivery Control Options

A Delivery Controller is a receiver-side object created for each source identified by the receiver through topic resolution. A delivery controller performs the following.

- Delivers messages to multiple receivers subscribed to the same topic.
- Orders received topic messages if `ordered_delivery` ([../Config/majoroptions.html#RECEIVERORDEREDEDELIVERY](#)) is set to 1 (default). This option applies to LBT-RU and LBT-RM transports.
- Determines unrecoverable loss and burst loss events for the receiver's topic over LBT-RU and LBT-RM transports.

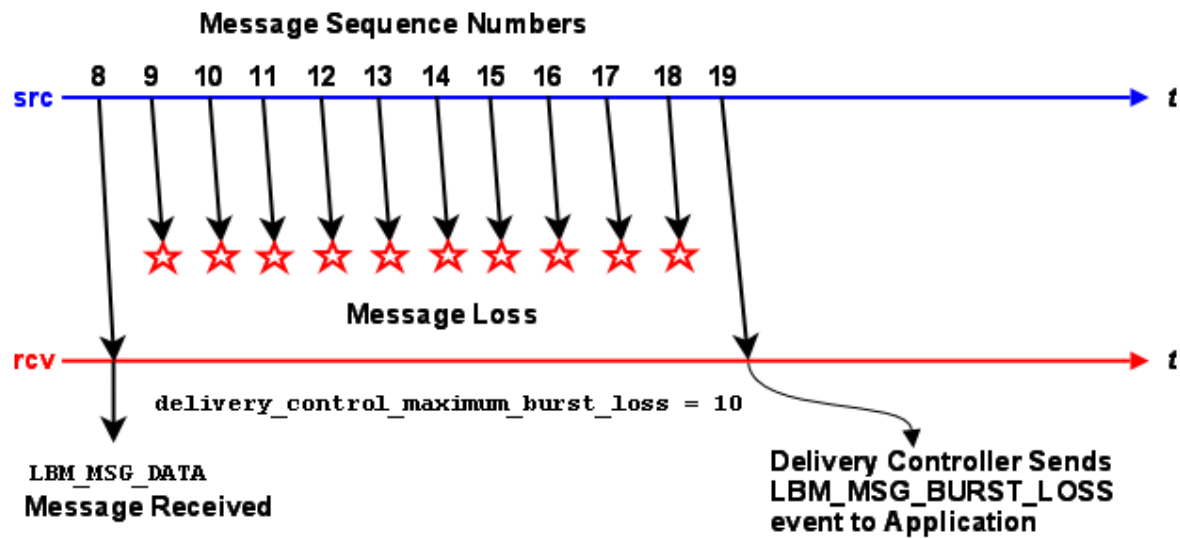
Unlike the loss depicted in LBT-RM Datagram Loss Resulting in Unrecovered Message Loss ([../Config/transportlbt-rmreliabilityoptions.html#LBT-RM-LOSS](#)) which is due to the inability of the transport or network to perform message retransmission, Figure 4-13 demonstrates how a receiver's Delivery Controller detects the loss of a topic message and notifies the receiving application. The TSNI ([../Config/majoroptions.html#SOURCETRANSPORTTOPICSEQUENCENUMBERINFOINTERVAL](#)) messages contain the sequence number of the last message sent by the source.

Figure 4-13. Generation of Unrecoverable Loss Event



The Delivery Controller detects burst loss by comparing the sequence numbers of the last two messages received. If the resulting gap in sequence numbers equals or exceeds the `delivery_control_maximum_burst_loss` (`../Config/deliverycontroloptions.html#RECEIVERDELIVERYCONTROLMAXIMUMBURSTLOSS`), the delivery controller sends `LBM_MSG_BURST_LOSS` to the application.

Figure 4-14. Generation of Burst Loss Event



### 4.26.1. `channel_map_tablesz` (receiver)

The size of the hash table that the receiver uses to store channel subscriptions. A larger table means more channels can be stored more efficiently, but takes up more memory. A smaller table uses less memory, but costs more CPU time for large numbers of channel subscriptions.

*Scope:* receiver  
*Type:* `size_t`  
*Default value:* 10273  
*When to Set:* Can only be set during object initialization.

### 4.26.2. `delivery_control_loss_check_interval` (receiver)

This controls the interval between mandatory topic loss checks for a receiver. A value of 0 turns this loss check off.

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.

### 4.26.3. `delivery_control_loss_tablesz` (receiver)

For LBT-RM and other datagram-based transport sessions only. This controls the size of the hash table index used for storing unrecoverable loss state on a per source per topic basis. Larger values mean larger hash tables and probably better CPU usage under loss scenarios at the cost of more memory per source per topic. Smaller values mean smaller hash tables and probably worse CPU usage under loss scenarios but with less memory usage. The value used should be a prime number for efficiency.

<i>Scope:</i>	receiver
<i>Type:</i>	size_t
<i>Units:</i>	table entries
<i>Default value:</i>	131
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	Deprecated

### 4.26.4. `delivery_control_maximum_burst_loss` (receiver)

This controls the maximum tolerable burst loss before a burst loss message is delivered to the application. A burst loss less than or equal to this size is treated normally. Larger burst loss is treated as unrecoverable immediately.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_uint_t
<i>Units:</i>	number of messages
<i>Default value:</i>	512
<i>When to Set:</i>	Can only be set during object initialization.

### 4.26.5. `delivery_control_maximum_total_map_entries` (context)

The maximum total buffered map entries (unrecoverable loss messages as well as data) that all topics can buffer. When this is exceeded, unrecoverable loss is signaled for data until the total buffered subsides. A value of 0 implies no maximum value setting and allows any amount required to be buffered.

<i>Scope:</i>	context
<i>Type:</i>	size_t
<i>Units:</i>	map entries
<i>Default value:</i>	200000
<i>When to Set:</i>	Can only be set during object initialization.

### 4.26.6. `delivery_control_order_tablesz` (receiver)

For LBT-RM and other datagram-based transport sessions only. This controls the size of the hash table index used

for storing buffered data on a per source per topic basis when ordered delivery is used. Larger values mean larger hash tables and probably better CPU usage under loss scenarios at the cost of more memory per source per topic. Smaller values mean smaller hash tables and probably worse CPU usage under loss scenarios but with less memory usage. The value used should be a prime number for efficiency.

*Scope:* receiver  
*Type:* size\_t  
*Units:* table entries  
*Default value:* 131  
*When to Set:* Can only be set during object initialization.  
*Version:* Deprecated

#### 4.26.7. mim\_delivery\_control\_loss\_check\_interval (context)

This controls the interval between mandatory loss checks for a Multicast Immediate Messaging (MIM) transport session. A value of 0 turns this loss check off.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2

#### 4.26.8. null\_channel\_behavior (receiver)

Behavior desired when a message without channel information (i.e. a standard **UM** message) is received by **UM**.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
deliver	LBM_RCV_TOPIC_ATTR_CHANNEL_BEHAVIOR_DELIVER	Messages sent without channel information will be delivered to the callback specified upon receiver creation. <b>Default for all.</b>
discard	LBM_RCV_TOPIC_ATTR_CHANNEL_BEHAVIOR_DISCARD	Messages sent without channel information will be discarded.

### 4.26.9. source\_notification\_function (receiver)

Callback functions (and associated client data pointer) that are called when a receiver creates or deletes a delivery controller associated with a source. For the creation function, the application has the ability to set the source client data pointer to be used in each message received from the source. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

*Scope:* receiver  
*Type:* lbm\_rcv\_src\_notification\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.26.10. unrecognized\_channel\_behavior (receiver)

Behavior desired when a message with channel information for a channel not in the receiver's set of subscribed channels is received by **UM**.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
deliver	LBM_RCV_TOPIC_ATTR_CHANNEL_BEHAVIOR_DELIVER	Messages sent with channel information for a channel not in the receiver's set of subscribed channels will be delivered to the callback specified upon receiver creation. <b>Default for all.</b>
discard	LBM_RCV_TOPIC_ATTR_CHANNEL_BEHAVIOR_DISCARD	Messages sent with channel information for a channel not in the receiver's set of subscribed channels will be discarded.

## 4.27. Wildcard Receiver Options

### 4.27.1. pattern\_callback (wildcard\_receiver)

Callback function (and associated client data pointer) that is called when a pattern match is desired for a topic discovered for a wildcard receiver if the pattern type is set to "appcb". This callback is called directly in line and does not use the event queue. A return value of 0 indicates the given topic should be considered part of the wildcard. A value of 1 or more indicates the topic should NOT be considered matching the wildcard.



*Scope:* wildcard\_receiver  
*Type:* lbm\_wildcard\_rcv\_compare\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.27.2. pattern\_type (wildcard\_receiver)

The type of pattern matching in use for the wildcard receiver. This is the behavior used for the pattern compare when new topics are seen.

*Scope:* wildcard\_receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
pcre	LBM_WILDCARD_RCV_PATTERN_TYPE_PCRE	The pattern of the wildcard is assumed to be a regular expression usable by PCRE (Perl Compatible Regular Expressions) library. <b>Default for all if supported by platform.</b>
regex	LBM_WILDCARD_RCV_PATTERN_TYPE_REGEX	The pattern of the wildcard is assumed to be a regular expression usable by POSIX Extended Regular Expressions. <b>Default for all if supported and PCRE is not supported.</b>
appcb	LBM_WILDCARD_RCV_PATTERN_TYPE_APPCB	The pattern is ignored and an application callback (set by the pattern_callback attribute) is called for each pattern match compare. <b>Default for all if neither PCRE nor REGEX is supported.</b>

### 4.27.3. receiver\_create\_callback (wildcard\_receiver)

Callback function (and associated client data pointer) that is called when a receiver is about to be created for a topic which matched a wildcard receiver pattern. This callback is called directly in line and does not use the event queue. The callback function should always return 0.

*Scope:* wildcard\_receiver  
*Type:* lbm\_wildcard\_rcv\_create\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

*Version:* This option was implemented in LBM 3.4/UME 2.1.

#### 4.27.4. receiver\_delete\_callback (wildcard\_receiver)

Callback function (and associated client data pointer) that is called when a receiver is about to be deleted. This callback is called directly in line and does not use the event queue. The callback function should always return 0.

*Scope:* wildcard\_receiver  
*Type:* lbm\_wildcard\_rcv\_delete\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

#### 4.27.5. resolver\_no\_source\_linger\_timeout (wildcard\_receiver)

This sets the linger timeout value before a topic with no sources is removed and cleaned up. Since wildcard receivers set the `resolution_no_source_notification_threshold` to 10, the linger timer starts after the wildcard receiver sends 10 queries and subsequently receives a no-source notification.

*Scope:* wildcard\_receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.

#### 4.27.6. resolver\_query\_maximum\_interval (wildcard\_receiver)

The longest - and last - interval in wildcard receiver topic querying. A value of 0 disables wildcard receiver topic querying. See also [Disabling Aspects of Topic Resolution \(disable-topic-res.html\)](#).

*Scope:* wildcard\_receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.0

### 4.27.7. resolver\_query\_minimum\_duration (wildcard\_receiver)

The duration of wildcard queries in wildcard receiver topic querying. Only PCRE and regex pattern types can use wildcard queries. A value of 0 guarantees that wildcard receiver topic querying never completes.

<i>Scope:</i>	wildcard_receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	seconds
<i>Default value:</i>	60 (1 minute)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

### 4.27.8. resolver\_query\_minimum\_interval (wildcard\_receiver)

Interval between the first topic query sent upon creation of the wildcard receiver and the second query sent by the receiver. A value of 0 disables wildcard receiver topic querying. See also [Disabling Aspects of Topic Resolution \(disable-topic-res.html\)](#). This option has an effective minimum of 30 ms. See [Minimum Values for Advertisement and Query Intervals \(resolveroperationoptions.html\)](#).

<i>Scope:</i>	wildcard_receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	50 (0.05 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

### 4.27.9. resolver\_wildcard\_queries\_per\_second (context)

Maximum number of queries sent within a one second period during wildcard receiver topic querying. A value of 0 sets no rate limit on queries in wildcard receiver topic querying.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	advertisements
<i>Default value:</i>	0
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

### 4.27.10. resolver\_wildcard\_query\_bps (context)

Maximum query rate during wildcard receiver topic querying. A value of 0 sets no rate limit on queries in wildcard receiver topic querying.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint64_t
<i>Units:</i>	bits per second
<i>Default value:</i>	1000000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.0

### 4.27.11. resolver\_wildcard\_receiver\_map\_tablesz (context)

The size of the hash table used for storing wildcard receiver patterns. A value of 0 disables caching wildcard receiver patterns. This value should be a prime number.

<i>Scope:</i>	context
<i>Type:</i>	size_t
<i>Units:</i>	map entries
<i>Default value:</i>	10273
<i>When to Set:</i>	Can only be set during object initialization.

## 4.28. Event Queue Options

### 4.28.1. event\_queue\_name (event\_queue)

The name of an event queue, limited to 128 alphanumeric characters, hyphens or underscores.

<i>Scope:</i>	event_queue
<i>Type:</i>	string
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.3/UMP 3.3/UMQ 2.3.

### 4.28.2. queue\_age\_enabled (event\_queue)

Controls whether the length of time each event spends on the event queue is measured. Useful only if you are monitoring event queue statistics.

*Scope:* event\_queue  
*Type:* int  
*Default value:* 0  
*When to Set:* May be set during operation.

Value	Description
1	Enables measuring of event queue entry ages.
0	Disables measuring of event queue entry ages. <b>Default for all.</b>

### 4.28.3. queue\_cancellation\_callbacks\_enabled (event\_queue)

Flag indicating whether the event queue is to do appropriate locking to provide cancellation callback support for cancel/delete functions.

*Scope:* event\_queue  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Provide support for cancellation callbacks.
0	Do not provide cancellation callback support. <b>Default for all.</b>

### 4.28.4. queue\_count\_enabled (event\_queue)

Controls whether the numbers of each type of queue entry are counted. Useful only if you are monitoring event queue statistics.

*Scope:* event\_queue  
*Type:* int  
*Default value:* 0  
*When to Set:* May be set during operation.

Value	Description
1	Enables counting event queue entries.
0	Disables counting of event queue entries. <b>Default for all.</b>

### 4.28.5. queue\_delay\_warning (event\_queue)

The event queue delay threshold (in microseconds) at which the monitor function for the event queue is called. This delay is the time that an event has been queued before being dispatched. A value of 0 indicates the event queue delay is not to be monitored and checked.

*Scope:* event\_queue  
*Type:* lbm\_ulong\_t  
*Units:* microseconds  
*Default value:* 0 (not monitored)  
*When to Set:* May be set during operation.

### 4.28.6. queue\_enqueue\_notification (event\_queue)

Flag indicating whether to call the monitor function when an event is enqueued into the given event queue. The thread enqueueing the event is the one that calls this function. So, when this is called, the monitoring function in use should only assume this is only notification of enqueueing. The monitor function should not dispatch events directly.

*Scope:* event\_queue  
*Type:* int  
*When to Set:* May be set during operation.

Value	Description
1	Enable notification.
0	Disable notification. <b>Default for all.</b>

### 4.28.7. queue\_objects\_purged\_on\_close (event\_queue)

Flag indicating whether the event queue should be immediately purged of any pending events associated with a recently closed object (e.g. source, receiver) during the close operation, or be left on the queue to be discarded as the event queue drains normally. In either case, UM does not deliver the defunct events to the application. The `Immediate purge` setting reclaims memory immediately, while the `Delay purge.` setting spreads the reclamation work over time, reducing the CPU impact of closing objects associated with the queue.

*Scope:* event\_queue  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	Immediate purge. <b>Default for all.</b>
0	Delay purge.

### 4.28.8. queue\_service\_time\_enabled (event\_queue)

Controls whether the amount of time required to service each event on the event queue is measured. Useful only if you are monitoring event queue statistics.

*Scope:* event\_queue  
*Type:* int  
*Default value:* 0  
*When to Set:* May be set during operation.

Value	Description
1	Enables measuring of event queue service times.
0	Disables measuring of event queue service times. <b>Default for all.</b>

### 4.28.9. queue\_size\_warning (event\_queue)

The event queue size threshold (in number of events) at which the monitor function for the event queue is called. A value of 0 indicates the event queue size is not to be monitored and checked.

*Scope:* event\_queue  
*Type:* lbm\_ulong\_t  
*Units:* number of events  
*Default value:* 0 (not monitored)  
*When to Set:* May be set during operation.

## 4.29. Ultra Messaging Persistence Options

### 4.29.1. ume\_ack\_batching\_interval (context)

The interval between checks by **UMP** of consumed, unacknowledged messages. See also ume\_use\_ack\_batching.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 100 (0.1 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0, UMP 5.0, UMQ 5.0.

### 4.29.2. ume\_activity\_timeout (receiver)

Establishes the period of time from a receiver's last activity to the release of the receiver's Reg ID. Stores return an error to any new request for the receiver's Reg ID during this period. Overrides the `receiver-activity-timeout` setting configured for the receiver's topic on the store. The default value of 0 (zero) disables this option. See also Proxy Sources ([../UME/ume-fault-tolerance.html#PROXYSOURCES](#)).

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.

### 4.29.3. ume\_activity\_timeout (source)

Establishes the period of time from a source's last activity to the release of the source's Reg ID. Stores return an error to any new source requesting the source's Reg ID during this period. If proxy sources are enabled (`ume_proxy_source`), the store does not release the source's Reg ID and **UMP** elects a proxy source. Overrides the `source-activity-timeout` setting configured for the source's topic on the store. The default value of 0 (zero) disables this option. If neither proxy sources nor `ume_state_lifetime` are configured, the store also deletes the source's state and cache. See also Proxy Sources ([../UME/ume-fault-tolerance.html#PROXYSOURCES](#)).

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.

### 4.29.4. ume\_allow\_confirmed\_delivery (receiver)

Specifies whether or not **UMP** allows the sending of confirmed delivery notifications back to the source.

*Scope:* receiver  
*Type:* `int`  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0, UMP 5.0, UMQ 5.0.

Value	Description
1	Indicates that <b>UMP</b> can send confirmed delivery notifications. <b>Default for all.</b>



Value	Description
0	Indicates that <b>UMP</b> can not send confirmed delivery notifications.

#### 4.29.5. ume\_confirmed\_delivery\_notification (source)

Flag indicating the source is interested in receiving notifications of delivery of messages to receivers (confirmed delivery) via the source event mechanism. When turned off, receivers do not send delivery confirmation notifications to the source unless the release policy dictates the need for them.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
0	LBM_SRC_TOPIC_ATTR_UME_CDELV_EVENT_0	The source does not wish to receive delivery confirmation notifications.
1	LBM_SRC_TOPIC_ATTR_UME_CDELV_EVENT_1	The source wishes to receive delivery confirmation notifications for all messages and message fragments. <b>Default for all.</b>
2	LBM_SRC_TOPIC_ATTR_UME_CDELV_EVENT_2	The source wishes to receive only one delivery confirmation for a message regardless of how many fragments it comprised.
3	LBM_SRC_TOPIC_ATTR_UME_CDELV_EVENT_3	The source wishes to receive delivery confirmation notifications for all messages and message fragments. In addition, the notification contains a <b>WHOLE_MESSAGE_CONFIRMED</b> flag when the last fragment of a message has been delivered.

#### 4.29.6. ume\_consensus\_sequence\_number\_behavior (receiver)

The behavior that the receiver will follow when determining the consensus sequence number used as the sequence number to begin reception at upon re-registration after a failure or suspension. This setting is only used when quorum-consensus is also used on the source.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
--------------	---------------	-------------

String value	Integer value	Description
lowest	LBM_RCV_TOPIC_ATTR_UME_QC_SEQN_BEHAV	Consensus is determined as the lowest of the latest sequence numbers seen from any store.
majority	LBM_RCV_TOPIC_ATTR_UME_QC_SEQN_BEHAV	Consensus is determined as the latest sequence number agreed upon by the majority of stores within a group. Between groups, the latest of all majority decisions is used. <b>Default for all.</b>
highest	LBM_RCV_TOPIC_ATTR_UME_QC_SEQN_BEHAV	Consensus is determined as the highest of the latest sequence numbers seen from any store.

#### 4.29.7. ume\_consensus\_sequence\_number\_behavior (source)

The behavior that the source will follow when determining the consensus sequence number used as the first message of a source upon re-registration after a failure or suspension. This setting is only used when quorum-consensus is also used.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
lowest	LBM_SRC_TOPIC_ATTR_UME_QC_SEQN_BEHAV	Consensus is determined as the lowest of the latest sequence numbers seen from any store.
majority	LBM_SRC_TOPIC_ATTR_UME_QC_SEQN_BEHAV	Consensus is determined as the latest sequence number agreed upon by the majority of stores within a group. Between groups, the latest of all majority decisions is used. <b>Default for all.</b>
highest	LBM_SRC_TOPIC_ATTR_UME_QC_SEQN_BEHAV	Consensus is determined as the highest of the latest sequence numbers seen from any store.

#### 4.29.8. ume\_explicit\_ack\_only (receiver)

Flag indicating if the receiver should automatically send acknowledgements to any stores and to the source or if the application desires to explicitly generate acknowledgements itself. See also [Explicit Acknowledgments](#) ([../UME/designing-persistent-applications.html#EXPLICITACKS](#)).

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
-------	-------------

Value	Description
1	The receiving application will generate acknowledgements explicitly and the <b>UMP</b> receiver should not automatically generate them.
0	The <b>UMP</b> receiver will automatically generate and send acknowledgements based on message consumption. <b>Default for all.</b>

#### 4.29.9. ume\_flight\_size (source)

Specifies the number of messages allowed to be in flight (unstabilized at a store and without delivery confirmation) before a new message send either blocks or triggers a notification (source event).

*Scope:* source  
*Type:* unsigned int  
*Units:* messages  
*Default value:* 1000  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1.1/UME 3.1.1

#### 4.29.10. ume\_flight\_size\_behavior (source)

The behavior that **UMP** follows when a message send exceeds the source's ume\_flight\_size.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1.1/UME 3.1.1

String value	Integer value	Description
Block	LBM_FLIGHT_SIZE_BEHAVIOR_BLOCK	The send call blocks when a source sends a message that exceeds its flight size. If the source uses a non-blocking send, the send returns an <b>LBM_EWOULD_BLOCK</b> . <b>Default for all.</b>
Notify	LBM_FLIGHT_SIZE_BEHAVIOR_NOTIFY	A message send that exceeds the configured flight size does not block but triggers a flight size notification (source event), indicating that the flight size has been surpassed. <b>UMP</b> also sends a source event notification if the number of in-flight messages falls below the configured flight size.

### 4.29.11. ume\_flight\_size\_bytes (source)

Specifies the message payload in bytes allowed to be in flight (unstabilized at a store and without delivery confirmation) before a new message send either blocks or triggers a notification source event. **UMP** monitors both this option and ume\_flight\_size. If either threshold is met, the configured blocking or notification behavior executes. See ume\_flight\_size\_behavior. When using Receiver-paced Persistence, set this option greater than 0 (zero) but less than or equal to the repository's source-flight-size-bytes-maximum value.

*Scope:* source  
*Type:* lbm\_uint64\_t  
*Units:* bytes  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

### 4.29.12. ume\_force\_reclaim\_function (source)

Callback function (and associated client data pointer) that is called when a source is forced to release a retained message due to size limitations specified. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

*Scope:* source  
*Type:* lbm\_ume\_src\_force\_reclaim\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.29.13. ume\_late\_join (source)

Flag indicating the source should allow late join operation for receivers and persistent stores. This is a compatibility setting. The late\_join setting should be used instead.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	The source allows late join receivers and persistent stores.

Value	Description
0	The source does not allow late join receivers or persistent stores. <b>Default for all.</b>

#### 4.29.14. ume\_message\_stability\_notification (source)

Flag indicating the source is interested in receiving notifications of message stability from persistent stores via the source event mechanism. Even when turned off, stores continue to send message stability notifications to the source for retention purposes. However, no notification will be delivered to the application.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
0	LBM_SRC_TOPIC_ATTR_UME_STABLE_EVENT_DISABLED	The source does not wish to receive message stability notifications from the store.
1	LBM_SRC_TOPIC_ATTR_UME_STABLE_EVENT_WHOLE_MESSAGE	The source wishes to receive all message and message fragment stability notifications from the store. <b>Default for all.</b>
2	LBM_SRC_TOPIC_ATTR_UME_STABLE_EVENT_LAST_FRAGMENT	The source wishes to receive only a single message stability notifications from the store when the entire message has been stabilized. This notification contains the Sequence Number of the last fragment of the whole message but does NOT contain store information.
3	LBM_SRC_TOPIC_ATTR_UME_STABLE_EVENT_WHOLE_MESSAGE_FRAGMENT	The source wishes to receive all message and message fragment stability notifications from the store. In addition, the notification contains a <b>WHOLE_MESSAGE_STABLE</b> flag when the last fragment of a message has been stabilized.

#### 4.29.15. ume\_proxy\_source (source)

Controls whether any stores with which the source registers should provide a proxy source in the event the actual source terminates. Proxy source support is only available for quorum/consensus store configurations. In addition, proxy source support requires that the source register with an actual registration ID, and not request that the store assign it a registration ID.

*Scope:* source  
*Type:* int  
*Default value:* 0

*When to Set:* Can only be set during object initialization.

Value	Description
1	Enables proxy source support.
0	Disables proxy source support. <b>Default for all.</b>

#### 4.29.16. `ume_receiver_liveness_interval` (context)

The maximum interval between delivery confirmations or keepalive messages send to the source. Expiration of this interval triggers another keepalive and an interval reset.

*Scope:* context  
*Type:* int  
*Units:* milliseconds  
*Default value:* 0 (disable; do not send keepalives)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.2.

#### 4.29.17. `ume_receiver_paced_persistence` (receiver)

Specifies that the receiver is a Receiver-paced Persistence (RPP) receiver. If the repository has set `repository-allow-receiver-paced-persistence` to 0 (disable), setting this option to 1 creates a store registration error.

*Scope:* receiver  
*Type:* `lbm_uint8_t`  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

Value	Description
1	Indicates that the receiver is a RPP receiver.
0	Indicates that the receiver is not a RPP receiver. <b>Default for all.</b>

#### 4.29.18. `ume_receiver_paced_persistence` (source)

Specifies that the source is a Receiver-paced Persistence (RPP) source and may change certain topic repository options to values allowed by the repository. If the repository has set `repository-allow-receiver-paced-persistence` to 0 (disable), setting this option to 1 creates a store

registration error.

*Scope:* source  
*Type:* lbm\_uint8\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

Value	Description
1	Indicates that source is a RPP source.
0	Indicates that source is not a RPP source. <b>Default for all.</b>

#### 4.29.19. ume\_recovery\_sequence\_number\_info\_function (receiver)

Callback function (and associated client data pointer) that is called when a receiver is about to complete registration from the stores in use by the source and the low sequence number is to be determined. The application has the ability to modify the sequence number to use if it desires. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

*Scope:* receiver  
*Type:* lbm\_ume\_rcv\_recovery\_info\_ex\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

#### 4.29.20. ume\_registration\_extended\_function (receiver)

Callback function (and associated client data pointer) that is called when a receiver is about to attempt to register with a persistent store. The app must return the registration ID to request from the store or 0 if it will allow the store to allocate one. This function passes additional extended information, such as the store being used and a source client data pointer, etc. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called.

*Scope:* receiver  
*Type:* lbm\_ume\_rcv\_regid\_ex\_func\_t  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.29.21. `ume_registration_function` (receiver)

Callback function (and associated client data pointer) that is called when a receiver is about to attempt to register with a persistent store. The app must return the registration ID to request from the store or 0 if it will allow the store to allocate one. This callback is called directly in line and does not use the event queue. Therefore the callback function used should not block or it will block the context thread processing. A value of NULL for the callback turns off the callback being called. This setting is provided for compatibility. The `ume_registration_extended_function` setting should be used instead.

*Scope:* receiver  
*Type:* `lbm_ume_rcv_regid_func_t`  
*Default value:* NULL  
*When to Set:* Can only be set during object initialization.  
*Config File:* Cannot be set from an UM configuration file.

### 4.29.22. `ume_registration_interval` (receiver)

The interval between registration attempts by the receiver to a persistent store in use by the source.

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.29.23. `ume_registration_interval` (source)

The interval between registration attempts by the source. When using the round-robin store behavior, this is the value between registration attempts with the various stores. In other words, attempt to register with primary, wait interval, attempt to register with secondary, wait interval, etc.

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.

### 4.29.24. `ume_repository_ack_on_reception` (source)

For topics with a `repository-type` of `disk` or `reduced-fd`, specifies that the stability acknowledgement should be sent upon message reception by the store instead of when the message has been written to disk. When using



Receiver-paced Persistence, if the repository has set `repository-allow-ack-on-reception` to 0 (disable), setting this option to 1 creates a store registration error. This option has no effect on Source-paced Persistence repositories.

*Scope:* source  
*Type:* `lbm_uint8_t`  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

Value	Description
1	The repository sends a stability acknowledgement for a message as soon as it has received the message.
0	The repository sends a stability acknowledgement for a message once it has been written to disk. <b>Default for all.</b>

#### 4.29.25. `ume_repository_disk_file_size_limit` (source)

For topics with a `repository-type` of `disk` or `reduced-fd`, specifies the maximum amount of disk space used to store retained messages. Using the default value of 0 (zero) implements the repository's `repository-disk-file-size-limit` value. When not set to 0, **UMP** enforces a minimum value of 196992. When using Receiver-paced Persistence and you require that the source dictates the repository's `repository-disk-file-size-limit` value, you must set this option greater than 0 (zero) but less than or equal to the repository's current `repository-disk-file-size-limit` value.

*Scope:* source  
*Type:* `lbm_uint64_t`  
*Units:* bytes  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

#### 4.29.26. `ume_repository_size_limit` (source)

For topics with a `repository-type` of `memory`, `disk` or `reduced-fd`, specifies the maximum number of message bytes retained (includes payload only). When using Receiver-paced Persistence and you require that the source dictates the repository's `repository-size-limit` value, you must set this option greater than 0 (zero) but less than or equal to the repository's current `repository-size-limit` value. For the `disk` or `reduced-fd` repository type, this value configures the size of the memory cache. Using the default value of 0 (zero) implements the repository's value for this option.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

#### 4.29.27. ume\_repository\_size\_threshold (source)

For topics with a repository-type of memory, disk or reduced-fd, specifies the minimum number of message bytes retained (includes payload only). When using Receiver-paced Persistence and you require that the source dictates the repository's repository-size-threshold value, you must set this option greater than 0 (zero) but less than or equal to the repository's current repository-size-threshold value. For the disk or reduced-fd repository type, this value configures the size of the memory cache. Using the default value of 0 (zero) implements the repository's value for this option.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

#### 4.29.28. ume\_retention\_intergroup\_stability\_behavior (source)

The behavior that the source will follow when determining the stability of a message from an inter-group perspective. This has a direct impact on the release policy for the source in that a message must be stable before it may be released. To be stable, a message must first be stable within the group and then stable between groups.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
any, any-group	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_ANY	Message is considered stable once it is stable in any group. <b>Default for all.</b>
majority	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_MAJORITY	Message is considered stable once it is stable in a majority of groups.
all, all-groups	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_ALL	Message is considered stable once all groups have reached intra-group stability for the message.

String value	Integer value	Description
all-active	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_ACTIVE	Message is considered stable once it is stable in all active groups. A group is considered active if it has at least a quorum of active or registered stores. Intergroup stability requires at least one stable group.

#### 4.29.29. ume\_retention\_intragroup\_stability\_behavior (source)

The behavior that the source will follow when determining the stability of a message from an intra-group perspective. This has a direct impact on the release policy for the source in that a message must be stable before it may be released. To be stable, a message must first be stable within the group and then stable between groups.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
quorum	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_QUORUM	Message is considered stable within the group once a quorum (or majority) of the stores have acknowledged the message as stable. <b>Default for all.</b>
all, all-stores	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_ALL	Message is considered stable within the group once all stores have acknowledged the message as stable.
all-active	LBM_SRC_TOPIC_ATTR_UME_STABLE_BEHAVIOR_ACTIVE	Message is considered stable within the group once each registered store in that group has acknowledged the message as stable. Only stores registered with a source are considered "active".

#### 4.29.30. ume\_retention\_size\_limit (source)

The release policy regarding aggregate size limit before messages are forced to be released. If the total number of bytes retained for the source is less than this amount, they may be released depending on other retention settings. If the total number of bytes exceeds this amount, then the message is forced to be released and a log message generated. This setting is provided for compatibility. The `retransmit_retention_size_limit` setting should be used instead.

*Scope:* source  
*Type:* size\_t  
*Units:* bytes  
*Default value:* 25165824 (24 MB)  
*When to Set:* Can only be set during object initialization.

### 4.29.31. `ume_retention_size_threshold` (source)

The release policy regarding aggregate size threshold before messages are released. If the total number of bytes retained for the source is less than this amount, they will not be released. If the total number of bytes exceeds this amount, then the message may be released if no other release policy setting overrides the decision. A value of 0 indicates there is no size threshold set. This setting is provided for compatibility. The `retransmit_retention_size_threshold` setting should be used instead.

<i>Scope:</i>	source
<i>Type:</i>	size_t
<i>Units:</i>	bytes
<i>Default value:</i>	0 (no threshold)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.29.32. `ume_retention_unique_confirmations` (source)

The release policy regarding the number of confirmations from different receivers required before the source can release a message. This option enhances, but does not supersede, message stability notification from the store(s). If the number of unique confirmations for a message is less than this amount, the message will not be released. If the number of unique confirmations for a message exceeds or equals this amount, then the message may be released if no other release policy setting overrides the decision. A value of 0 indicates there is no unique number of confirmations required for reclamation.

<i>Scope:</i>	source
<i>Type:</i>	size_t
<i>Units:</i>	number of confirmations
<i>Default value:</i>	0 (none required)
<i>When to Set:</i>	Can only be set during object initialization.

### 4.29.33. `ume_retransmit_request_generation_interval` (receiver)

The maximum interval between when a retransmission request is first sent and when it is given up on and loss is reported. This setting is provided for compatibility. The `retransmit_request_generation_interval` setting should be used instead.

<i>Scope:</i>	receiver
<i>Type:</i>	unsigned long int
<i>Units:</i>	milliseconds
<i>Default value:</i>	10000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.29.34. `ume_retransmit_request_interval` (receiver)

The interval between retransmission request messages to the persistent store or to the source. This setting is provided for compatibility. The `retransmit_request_interval` setting should be used instead.

<i>Scope:</i>	receiver
<i>Type:</i>	unsigned long int
<i>Units:</i>	milliseconds
<i>Default value:</i>	500 (0.5 seconds)
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.29.35. `ume_retransmit_request_maximum` (receiver)

The maximum number of messages to request back from the current latest message when late joining a topic or when registering with a **UMP** store. A value of 0 indicates no maximum. This setting is provided for compatibility. The `retransmit_request_maximum` setting should be used instead.

<i>Scope:</i>	receiver
<i>Type:</i>	unsigned long int
<i>Units:</i>	messages
<i>Default value:</i>	0
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.29.36. `ume_retransmit_request_outstanding_maximum` (receiver)

The maximum number of messages to request at a single time from the store or source. A value of 0 indicates no maximum. This setting is provided for compatibility. The `retransmit_request_outstanding_maximum` setting should be used instead.

<i>Scope:</i>	receiver
<i>Type:</i>	unsigned long int
<i>Units:</i>	messages
<i>Default value:</i>	200
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.29.37. `ume_session_id` (context)

Specifies the default Session ID to use for sources and receivers within a context. A value of 0 (zero) indicates no Session ID is to be set. See also [Managing RegIDs with Session IDs](#).

([../UME/designing-persistent-applications.html#SESSION-IDS](#)). Valid formats for session IDs are as follows: A hexadecimal string with a maximum value of FFFFFFFFFFFFFFFF, prefixed with '0x'. An octal string with a

maximum value of 177777777777777776 prefixed with '0'. A decimal string with a maximum value of 18446744073709551614. Prior to **LBM 5.2.2**, all **UME** session IDs were interpreted as hexadecimal, and did not accept the '0x' prefix. If upgrading from an earlier version to **LBM 5.2.2** or later, prepend '0x' to the original setting to use the originally assigned session ID.

*Scope:* context  
*Type:* lbm\_uint64\_t  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2

#### 4.29.38. ume\_session\_id (receiver)

Specifies the Session ID to use for a receiver. A value of 0 (zero) indicates the context ume\_session\_id will be used. See also [Managing RegIDs with Session IDs. \(../UME/designing-persistent-applications.html#SESSION-IDS\)](#). Valid formats for session IDs are as follows: A hexadecimal string with a maximum value of FFFFFFFFFFFFFFFFE, prefixed with '0x'. An octal string with a maximum value of 177777777777777776 prefixed with '0'. A decimal string with a maximum value of 18446744073709551614. Prior to **LBM 5.2.2**, all **UME** session IDs were interpreted as hexadecimal, and did not accept the '0x' prefix. If upgrading from an earlier version to **LBM 5.2.2** or later, prepend '0x' to the original setting to use the originally assigned session ID.

*Scope:* receiver  
*Type:* lbm\_uint64\_t  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2

#### 4.29.39. ume\_session\_id (source)

Specifies the Session ID to use for a source. A value of 0 (zero) indicates the context ume\_session\_id will be used. See also [Managing RegIDs with Session IDs. \(../UME/designing-persistent-applications.html#SESSION-IDS\)](#). Valid formats for session IDs are as follows: A hexadecimal string with a maximum value of FFFFFFFFFFFFFFFFE, prefixed with '0x'. An octal string with a maximum value of 177777777777777776 prefixed with '0'. A decimal string with a maximum value of 18446744073709551614. Prior to **LBM 5.2.2**, all **UME** session IDs were interpreted as hexadecimal, and did not accept the '0x' prefix. If upgrading from an earlier version to **LBM 5.2.2** or later, prepend '0x' to the original setting to use the originally assigned session ID.

*Scope:* source  
*Type:* lbm\_uint64\_t  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2

#### 4.29.40. ume\_source\_liveness\_timeout (context)

The expected maximum interval between keepalive or delivery confirmation messages from a receiver. If neither are received within the interval, the source declares the receiver "dead".

<i>Scope:</i>	context
<i>Type:</i>	int
<i>Units:</i>	milliseconds
<i>Default value:</i>	0 (disable; do not track receivers)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMP 5.2.

#### 4.29.41. ume\_state\_lifetime (receiver)

Establishes the period of time from a receiver's last activity to the deletion of the receiver's state and cache by the store. You can also configure a `receiver-state-lifetime` for the receiver's topic on the store. The store uses whichever is shorter. The default value of 0 (zero) disables this option. See also Proxy Sources ([../UME/ume-fault-tolerance.html#PROXYSOURCES](#)).

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	0 (zero)
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.29.42. ume\_state\_lifetime (source)

Establishes the period of time from a source's last activity to the deletion of the source's state and cache by the store, regardless of whether a proxy source has been created or not. You can also configure a `source-state-lifetime` for the source's topic on the store. The store uses whichever is shorter. The default value of 0 (zero) disables this option. See also Proxy Sources ([../UME/ume-fault-tolerance.html#PROXYSOURCES](#)).

<i>Scope:</i>	source
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	0 (zero)
<i>When to Set:</i>	Can only be set during object initialization.

#### 4.29.43. ume\_store (source)

Add a store specification to the list of stores specified for the source. Unlike most other **UMP** settings, every time

this setting is called, it adds another store specification to the list and does NOT overwrite previous specifications. Each entry contains the IP address, TCP port, registration ID, and group index for the store. For the configuration file as well as string versions of setting this option, the string value is formatted as "IP:port:RegID:GroupIDX" where IP is the stores IP address, port is the TCP port for the store, RegID is the registration ID that the source desires to use, and GroupIDX is the group index that the store belongs to. The RegID and GroupIDX pieces may be left off the string if desired. If so, then the value of 0 is assumed for them. Because each entry adds a new store specification and does not overwrite previous values, an entry or string with the IP address of 0.0.0.0 and TCP port of 0 will cause all previous store specifications to be removed. A single store specification means the source will use persistence. If no stores are specified, then persistence will not be provided for the source.

*Scope:* source  
*Type:* lbm\_ume\_store\_entry\_t  
*When to Set:* Can only be set during object initialization.

#### 4.29.44. ume\_store\_activity\_timeout (source)

The timeout value used to indicate when a store is unresponsive. The store must not be active within this interval to be considered unresponsive. This value must be much larger than the check interval.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 3000 (3 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.29.45. ume\_store\_behavior (source)

The behavior that the source will follow for handling store failures.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.

String value	Integer value	Description
rr, round-robin	LBM_SRC_TOPIC_ATTR_UME_STORE_BEHAVIOR_ROUNDROBIN	The source will use a single store at a time and when a store is unresponsive due to failure or disconnect, the next store in the list will be used. This will continue in a round-robin fashion until a store is found that is available. <b>Default for all.</b>



String value	Integer value	Description
qc, quorum-consensus	LBM_SRC_TOPIC_ATTR_UME_STORE_BEHAVIOR	The source will use multiple stores at the same time based on store and store group configuration.

#### 4.29.46. ume\_store\_check\_interval (source)

The interval between activity checks of the current store.

*Scope:* source  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.

#### 4.29.47. ume\_store\_group (source)

Add a store group specification to the list of store groups specified for the source. Unlike other **UMP** settings, every time this setting is called, it adds another store group specification to the list and does NOT overwrite previous specifications. Each entry contains the group index and group size for the group. For the configuration file as well as string versions of setting this option, the string value is formatted as "GroupIDX:GroupSZ" where GroupIDX is the index of the group and GroupSZ is the size of the group. Because each entry adds a new store specification and does not overwrite previous values, an entry or string with the group index of 0 and group size of 0 will cause all previous store group specifications to be removed.

*Scope:* source  
*Type:* lbm\_ume\_store\_group\_entry\_t  
*When to Set:* Can only be set during object initialization.

#### 4.29.48. ume\_store\_name (source)

Add a store specification to the list of stores specified for the source. Unlike other **UMP** settings, every time this setting is called, it adds another store specification to the list and does NOT overwrite previous specifications. Each entry contains the store name, registration ID, and group index for the store. For the configuration file as well as string versions of setting this option, the string value is formatted as "name:RegID:GroupIDX" where *name* is the name of the store configured with the store attribute, *context-name* in the *umestored* XML configuration file, *RegID* is the registration ID that the source desires to use, and *GroupIDX* is the group index that the store belongs to. The *RegID* and *GroupIDX* pieces may be left off the string if desired. If so, then the value of 0 is assumed for them. Store names are restricted to 128 characters in length, and may contain only alphanumeric characters, hyphens, and underscores.

*Scope:* source  
*Type:* lbm\_ume\_store\_name\_entry\_t  
*When to Set:* Can only be set during object initialization.

#### 4.29.49. ume\_use\_ack\_batching (receiver)

Specifies whether or not **UMP** allows the batching of consumption acknowledgments sent to the store(s). If enabled, **UMP** checks for contiguous sequence numbered messages at the ume\_ack\_batching\_interval. See also [Batching Acknowledgments \(../UME/designing-persistent-applications.html#BATCHINGACKS\)](#).

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.0, UMP 5.0, UMQ 5.0.

Value	Description
1	Indicates that <b>UMP</b> can acknowledge the consumption of a batch of messages.
0	Indicates that <b>UMP</b> acknowledges the consumption of individual messages by the receiver. <b>Default for all.</b>

#### 4.29.50. ume\_use\_late\_join (receiver)

Flag indicating if the receiver should participate in late join operation or not. This is a compatibility setting. The use\_late\_join setting should be used instead.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	The receiver will participate in using late join if requested to by the source. <b>Default for all.</b>
0	The receiver will not participate in using late join even if requested to by the source.

### 4.29.51. ume\_use\_store (receiver)

Flag indicating if the receiver should participate in using a persistent store or not.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	The receiver will participate in using a persistent store if requested to by the source. <b>Default for all.</b>
0	The receiver will not participate in using a persistent store even if requested to by the source.

### 4.29.52. ume\_user\_receiver\_registration\_id (context)

32-bit value that is used as a user set identifier to be included as the receiver registration ID in acknowledgements send by any receivers in the context to sources as confirmed delivery notifications. The value is not interpreted by **UMP** in any way and has no relation to registration IDs used by the receiver. A value of 0 indicates no user set value is in use and should not be sent with acknowledgements

*Scope:* context  
*Type:* lbm\_uint\_t  
*Units:* identifier  
*Default value:* 0 (no user set value in use)  
*When to Set:* Can only be set during object initialization.

### 4.29.53. ume\_write\_delay (source)

For topics with a `repository-type` of `disk`, specifies the delay in milliseconds before the repository persists a message to disk. When using Receiver-paced Persistence, you must set this option greater than 0 (zero) but less than or equal to the repository's `write-delay` value.

*Scope:* source  
*Type:* lbm\_uint32\_t  
*Units:* milliseconds  
*Default value:* 0 (disabled)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMP 5.3

## 4.30. Ultra Messaging Queuing Options

### 4.30.1. umq\_command\_interval (context)

The interval at which all currently outstanding UMQ commands (registrations, de-registrations, message list commands, indexed queueing commands, etc.) are re-sent if they have not yet been acknowledged by the queue.

<i>Scope:</i>	context
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	500 (0.5 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.2. umq\_command\_outstanding\_maximum (context)

The maximum number of **UMQ** commands (registrations, de-registrations, message list commands, indexed queueing commands, etc.) that may be outstanding at one time for each configured queue. This option value must be greater than 0. Reducing this value may help alleviate some load on the **UMQ** queue daemon, but may potentially cause registrations and other commands to take longer to complete.

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint32_t
<i>Units:</i>	number of outstanding commands
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.3.1.

### 4.30.3. umq\_delayed\_consumption\_report\_interval (receiver)

The maximum interval to delay sending consumption reports on the receiver. Delaying consumption reports allows them to be batched together for efficiency but at the expense of delaying the consumption reports themselves individually. The value of 0 indicates the consumption reports should not be delayed.

<i>Scope:</i>	receiver
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	0
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.4. umq\_flight\_size (context)

Specifies the number of Multicast Immediate Messages allowed to be in flight (unstabilized at a queue) before a new message send either blocks or triggers a notification (source event).

<i>Scope:</i>	context
<i>Type:</i>	unsigned int
<i>Units:</i>	messages
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1.1/UME 3.1.1/UMQ 1.1.1

### 4.30.5. umq\_flight\_size (source)

Specifies the number of messages allowed to be in flight (unstabilized at a queue) before a new message send either blocks or triggers a notification (source event).

<i>Scope:</i>	source
<i>Type:</i>	unsigned int
<i>Units:</i>	messages
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1.1/UME 3.1.1/UMQ 1.1.1

### 4.30.6. umq\_flight\_size\_behavior (context)

The behavior that **UMQ** follows when a Multicast Immediate Message send exceeds the context's umq\_flight\_size.

<i>Scope:</i>	context
<i>Type:</i>	int
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1.1/UME 3.1.1/UMQ 1.1.1

String value	Integer value	Description
Block	LBM_FLIGHT_SIZE_BEHAVIOR_BLOCK	The send call blocks when a MIM send exceeds the context's flight size. If the MIM send is a non-blocking send, the send returns an LBM_EWOULD_BLOCK. <b>Default for all.</b>

String value	Integer value	Description
Notify	LBM_FLIGHT_SIZE_BEHAVIOR_NOTIFY	A message send that exceeds the configured flight size does not block but triggers a flight size notification (context event), indicating that the flight size has been surpassed. <b>UMQ</b> also sends a context event notification if the number of in-flight messages falls below the configured flight size.

### 4.30.7. umq\_flight\_size\_behavior (source)

The behavior that **UMQ** follows when a message send exceeds the source's `umq_flight_size`.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1.1/UME 3.1.1/UMQ 1.1.1

String value	Integer value	Description
Block	LBM_FLIGHT_SIZE_BEHAVIOR_BLOCK	The send call blocks when a source sends a message that exceeds its flight size. If the source uses a non-blocking send, the send returns an <b>LBM_EWOULD_BLOCK</b> . <b>Default for all.</b>
Notify	LBM_FLIGHT_SIZE_BEHAVIOR_NOTIFY	A message send that exceeds the configured flight size does not block but triggers a flight size notification (source event), indicating that the flight size has been surpassed. <b>UMQ</b> also sends a source event notification if the number of in-flight messages falls below the configured flight size.

### 4.30.8. umq\_hold\_interval (receiver)

The maximum interval to hold control and data information within the **UM** queue delivery controller.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.9. umq\_index\_assignment\_eligibility\_default (receiver)

Controls whether new receivers are immediately eligible for index assignment upon registration with a queue (the default) or whether they are ineligible upon registration and must be explicitly made eligible via a call to

`lbm_rcv_umq_index_start_assignment()`.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2/UME 3.2/UMQ 1.2

String value	Integer value	Description
Eligible	LBM_RCV_TOPIC_ATTR_UMQ_INDEX_ASSIGNMENT_ELIGIBLE	The receiver may be assigned indices as soon as it registers with a queue. <b>Default for all.</b>
Ineligible	LBM_RCV_TOPIC_ATTR_UMQ_INDEX_ASSIGNMENT_INELIGIBLE	The receiver must first call <code>lbm_rcv_umq_index_start_assignment()</code> before it can be assigned any indices.

### 4.30.10. umq\_message\_retransmission\_interval (context)

The interval between retransmissions of data messages when submitting to a Queue.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.11. umq\_message\_stability\_notification (context)

Flag indicating the context is interested in receiving notifications of message stability from Queues via the context event mechanism. Even when turned off, Queues will continue to send message stability notifications to the context for retention purposes. However, no notification will be delivered to the application.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

Value	Description
-------	-------------

Value	Description
1	The context wishes to receive message stability notification. <b>Default for all.</b>
0	The context does not wish to receive message stability notifications.

### 4.30.12. umq\_message\_stability\_notification (source)

Flag indicating the source is interested in receiving notifications of message stability from **UMQ** via the source event mechanism. Even when turned off, **UMQ** continues to send message stability notifications to the source for retention purposes. However, **UMQ** delivers no notification to the application.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

Value	Description
1	The source wishes to receive message stability notification. <b>Default for all.</b>
0	The source does not wish to receive message stability notifications.

### 4.30.13. umq\_msg\_total\_lifetime (context)

Establishes the period of time from when a queue enqueues a Multicast Immediate Message (MIM) until the time the message cannot be assigned or reassigned to a receiver. The default value of 0 (zero) disables this option. See also Message Lifetimes and Reassignment ([../UME/designing-queuing-applications.html#MESSAGE-LIFETIMES-REASSIGN](http://www.ibm.com/developer/ume/designing-queuing-applications.html#MESSAGE-LIFETIMES-REASSIGN)).

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2 / UME 3.2 / UMQ 2.1

### 4.30.14. umq\_msg\_total\_lifetime (source)

Establishes the period of time from when a queue enqueues a message until the time the message cannot be assigned



or reassigned to a receiver. The queue deletes the message upon expiration of the lifetime. You can also configure a `message-total-lifetime` for the source's topic on the queue. The queue uses whichever is shorter. The default value of 0 (zero) disables this option. See also [Message Lifetimes and Reassignment](#) ([../UME/designing-queuing-applications.html#MESSAGE-LIFETIMES-REASSIGN](#)).

*Scope:* source  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2 / UME 3.2 / UMQ 2.1

#### 4.30.15. `umq_queue_activity_timeout` (context)

The timeout value used to indicate when a queue is marked inactive. The queue must be active within this interval to be marked inactive. This value must be much larger than the check interval.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 3000 (3.0 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

#### 4.30.16. `umq_queue_check_interval` (context)

The interval between activity checks of the individual UMQ queues.

*Scope:* context  
*Type:* `lbm_ulong_t`  
*Units:* milliseconds  
*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

#### 4.30.17. `umq_queue_name` (source)

The queue to submit messages to when sending.

*Scope:* source  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

#### 4.30.18. umq\_queue\_participants\_only (source)

Flag indicating the source only desires queue participants to listen to the topic.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

Value	Description
1	The source desires that only queue participants listen to the topic.
0	The source desires anyone to listen to the topic without regard to queue participation. <b>Default for all.</b>

#### 4.30.19. umq\_queue\_participation (receiver)

Flag indicating if the receiver desires to participate in Queuing operations or not.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0. Value "2" was added in UMQ 5.2.1.

String value	Integer value	Description
"1" (Integer value as a string.)	1	The receiver desires to participate in Queuing operations. <b>Default for all.</b>
"0" (Integer value as a string.)	0	The receiver does not wish to participate in Queuing operations.

String value	Integer value	Description
"2" (Integer value as a string.)	2	The receiver desires to participate in Queuing operations as an observer receiver only. Observer receivers cannot be assigned nor consume queue messages, but can retrieve information about currently enqueued messages.

### 4.30.20. umq\_queue\_query\_interval (context)

The interval between queries sent for resolving Queues.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 200 (0.2 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.21. umq\_queue\_registration\_id (context)

This 64-bit string value assigns a registration ID to a context and associates it with a queue, using the format "qname:RegID". The RegID must be unique for each queue-context pair, however, a context can register with more than one queue (using its same RegID or a different RegID). A NULL value erases all RegIDs created via this option (required before you reassign a new RegID value to a context). NOTE: We no longer recommend setting this option, because a) if not set, UM automatically generates a unique registration ID when registering with the queue, and b) the umq\_session\_id option provides a more efficient way to manage UMQ context registration IDs.

*Scope:* context  
*Type:* lbm\_umq\_queue\_entry\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.22. umq\_receiver\_type\_id (receiver)

32-bit value that is used as an identifier to instruct the queue as to the type of receiver the receiver should be.

*Scope:* receiver  
*Type:* lbm\_uint\_t  
*Units:* identifier  
*Default value:* 0

*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.23. umq\_require\_queue\_authentication (context)

Indicates if an application requires a queue to authenticate itself before accepting the queue's responses to Queue Browser commands. See also Queue Browser ([../UME/ume-ops-operational-view.html#QUEUEBROWSER](http://../UME/ume-ops-operational-view.html#QUEUEBROWSER)).

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2.2.

Value	Description
1	An application requires the queue to successfully authenticate before using browsing command responses from the queue. <b>Default for all.</b>
0	An application does not require queue authentication.

### 4.30.24. umq\_retention\_intergroup\_stability\_behavior (context)

The behavior that the context will follow when determining the stability of a message from an inter-group perspective. This has a direct impact on the release policy for the context in that a message must be stable before it may be released. To be stable, a message must first be stable within the group and then stable between groups.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

String value	Integer value	Description
any, any-group	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_ANY	Message is considered stable once it is stable in any group. <b>Default for all.</b>
majority	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_MAJORITY	Message is considered stable once it is stable in a majority of groups.
all, all-groups	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_ALL	Message is considered stable once it is stable in all groups.

String value	Integer value	Description
all-active	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAV	Message is considered stable once it is stable in all active groups. A group is considered active if it has at least a quorum of active or registered queues. Intergroup stability requires at least one stable group.

#### 4.30.25. umq\_retention\_intergroup\_stability\_behavior (source)

The behavior that the source will follow when determining the stability of a message from an inter-group perspective. This has a direct impact on the release policy for the context in that a message must be stable before it may be released. To be stable, a message must first be stable within the group and then stable between groups.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

String value	Integer value	Description
any, any-group	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAV	Message will be considered stable once any group has reached intra-group stability for the message. <b>Default for all.</b>
majority	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAV	Message will be considered stable once a majority of groups have reached intra-group stability for the message.
all, all-groups	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAV	Message will be considered stable once all groups have reached intra-group stability for the message.
all-active	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAV	Message will be considered stable once all active groups have reached intra-group stability for the message.

#### 4.30.26. umq\_retention\_intragroup\_stability\_behavior (context)

The behavior that the context will follow when determining the stability of a message from an intra-group perspective. This has a direct impact on the release policy for the context in that a message must be stable before it may be released. To be stable, a message must first be stable within the group and then stable between groups.

*Scope:* context  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

String value	Integer value	Description
quorum	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR	Message is considered stable within the group once a quorum (or majority) of the queues have acknowledged the message as stable. <b>Default for all.</b>
all, all-stores	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR	Message is considered stable with the group once all queues have acknowledged the message as stable.
all-active	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR	Message is considered stable with the group once all active queues have acknowledged the message as stable.

### 4.30.27. umq\_retention\_intragroup\_stability\_behavior (source)

The behavior that the source will follow when determining the stability of a message from an intra-group perspective. This has a direct impact on the release policy for the context in that a message must be stable before it may be released. To be stable, a message must first be stable within the group and then stable between groups.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

String value	Integer value	Description
quorum	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR	Message will be considered stable within the group once a quorum (or majority) of the queues have acknowledged the message as stable. <b>Default for all.</b>
all, all-stores	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR	Message will be considered stable with the group once all queues have acknowledged the message as stable.
all-active	LBM_SRC_TOPIC_ATTR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR_UMQ_STABLE_BEHAVIOR	Message will be considered stable with the group once all active queues have acknowledged the message as stable.

### 4.30.28. umq\_retransmit\_request\_interval (receiver)

The interval between retransmission request messages to the queue.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds

*Default value:* 500 (0.5 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.29. `umq_retransmit_request_outstanding_maximum` (receiver)

The maximum number of messages to request at a single time from the queue. A value of 0 indicates no maximum.

*Scope:* receiver  
*Type:* `lbm_ulong_t`  
*Units:* messages  
*Default value:* 100  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.6/UME 3.0/UMQ 1.0.

### 4.30.30. `umq_session_id` (context)

Specifies the Session ID to use for managing sources and receivers within a context. A value of 0 (zero) indicates no Session ID is to be set. See also [Queue Session IDs](#) ([../UME/designing-queuing-applications.html#QUEUE-SES-IDS](#)). Valid formats for session IDs are as follows: A hexadecimal string with a maximum value of FFFFFFFFFFFFFFFF, prefixed with '0x'. An octal string with a maximum value of 177777777777777776 prefixed with '0'. A decimal string with a maximum value of 18446744073709551614.

*Scope:* context  
*Type:* `lbm_uint64_t`  
*Default value:* 0 (zero)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.3.

### 4.30.31. `umq_ulb_application_set` (source)

Application set specifications to be used by a source when using **UMQ** Ultra Load Balancing. Specifications use the format, "Index1:ID1,ID2,...;Index2:ID3,ID4,...", where "Index1" and "Index2" are the indices of the application sets and "ID1", "ID2" are receiver type IDs that belong in application set "Index1" and "ID3" and "ID4" are receiver type IDs that belong in application set "Index2". The index values used for application sets may be specified in any order, however, they must be contiguous starting with 0 when the topic is allocated. This specification format applies to the string version of this setting as well. At least one application set specification means the source will use **UMQ** Ultra Load Balancing. If no application sets are specified, the source will not use **UMQ** Ultra Load Balancing.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_receiver\_type\_entry\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.32. umq\_ulb\_application\_set\_assignment\_function (source)

The assignment function for one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

String value	Integer value	Description
default	LBM_SRC_TOPIC_ATTR_UMQ_ULB_ASSIGNMENT_DEFAULT	The default assignment function. <b>Default for all.</b>
random	LBM_SRC_TOPIC_ATTR_UMQ_ULB_ASSIGNMENT_RANDOM	Randomized assignment function.

### 4.30.33. umq\_ulb\_application\_set\_events (source)

The events mask of one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses a mask of "value1" and application set "Index2" uses a mask of "value2". The values may follow the same format as described in umq\_ulb\_events. Application sets use the value of 0 as the default event mask. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.34. umq\_ulb\_application\_set\_load\_factor\_behavior (source)

The behavior for the load factor for one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". **UMQ** uses the unsigned long int value of the entry.



*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

String value	Integer value	Description
ignored	LBM_SRC_TOPIC_ATTR_UMQ_ULB_LF_BEHAVIOR_IGNORED	Load Factor information not sent and not processed or taken into assignment consideration. <b>Default for all.</b>
provisioned	LBM_SRC_TOPIC_ATTR_UMQ_ULB_LF_BEHAVIOR_PROVISIONED	Load Factor information on number of sources sent and processed as well as taken into consideration to reduce the active portion size for each receiver.
dynamic	LBM_SRC_TOPIC_ATTR_UMQ_ULB_LF_BEHAVIOR_DYNAMIC	Load Factor information sent and processed as well as taken into consideration during assignment to weight receiver choice.

#### 4.30.35. umq\_ulb\_application\_set\_message\_lifetime (source)

The message lifetime in milliseconds of one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". A message lifetime of 0 means **UMQ** never discards the message. Application sets use the value of 0 as the default message lifetime. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

#### 4.30.36. umq\_ulb\_application\_set\_message\_max\_reassignments (source)

The maximum number of message reassignments before **UMQ** discards a message for one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". **UMQ** applies the initial assignment to this maximum. Setting this option to 1 means that the message will never be reassigned. The default value of 0 means **UMQ** never discards the message due to too many reassignments. Application sets use the value of 0 as the default. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t

*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.37. umq\_ulb\_application\_set\_message\_reassignment\_timeout (source)

The message reassignment timeout (in milliseconds) of one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". A message reassignment timeout of 0 means **UMQ** never reassigns the message. Application sets use the value of 10000 (10 seconds) as the default. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.38. umq\_ulb\_application\_set\_receiver\_activity\_timeout (source)

The receiver activity timeout (in milliseconds) of one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". Application sets use the value of 10000 (10 seconds) as the default. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.39. umq\_ulb\_application\_set\_receiver\_keepalive\_interval (source)

The interval (in milliseconds) between keepalive messages to receivers for one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". Application sets use the value of 1000 (1 seconds) as the default. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_application\_set\_attr\_t

*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

#### 4.30.40. `umq_ulb_application_set_round_robin_bias` (source)

The bias assignment towards unassigned receivers for one or more application sets specified as a list of entries in the format, "Index1:value1;Index2:value2;..." where application set "Index1" uses "value1" and application set "Index2" uses "value2". Large values increase the bias toward unassigned receivers. Zero (0) disables the bias.

*Scope:* source  
*Type:* `lbm_umq_ulb_application_set_attr_t`  
*Default value:* 1  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

#### 4.30.41. `umq_ulb_check_interval` (source)

The interval upon which **UMQ** Ultra Load Balancing sources check for message reassignment, message discards, and receiver liveness.

*Scope:* source  
*Type:* unsigned long int  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

#### 4.30.42. `umq_ulb_events` (source)

A mask indicating what **UMQ** Ultra Load Balancing events should be delivered to the source event callback. Applies to all application sets and receiver types for the source. For the configuration file as well as string versions of this option, the string value may be formatted as hexadecimal value or a list of enumerated values separated by a '!' or ','.

*Scope:* source  
*Type:* string  
*Units:* mask  
*Default value:* 0  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

String value	Integer value	Description
MSG_CONSUME, MsgConsume	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_M	Deliver message consumption events.
MSG_TIMEOUT, MsgTimeout	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_M	Deliver message timeout/discard events.
MSG_ASSIGNMENT, MsgAssignment	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_M	Deliver message assignment events.
MSG_REASSIGNMENT, MsgReassignment	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_M	Deliver message reassignment events.
MSG_COMPLETE, MsgComplete	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_M	Deliver message completion events. Messages are complete once they are consumed or discarded from all application sets.
RCV_TIMEOUT, RcvTimeout	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_R	Deliver receiver timeout events.
RCV_REGISTRATION, RcvRegistration	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_R	Deliver receiver registration events.
RCV_DEREGISTRATION, RcvDeregistration	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_R	Deliver receiver deregistration events.
RCV_READY, RcvReady	LBM_SRC_TOPIC_ATTR_UMQ_ULB_EVENT_R	Deliver receiver ready events.

#### 4.30.43. umq\_ulb\_flight\_size (source)

Specifies the number of messages allowed to be in flight (unconsumed) before a new message send either blocks or triggers a notification (source event).

<i>Scope:</i>	source
<i>Type:</i>	unsigned int
<i>Units:</i>	messages
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.1.1/UME 3.1.1/UMQ 1.1.1

#### 4.30.44. umq\_ulb\_flight\_size\_behavior (source)

The behavior that **UMQ** follows when a message send exceeds the source's ulb\_flight\_size.

*Scope:* source  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1.1/UME 3.1.1/UMQ 1.1.1

String value	Integer value	Description
Block	LBM_FLIGHT_SIZE_BEHAVIOR_BLOCK	The send call blocks when a message send exceeds the source's flight size. If the message send is a non-blocking send, the send returns an LBM_EWOULD_BLOCK. <b>Default for all.</b>
Notify	LBM_FLIGHT_SIZE_BEHAVIOR_NOTIFY	A message send that exceeds the configured flight size does not block but triggers a flight size notification (source event), indicating that the flight size has been surpassed. <b>UMQ</b> also sends a source event notification if the number of in-flight messages falls below the configured flight size.

#### 4.30.45. umq\_ulb\_receiver\_events (source)

Set the events mask of one or more receiver types specified as a list of entries in the format, "ID1:value1;ID2:value2;..." where receiver type "ID1" uses a mask of "value1" and receiver type "ID2" uses a mask of "value2". The values may follow the same format as described in umq\_ulb\_events. Receiver types use the value of 0 as the default event mask. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_receiver\_type\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

#### 4.30.46. umq\_ulb\_receiver\_portion (source)

The portion size of one or more receiver types specified as a list of entries in the format, "ID1:value1;ID2:value2;..." where receiver type "ID1" uses a portion size of "value1" and receiver type ID2 uses a portion size of "value2". Receiver types use the value of 1 as the default portion size. **UMQ** uses the unsigned long int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_receiver\_type\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.47. umq\_ulb\_receiver\_priority (source)

The priority of one or more receiver types specified as a list of entries in the format, "ID1:value1;ID2:value2;..." where receiver type "ID1" uses a priority of "value1" and receiver type "ID2" uses a priority of "value2". Receiver types use the value of 0 as the default priority. **UMQ** uses the int value of the entry.

*Scope:* source  
*Type:* lbm\_umq\_ulb\_receiver\_type\_attr\_t  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.1/UME 3.1/UMQ 1.1.

### 4.30.48. umq\_ulb\_source\_activity\_timeout (receiver)

The timeout value used to indicate when a ULB source is unresponsive. The ULB source must not be active within this interval to be considered unresponsive. This value must be much larger than the source check interval.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 10000 (10 seconds)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2 / UME 3.2 / UMQ 2.1

### 4.30.49. umq\_ulb\_source\_check\_interval (receiver)

The interval between activity checks of a ULB source. Allow a ULB receiver to proactively attempt re-registration with a ULB source if the receiver has not seen any activity (including keepalives) from that source in a specified amount of time, provided the source's transport session is still alive and valid.

*Scope:* receiver  
*Type:* lbm\_ulong\_t  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2 / UME 3.2 / UMQ 2.1

## 4.31. Ultra Messaging JMS Options

### 4.31.1. client\_id (ConnectionFactory)

Unique identifier for a JMS Client.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

### 4.31.2. context\_index (ConnectionFactory)

Optional. Normally a new Connection creates a new **UM** context, which means a ConnectionFactory using multiple Connections has that many corresponding contexts. This is the behavior if this option is NULL or simply not used. To force this ConnectionFactory to use the same context for all Connections it creates, set this string value to a desired identifier for the context.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

### 4.31.3. create\_queue\_browser\_context (ConnectionFactory)

Create a secondary context per connection for queue browsing. If not using JMS queue browsing in the JMS application, set this option to `false`. Not creating the the queue browser context can improve application performance.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
True	True	Create a secondary context for queue browsing. <b>Default for Unix.</b>
False	False	Do not create a secondary, queue browsing context.

### 4.31.4. debug (ConnectionFactory)

Turns debugging on or off. Setting this to `false` has the same effect as setting `com.latencybusters.jms.level = INFO` in `jmsclient/bin/logging.properties`.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
True	True	True
False	False	False <b>Default for Unix.</b>

### 4.31.5. default\_message\_type (ConnectionFactory)

Determines what type of message Ultra Messaging JMS creates when it receives a message without an App Header. The message could be from a non-JMS source or a JMS application that has App Headers turned off (`<Attribute name="USE_APP_HEADER" value="false"/>`). Possible values are shown in the table below.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
TextMessage	TextMessage	TextMessage
BytesMessage	BytesMessage	BytesMessage <b>Default for Unix.</b>
MapMessage	MapMessage	MapMessage
StreamMessage	StreamMessage	StreamMessage
ObjectMessage	ObjectMessage	ObjectMessage

### 4.31.6. default\_temp\_topic\_type (ConnectionFactory)

Sets the default destination for `Session.createTemporaryTopic(topicName)`.

*Scope:* ConnectionFactory  
*Type:* string



*When to Set:* Can only be set during object initialization.

*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
UME	UME	Used as the default by topics.
LBM	LBM	Setting this option to LBM increases performance. <b>Default for Unix.</b>

### 4.31.7. default\_topic\_type (ConnectionFactory)

Sets the default destination for `Session.createTopic(topicName)`.

*Scope:* ConnectionFactory

*Type:* string

*When to Set:* Can only be set during object initialization.

*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
UME	UME	Used as the default by topics.
LBM	LBM	Set to LBM to create a topic that uses basic streaming. <b>Default for Unix.</b>

### 4.31.8. DestType (Destination)

The DestType attribute determines which JMS destination type should be created.

*Scope:* Destination

*Type:* string

*When to Set:* Can only be set during object initialization.

*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
Queue	Queue	Queue
Topic	Topic	Topic <b>Default for Unix.</b>
Destination	Destination	Destination <b>Default for Unix.</b>

### 4.31.9. queue\_browser\_creation\_delay (ConnectionFactory)

When creating a JMS QueueBrowser, the delay to allow the underlying queue browser's context and receiver to register with the queue. Increasing this value may help if an exception returns from the call to `getEnumeration` indicating that the underlying receiver was not yet registered with the queue.

<i>Scope:</i>	ConnectionFactory
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	500
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2.

### 4.31.10. queue\_browser\_timeout (ConnectionFactory)

The length of time (in milliseconds) to wait before giving up and returning after calling the JMS QueueBrowser's `getEnumeration` method. A value of 0 (zero) disables the timeout, allowing the method to wait an unlimited amount of time.

<i>Scope:</i>	ConnectionFactory
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	10000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2.

### 4.31.11. RECEIVER\_CREATION\_DELAY (ConnectionFactory)

When creating a consumer in JMS, the amount of additional time to wait after creating the underlying `LBMSReceiver`. A value of 0 (zero) indicates no delay. When setting this option in a configuration file, it **MUST** be uppercase.

<i>Scope:</i>	ConnectionFactory
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	0
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2.

### 4.31.12. regid (Destination)

The Registration ID assigned to the source sending to the Destination. Required for **UMP**.

<i>Scope:</i>	Destination
<i>Type:</i>	int
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2

### 4.31.13. SOURCE\_CREATION\_DELAY (ConnectionFactory)

When creating a producer in JMS, the amount of additional time to wait after creating the underlying `LBMSource`. A value of 0 (zero) indicates no delay. When setting this option in a configuration file, it **MUST** be uppercase.

<i>Scope:</i>	ConnectionFactory
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	1000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2.

### 4.31.14. SOURCE\_REGISTRATION\_TIMEOUT (ConnectionFactory)

When `wait_for_source_registration` is set to true, the length of time (in milliseconds) to wait before throwing an exception to indicate the source could not register with the store or queue. A value of 0 (zero) disables the timeout, indicating the source should wait an unlimited amount of time. When setting this option in a configuration file, it **MUST** be uppercase.

<i>Scope:</i>	ConnectionFactory
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	milliseconds
<i>Default value:</i>	30000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2.

### 4.31.15. Topic (Destination)

Sets the Topic Name. The maximum length for this string is 246 characters.

<i>Scope:</i>	Destination
---------------	-------------

*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

#### 4.31.16. Type (Destination)

Determines the type of destination. Overrides the value set for the ConnectionFactory option, `default_topic_type` for this Destination only. Queueing applications would use UMQ. Topics would use UME or LBM.

*Scope:* Destination  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
UME	UME	Set this option to UME for topics in <b>UMP</b> applications.
LBM	LBM	Set this option to LBM topics in <b>UMS</b> applications. <b>Default for Unix.</b>
UMQ	UMQ	Set this option to UMQ for <b>UMP</b> applications <b>Default for Unix.</b>

#### 4.31.17. use\_app\_header (ConnectionFactory)

When set to `true`, this option enables Ultra Messaging JMS to transmit JMS message header and JMS message properties information (including message selectors) via a **UM** Message Properties object. See the Oracle JMS Specification for information on JMS message header fields and message properties. Setting this option to `false` causes Ultra Messaging JMS to use the default message type, and can greatly increase performance.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
True	True	True
False	False	False <b>Default for Unix.</b>

### 4.31.18. use\_index\_queueing (ConnectionFactory)

Allows you to turn on index queueing which can be used to simulate the behaviour of the `JMSXGroupID` flag in a JMS broker. UM JMS sends all of the message for a particular transaction to the same consumer. **UMQ** load balances the groups of messages across consumers.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
True	True	True
False	False	False <b>Default for Unix.</b>

### 4.31.19. use\_ump\_session\_ids (ConnectionFactory)

Use this option (with **UMP**) to enable the use of **UMP** session IDs. If you use this option, do not set `client_id`. For more information, see **UMP** Session IDs in the UM JMS Guide.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.3

String value	Integer value	Description
True	True	True
False	False	False <b>Default for Unix.</b>

### 4.31.20. wait\_for\_source\_registration (ConnectionFactory)

When creating a producer in JMS, the value that controls whether or not to wait for the underlying `LBMSource` to complete registration with the store or queue. If set to false, then it becomes the responsibility of the application to handle exceptions that occur during sending due to the source not yet being registered.

*Scope:* ConnectionFactory  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMQ 5.2

String value	Integer value	Description
True	True	Wait for the underlying <code>LBMSource</code> to complete registration with the store or queue. <b>Default for Unix.</b>
False	False	Do not wait for the underlying <code>LBMSource</code> to complete registration.

### 4.31.21. Wildcard (Destination)

Indicates if the topic name is a wildcard pattern.

<i>Scope:</i>	Destination
<i>Type:</i>	string
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in UMQ 5.2

String value	Integer value	Description
True	True	True
False	False	False <b>Default for Unix.</b>

## 4.32. Hot Failover Operation Options

Hot Failover (HF) allows your applications to build in sender redundancy. See Hot Failover ([../Design/lbm-features.html#HOT-FAILOVER](#)) for a discussion of using Hot Failover within a single receiver context or across multiple receiver contexts.

### 4.32.1. delivery\_control\_loss\_check\_interval (hfx)

The interval between periodic forced loss checks. This option defaults to 0, indicating that loss checks should only be made when a new message arrives.

<i>Scope:</i>	hfx
<i>Type:</i>	lbm_ulong_t
<i>Units:</i>	msec
<i>Default value:</i>	0 (no periodic loss checks)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.2

### 4.32.2. `delivery_control_max_delay` (hfx)

The minimum interval that must expire before the HFX Receiver declares a message unrecoverable and delivers an unrecoverable loss message the application. By default, the HFX Receiver only checks loss when it receives new messages. To enable periodic loss checks, set the `delivery_control_loss_check_interval` option.

<i>Scope:</i>	hfx
<i>Type:</i>	<code>lbm_ulong_t</code>
<i>Units:</i>	msec
<i>Default value:</i>	10000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.2

### 4.32.3. `delivery_control_maximum_burst_loss` (hfx)

Specifies the largest permissible gap between the next expected message and the most recently received message. When the difference in sequence numbers between the most recently received message and the next expected message exceeds this amount, the HFX Receiver delivers a burst loss notification. The HFX Receiver discards any messages currently pending delivery. Normal delivery resumes with the most recently received message.

<i>Scope:</i>	hfx
<i>Type:</i>	<code>lbm_uint_t</code>
<i>Units:</i>	number of messages
<i>Default value:</i>	512
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.2

### 4.32.4. `delivery_control_maximum_total_map_entries` (hfx)

The maximum number of map entries for the HFX order and loss maps. This is a soft limit. When the sum of the number of loss records and the number of messages held for ordering (messages that will be delivered once all prior messages have been delivered) is greater than this value, the oldest consecutive sequence of loss records will be declared lost immediately to reduce the number of outstanding map entries. A value of 0 indicates that the map should be allowed to grow without bound.

<i>Scope:</i>	hfx
<i>Type:</i>	<code>size_t</code>
<i>Units:</i>	map entries
<i>Default value:</i>	200000
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 4.2

### 4.32.5. duplicate\_delivery (hfx)

Flag indicating whether duplicate messages should be discarded or simply marked as duplicates. Setting this to 1 overrides the hf\_duplicate\_delivery setting on all underlying HFX Receivers.

*Scope:* hfx  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2

Value	Description
1	The HFX delivers duplicate messages.
0	The HFX does not deliver duplicate messages. <b>Default for all.</b>

### 4.32.6. hf\_duplicate\_delivery (receiver)

Flag indicating if the Hot Failover receiver delivers duplicate messages or not.

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.

Value	Description
1	The Hot Failover receiver delivers duplicate messages.
0	The Hot Failover receiver does not deliver duplicate messages. <b>Default for all.</b>

### 4.32.7. hf\_optional\_messages (receiver)

Indicates if a Hot Failover receiver can receive optional messages. See also Hot Failover Optional Messages ([../Design/lbm-features.html#OPTIONAL-HF-MESSAGES](#)).

*Scope:* receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2.5/UME 3.2.5/UMQ 2.1.5

Value	Description
-------	-------------



Value	Description
1	Hot Failover receivers can receive optional messages. <b>Default for all.</b>
0	Hot Failover receivers do not receive optional messages.

#### 4.32.8. hf\_receiver (wildcard\_receiver)

Specifies whether to create hot failover receivers for each topic that maps to the wildcard receiver pattern.

*Scope:* wildcard\_receiver  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in UMS 5.2.2

Value	Description
1	Create hot failover receivers for each matched topic.
0	Normal wildcard receiver operation. Hot failover sequence numbers are ignored. <b>Default for all.</b>

#### 4.32.9. ordered\_delivery (hfx)

Flag indicating if the HFX Receiver orders messages before delivery.

*Scope:* hfx  
*Type:* int  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 4.2

String value	Integer value	Description
"1" (Integer value as a string.)	1	The HFX Receiver delivers messages in order. <b>Default for all.</b>
"-1" (Integer value as a string.)	-1	The HFX Receiver delivers messages as soon as they are received. In the case of fragmented messages, as soon as all fragments have been received and reassembled.

## 4.33. Automatic Monitoring Options

The Monitoring Options below apply to a given **UMS** context. You can override the default values of these options and apply monitoring option values to all **UMS** contexts (transports and event queues) with the following environment variables.

- LBM\_MONITOR\_INTERVAL
- LBM\_MONITOR\_TRANSPORT
- LBM\_MONITOR\_TRANSPORT\_OPTS
- LBM\_MONITOR\_APPID

These variables will not override any Monitoring Options you specifically set. The environment variables only override Monitoring Options default values.

If you do not specify any monitoring options either in an **UMS** configuration file or via `lbm_context_attr_setopt()` calls, no monitoring will occur. However, if you then set the `LBM_MONITOR_INTERVAL` environment variable to 5, you will turn on automatic monitoring for every **UMS** context your application creates at 5 second intervals. If you then set `monitor_interval` to 10 for a particular context, all transport sessions in that context will be monitored every 10 seconds.

See also Automatic Monitoring ([../Design/monitoring.html#AUTOMONITOR](#)) for more information about this feature.

### 4.33.1. monitor\_appid (context)

An application ID string used by automatic monitoring to identify the application generating the statistics.

<i>Scope:</i>	context
<i>Type:</i>	string
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.4/UME 2.1.

### 4.33.2. monitor\_appid (event\_queue)

An application ID string used by automatic monitoring to identify the application generating the statistics.

<i>Scope:</i>	event_queue
<i>Type:</i>	string
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.4/UME 2.1.

### 4.33.3. monitor\_interval (context)

Interval at which automatic monitoring retrieves the statistics for all transport sessions on a context. Setting this option to zero (the default) disables the automatic monitoring of a context's transport sessions.

*Scope:* context  
*Type:* lbm\_ulong\_t  
*Units:* seconds  
*Default value:* 0  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

### 4.33.4. monitor\_interval (event\_queue)

Interval at which automatic monitoring retrieves the statistics for an event queue. Setting this option to zero (the default) disables the automatic monitoring of an event queue. When monitoring Event Queue statistics you must enable the Event Queue **UM** Configuration Options, `queue_age_enabled` (`../Config/eventqueueoptions.html#EVENTQUEUEQUEUEAGEENABLED`), `queue_count_enabled` (`../Config/eventqueueoptions.html#EVENTQUEUEQUEUEECOUNTENABLED`) and `queue_service_time_enabled` (`../Config/eventqueueoptions.html#EVENTQUEUEQUEUESERVICETIMEENABLED`). **UM** disables these options by default, which produces no event queue statistics.

*Scope:* event\_queue  
*Type:* lbm\_ulong\_t  
*Units:* seconds  
*Default value:* 0  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

### 4.33.5. monitor\_transport (context)

The LBMMON transport module to be used for automatic monitoring.

*Scope:* context  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

String value	Integer value	Description
--------------	---------------	-------------

String value	Integer value	Description
lbm	LBM_CTX_ATTR_MON_TRANSPORT_LBM	Use the LBMMON lbm transport module. <b>Default for all.</b>
lbmsnmp	LBM_CTX_ATTR_MON_TRANSPORT_LBMSNMP	Use the LBMMON lbmsnmp transport module. This value is required if you use the <b>Ultra Messaging SNMP Agent</b> .

#### 4.33.6. monitor\_transport (event\_queue)

The LBMMON transport module to be used for automatic monitoring.

*Scope:* event\_queue  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

String value	Integer value	Description
lbm	LBM_CTX_ATTR_MON_TRANSPORT_LBM	Use the LBMMON lbm transport module. <b>Default for all.</b>
lbmsnmp	LBM_CTX_ATTR_MON_TRANSPORT_LBMSNMP	Use the LBMMON lbmsnmp transport module. This value is required if you use the <b>Ultra Messaging SNMP Agent</b> .

#### 4.33.7. monitor\_transport\_opts (context)

An option string to be passed to the LBMMON transport module for automatic monitoring. See The **UM Transport Module** (../Design/monitoring.html#LBM-TRANSPORT-MODULE) for more information about Transport Options. (Options for the lbm transport module and the lbmsnmp transport module are identical.)

*Scope:* context  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

#### 4.33.8. monitor\_transport\_opts (event\_queue)

An option string to be passed to the LBMMON transport module for automatic monitoring. See The **UM Transport Module** (../Design/monitoring.html#LBM-TRANSPORT-MODULE) for more information about Transport Options. (Options for the lbm transport module and the lbmsnmp transport module are identical.)

*Scope:* event\_queue  
*Type:* string  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was implemented in LBM 3.4/UME 2.1.

## 4.34. Deprecated Options

### 4.34.1. resolver\_active\_source\_interval (context)

Interval between sending Topic Resolution advertisements for active sources. A value of 0 indicates that periodic advertisements should not be sent (sources will still respond to queries). NOTE: when set to 0, the `resolver_active_threshold` should typically also be set to 0. See also [Disabling Aspects of Topic Resolution \(disable-topic-res.html\)](#). NOTE: Although this option is eligible to be set during operation, two considerations exist.

- If this option is disabled at initialization (set to 0), you cannot re-set the option during operation.
- Disabling this option by setting it to 0 (zero) during operation prevents you from re-setting the option a second time during operation.

*Scope:* context  
*Type:* unsigned long int  
*Units:* milliseconds  
*Default value:* 1000 (1 second)  
*When to Set:* May be set during operation.  
*Version:* This option was deprecated in LBM 4.0

### 4.34.2. resolver\_active\_threshold (context)

Number of seconds since the last application message was sent to a source that causes that source to be marked inactive. Inactive sources are not advertised periodically (but will continue to respond to queries). A value of 0 indicates that sources will advertise periodically regardless of how often the application sends messages. Note that for publishers with large numbers of sources, this can increase the topic resolution traffic load. However, also note that this option **SHOULD** be set to 0 if periodic advertisements are disabled (by setting `resolver_active_source_interval` to 0). See also [Disabling Aspects of Topic Resolution \(disable-topic-res.html\)](#) and [Interrelated Configuration Options \(../Config/grouprelated.html\)](#).

*Scope:* context  
*Type:* unsigned long int

<i>Units:</i>	seconds
<i>Default value:</i>	60
<i>When to Set:</i>	May be set during operation.
<i>Version:</i>	This option was deprecated in LBM 4.0

### 4.34.3. resolver\_maximum\_advertisements (context)

Maximum number of topics that will be advertised per active source interval. A value of 0 means to advertise all topics.

<i>Scope:</i>	context
<i>Type:</i>	unsigned long int
<i>Units:</i>	Number of topics
<i>Default value:</i>	0 (all topics)
<i>When to Set:</i>	May be set during operation.
<i>Version:</i>	This option was deprecated in LBM 4.0

### 4.34.4. resolver\_maximum\_queries (context)

Maximum number of topics that will be queried for per query interval. A value of 0 means to query for all topics that do not have at least one source.

<i>Scope:</i>	context
<i>Type:</i>	unsigned long int
<i>Units:</i>	Number of topics
<i>Default value:</i>	0 (all topics with no source)
<i>When to Set:</i>	May be set during operation.
<i>Version:</i>	This option was deprecated in LBM 4.0

### 4.34.5. resolver\_query\_interval (context)

Interval between query transmissions for receivers attempting Topic Resolution. A value of 0 indicates queries should not be sent. See also Disabling Aspects of Topic Resolution ([disable-topic-res.html](#)). NOTE: Although this option is eligible to be set during operation, two considerations exist.

- If this option is disabled at initialization (set to 0), you cannot re-set the option during operation.
- Disabling this option by setting it to 0 (zero) during operation prevents you from re-setting the option a second time during operation.

<i>Scope:</i>	context
<i>Type:</i>	unsigned long int
<i>Units:</i>	milliseconds
<i>Default value:</i>	100 (0.1 seconds)
<i>When to Set:</i>	May be set during operation.
<i>Version:</i>	This option was deprecated in LBM 4.0

#### 4.34.6. resolver\_query\_max\_interval (wildcard\_receiver)

This sets the maximum interval between wildcard queries in topic resolution (when used). Only PCRE and regex pattern types can use wildcard queries. A value of 0 indicates wildcard queries should not be sent. **UM** currently queries a maximum of 250 unique wildcard patterns (receivers). NOTE: Although this option is eligible to be set during operation, two considerations exist.

- If this option is disabled at initialization (set to 0), you cannot re-set the option during operation.
- Disabling this option by setting it to 0 (zero) during operation prevents you from re-setting the option a second time during operation.

<i>Scope:</i>	wildcard_receiver
<i>Type:</i>	unsigned long int
<i>Units:</i>	milliseconds
<i>Default value:</i>	0 (do not query)
<i>When to Set:</i>	May be set during operation.
<i>Version:</i>	This option was deprecated in LBM 4.0

#### 4.34.7. resolver\_unicast\_address (context)

The IP address to send unicast topic resolution messages to. If set to 0.0.0.0 (`INADDR_ANY`), then topic resolution uses multicast (the default). If set to anything else, then topic resolution messages go to the IP address specified.

<i>Scope:</i>	context
<i>Type:</i>	struct in_addr
<i>Default value:</i>	0.0.0.0 ( <code>INADDR_ANY</code> )
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UMS 5.0. See <code>resolver_unicast_daemon</code> instead.

### 4.34.8. resolver\_unicast\_destination\_port (context)

The UDP port to send unicast topic resolution messages to. This is the UDP port used by the **UM** resolution daemon (lbmrdr).

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint16_t
<i>Default value:</i>	15380
<i>Byte order:</i>	Network
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UMS 5.0. See resolver_unicast_daemon instead.

### 4.34.9. resolver\_unicast\_port (context)

The local UDP port used for unicast topic resolution messages. The **UM** resolution daemon (lbmrdr) will send unicast topic resolution messages to this UDP port. A value of 0 indicates that **UM** should pick an open port in the range (resolver\_unicast\_port\_low, resolver\_unicast\_port\_high).

<i>Scope:</i>	context
<i>Type:</i>	lbm_uint16_t
<i>Default value:</i>	0 (pick open port)
<i>Byte order:</i>	Network
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UMS 5.0. See resolver_unicast_daemon instead.

### 4.34.10. retransmit\_message\_map\_tablesz (source)

The size of the hash table that the source uses to store messages for the retention policy in effect. A larger table means more messages can be stored more efficiently, but takes up more memory. A smaller table uses less memory, but costs more CPU time as more messages are retained. See [Configuring Late Join for Large Numbers of Messages](#) (../Design/lbm-features.html#USING-LATE-JOIN-LARGENUMBERS) for additional information about this option.

<i>Scope:</i>	source
<i>Type:</i>	size_t
<i>Default value:</i>	131
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option has been deprecated.



### 4.34.11. transport\_datagram\_max\_size (context)

The maximum datagram size that can be generated by UM. The default value is 8192, the minimum is 400 bytes, and the maximum is 65535.

<i>Scope:</i>	context
<i>Type:</i>	unsigned int
<i>Units:</i>	bytes
<i>Default value:</i>	8192
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was implemented in LBM 3.3.5/UME 2.0.3.
<i>Version:</i>	This option was deprecated in LBM 4.1

### 4.34.12. transport\_lbtipc\_acknowledgement\_interval (receiver)

Period of time between acknowledgement (keepalive) messages sent from the receiver to the IPC source. See also transport\_lbtipc\_client\_activity\_timeout. Refer to Receiver Configuration and Transport Sessions (../Design/lbm-objects.html#RECEIVER-OBJECT) and Interrelated Configuration Options (../Config/grouprelated.html) for additional information.

<i>Scope:</i>	receiver
<i>Type:</i>	unsigned long int
<i>Units:</i>	milliseconds
<i>Default value:</i>	500 (0.5 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in LBM 4.0

### 4.34.13. transport\_lbtipc\_client\_activity\_timeout (source)

The maximum period of inactivity (lack of acknowledgement keepalive messages) from a receiver before the source deletes the receiver from it's active receiver table. The IPC source signals all receivers in it's active receiver's table when it writes new data to the shared memory area. See also transport\_lbtipc\_acknowledgement\_interval. Refer to Source Configuration and Transport Sessions (../Design/lbm-objects.html#SOURCE-OBJECT) and Interrelated Configuration Options (../Config/grouprelated.html) for additional information.

<i>Scope:</i>	source
<i>Type:</i>	unsigned long int
<i>Units:</i>	milliseconds
<i>Default value:</i>	10,000 (10 seconds)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in LBM 4.0

#### 4.34.14. `ume_message_map_tablesz` (source)

The size of the hash table that the source uses to store messages for the retention policy in effect. A larger table means more messages can be stored more efficiently, but takes up more memory. A smaller table uses less memory, but costs more CPU time as more messages are retained. This setting no longer has any effect.

<i>Scope:</i>	source
<i>Type:</i>	size_t
<i>Default value:</i>	131
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option has been deprecated.

#### 4.34.15. `ume_primary_store_address` (source)

IPv4 address of the persistent store to be used as the primary store. A value of 0.0.0.0 (or `INADDR_ANY`) indicates no store is set as the primary. In other words, persistence is not enabled for the source. This setting is deprecated. Its use is not recommended except by legacy systems. Please use the `ume_store` option instead.

<i>Scope:</i>	source
<i>Type:</i>	struct in_addr
<i>Default value:</i>	0.0.0.0 ( <code>INADDR_ANY</code> )
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UME 2.0

#### 4.34.16. `ume_primary_store_port` (source)

TCP port of the primary persistent store. This setting is deprecated. Its use is not recommended except by legacy systems. Please use the `ume_store` option instead.

<i>Scope:</i>	source
<i>Type:</i>	lbm_uint16_t
<i>Default value:</i>	14567
<i>Byte order:</i>	Network
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UME 2.0

#### 4.34.17. `ume_registration_id` (source)

32-bit value that is used by a persistent store to identify a source. If a source desires to identify itself as a previously known source (after a crash or shutdown), it should set the ID to the value it was using before. A value of 0 indicates the source will allow the persistent store to assign an ID. This setting is deprecated. Its use is not recommended

except by legacy systems. Please use the `ume_store` option instead.

<i>Scope:</i>	source
<i>Type:</i>	<code>lbm_uint_t</code>
<i>Units:</i>	identifier
<i>Default value:</i>	0 (allow persistent store to assign ID)
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UME 2.0

#### 4.34.18. `ume_secondary_store_address` (source)

IPv4 address of the persistent store to be used as the secondary store. A value of 0.0.0.0 (or `INADDR_ANY`) indicates no store is set as the secondary. This setting is deprecated. Its use is not recommended except by legacy systems. Please use the `ume_store` option instead.

<i>Scope:</i>	source
<i>Type:</i>	<code>struct in_addr</code>
<i>Default value:</i>	0.0.0.0 ( <code>INADDR_ANY</code> )
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UME 2.0

#### 4.34.19. `ume_secondary_store_port` (source)

TCP port of the secondary persistent store. This setting is deprecated. Its use is not recommended except by legacy systems. Please use the `ume_store` option instead.

<i>Scope:</i>	source
<i>Type:</i>	<code>lbm_uint16_t</code>
<i>Default value:</i>	14567
<i>Byte order:</i>	Network
<i>When to Set:</i>	Can only be set during object initialization.
<i>Version:</i>	This option was deprecated in UME 2.0

#### 4.34.20. `ume_tertiary_store_address` (source)

IPv4 address of the persistent store to be used as the tertiary store. A value of 0.0.0.0 (or `INADDR_ANY`) indicates no store is set as the tertiary. This setting is deprecated. Its use is not recommended except by legacy systems. Please use the `ume_store` option instead.

<i>Scope:</i>	source
---------------	--------

*Type:* struct in\_addr  
*Default value:* 0.0.0.0 (INADDR\_ANY)  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was deprecated in UME 2.0

### 4.34.21. ume\_tertiary\_store\_port (source)

TCP port of the tertiary persistent store. This setting is deprecated. Its use is not recommended except by legacy systems. Please use the ume\_store option instead.

*Scope:* source  
*Type:* lbm\_uint16\_t  
*Default value:* 14567  
*Byte order:* Network  
*When to Set:* Can only be set during object initialization.  
*Version:* This option was deprecated in UME 2.0

## 4.35. UMS Port Values

This section lists the default port values use by UMS.

### 4.35.1. UMS UDP Port Values

**Table 4-1. Default UMS UDP Port Values**

Configuration Option	Scope	Default Value
mim_destination_port	context	14401
mim_incoming_destination_port	context	14401
mim_outgoing_destination_port	context	14401
resolver_multicast_incoming_port	context	12965
resolver_multicast_outgoing_port	context	12965
resolver_multicast_port	context	12965
resolver_unicast_destination_port	context	15380
resolver_unicast_port	context	0 (pick open port)
resolver_unicast_port_high	context	14406
resolver_unicast_port_low	context	14402
transport_lbtrm_destination_port	source	14400

Configuration Option	Scope	Default Value
transport_lbtrm_source_port_high	context	14399
transport_lbtrm_source_port_low	context	14390
transport_lbtru_maximum_ports	context	5
transport_lbtru_port	source	0 (use open port)
transport_lbtru_port_high	context	14389
transport_lbtru_port_high	receiver	14379
transport_lbtru_port_low	context	14380
transport_lbtru_port_low	receiver	14360

### 4.35.2. UMS TCP Port Values

**Table 4-2. Default UMS TCP Port Values**

Configuration Option	Scope	Default Value
request_tcp_port	context	0 (use open port)
request_tcp_port_high	context	14395
request_tcp_port_low	context	14391
transport_tcp_maximum_ports	context	10
transport_tcp_port	source	0 (pick open port)
transport_tcp_port_high	context	14390
transport_tcp_port_low	context	14371
ume_primary_store_port	source	14567
ume_secondary_store_port	source	14567
ume_tertiary_store_port	source	14567

### 4.36. UMS Multicast Group Values

This section lists the default multicast group values use by UMS.

**Table 4-3. Default UMS Multicast Group Values**

Configuration Option	Scope	Default Value
mim_address	context	224.10.10.21
mim_incoming_address	context	224.10.10.21
mim_outgoing_address	context	224.10.10.21
resolver_multicast_address	context	224.9.10.11

Configuration Option	Scope	Default Value
resolver_multicast_incoming_address	context	224.9.10.11
resolver_multicast_outgoing_address	context	224.9.10.11
transport_lbtrm_multicast_address	source	0.0.0.0 (INADDR_ANY)
transport_lbtrm_multicast_address_high	context	224.10.10.14
transport_lbtrm_multicast_address_low	context	224.10.10.10

## 4.37. UMS Timer Interval Values

This section lists the default timer interval values use by UMS. All values are in milliseconds.

**Table 4-4. Default UMS Timer Interval Values**

Configuration Option	Scope	Default Value
delivery_control_loss_check_interval	receiver	0 (disabled)
implicit_batching_interval	source	200 (0.2 seconds)
mim_activity_timeout	context	60000 (60 seconds)
mim_delivery_control_loss_check_interval	context	0 (disabled)
mim_ignore_interval	context	500 (0.5 seconds)
mim_implicit_batching_interval	context	200 (0.2 seconds)
mim_nak_backoff_interval	context	200 (0.2 seconds)
mim_nak_generation_interval	context	10000 (10 seconds)
mim_nak_initial_backoff_interval	context	50 (0.05 seconds)
mim_nak_suppress_interval	context	1000 (1 second)
mim_sm_maximum_interval	context	10000 (10 seconds)
mim_sm_minimum_interval	context	200 (0.2 seconds)
mim_src_deletion_timeout	context	30000 (30 seconds)
rcv_sync_cache_timeout	receiver	2000 (2 seconds)
resolver_active_source_interval	context	1000 (1 second)
resolver_advertisement_maximum_initial_interval	source	500 (0.5 seconds)
resolver_advertisement_minimum_initial_duration	source	5000 (5 seconds)
resolver_advertisement_minimum_initial_interval	source	10 (0.01 seconds)

Configuration Option	Scope	Default Value
resolver_advertisement_minimum_sustain_duration	source	60 (1 minute)
resolver_advertisement_sustain_interval	source	1000 (1 second)
resolver_context_advertisement_interval	context	10000 (10 seconds)
resolver_no_source_linger_timeout	wildcard_receiver	1000 (1 second)
resolver_query_interval	context	100 (0.1 seconds)
resolver_query_max_interval	wildcard_receiver	0 (do not query)
resolver_query_maximum_initial_interval	receiver	200 (0.2 seconds)
resolver_query_maximum_interval	wildcard_receiver	1000 (1 second)
resolver_query_minimum_duration	wildcard_receiver	60 (1 minute)
resolver_query_minimum_initial_duration	receiver	5000 (5 seconds)
resolver_query_minimum_initial_interval	receiver	20 (0.02 seconds)
resolver_query_minimum_interval	wildcard_receiver	50 (0.05 seconds)
resolver_query_minimum_sustain_duration	receiver	60 (1 minute)
resolver_query_sustain_interval	receiver	1000 (1 second)
response_tcp_deletion_timeout	context	2000 (2 seconds)
retransmit_request_generation_interval	receiver	10000 (10 seconds)
retransmit_request_interval	receiver	500 (0.5 seconds)
transport_lbtipc_acknowledgement_interval	receiver	500 (0.5 seconds)
transport_lbtipc_activity_timeout	receiver	60,000 (60 seconds)
transport_lbtipc_client_activity_timeout	source	10,000 (10 seconds)
transport_lbtipc_sm_interval	source	10,000 (10 seconds)
transport_lbtrm_activity_timeout	receiver	60000 (60 seconds)
transport_lbtrm_ignore_interval	source	500 (0.5 seconds)
transport_lbtrm_nak_backoff_interval	receiver	200 (0.2 seconds)
transport_lbtrm_nak_generation_interval	receiver	10000 (10 seconds)
transport_lbtrm_nak_initial_backoff_interval	receiver	50 (0.05 seconds)
transport_lbtrm_nak_suppress_interval	receiver	1000 (1 second)

Configuration Option	Scope	Default Value
transport_lbtrm_preactivity_timeout	receiver	0 (zero)
transport_lbtrm_rate_interval	context	100
transport_lbtrm_sm_maximum_interval	source	10000 (10 seconds)
transport_lbtrm_sm_minimum_interval	source	200 (0.2 seconds)
transport_lbtru_acknowledgement_interval	receiver	500 (0.5 seconds)
transport_lbtru_activity_timeout	receiver	60000 (60 seconds)
transport_lbtru_client_activity_timeout	source	10000 (10 seconds)
transport_lbtru_connect_interval	receiver	100 (0.1 seconds)
transport_lbtru_ignore_interval	source	500 (0.5 seconds)
transport_lbtru_nak_backoff_interval	receiver	200 [100,300] (0.2 [0.1,0.3] seconds)
transport_lbtru_nak_generation_interval	receiver	10000 (10 seconds)
transport_lbtru_nak_suppress_interval	receiver	1000 (1 second)
transport_lbtru_rate_interval	context	100
transport_lbtru_sm_maximum_interval	source	10000 (10 seconds)
transport_lbtru_sm_minimum_interval	source	200 (0.2 seconds)
transport_tcp_activity_timeout	receiver	0
transport_topic_sequence_number_info_source	active_threshold	60
transport_topic_sequence_number_info_source	interval	5000 (5 second)
ume_ack_batching_interval	context	100 (0.1 seconds)
ume_activity_timeout	receiver	0 (zero)
ume_activity_timeout	source	0 (zero)
ume_receiver_liveness_interval	context	0 (disable; do not send keepalives)
ume_registration_interval	receiver	500 (0.5 seconds)
ume_registration_interval	source	500 (0.5 seconds)
ume_retransmit_request_generation_interval	receiver	10000 (10 seconds)
ume_retransmit_request_interval	receiver	500 (0.5 seconds)
ume_source_liveness_timeout	context	0 (disable; do not track receivers)
ume_state_lifetime	receiver	0 (zero)



Configuration Option	Scope	Default Value
ume_state_lifetime	source	0 (zero)
ume_store_activity_timeout	source	3000 (3 seconds)
ume_store_check_interval	source	500 (0.5 seconds)
umq_command_interval	context	500 (0.5 seconds)
umq_delayed_consumption_report_interval	receiver	0
umq_hold_interval	receiver	10000 (10 seconds)
umq_message_retransmission_interval	context	500 (0.5 seconds)
umq_msg_total_lifetime	context	0 (zero)
umq_msg_total_lifetime	source	0 (zero)
umq_queue_activity_timeout	context	3000 (3.0 seconds)
umq_queue_check_interval	context	500 (0.5 seconds)
umq_queue_query_interval	context	200 (0.2 seconds)
umq_retransmit_request_interval	receiver	500 (0.5 seconds)
umq_ulb_check_interval	source	1000 (1 second)
umq_ulb_source_activity_timeout	receiver	10000 (10 seconds)
umq_ulb_source_check_interval	receiver	1000 (1 second)

## 4.38. Options That May Be Set During Operation

This section lists options that may be set during operation with a `lbm_*_setopt()` function.

**Table 4-5. Options That May Be Set During Operation**

Configuration Option	Scope	Default Value
implicit_batching_interval	source	200 (0.2 seconds)
implicit_batching_minimum_length	source	2048 (8192 for Microsoft Windows)
implicit_batching_type	source	>
queue_age_enabled	event_queue	0
queue_count_enabled	event_queue	0
queue_delay_warning	event_queue	0 (not monitored)
queue_enqueue_notification	event_queue	>
queue_service_time_enabled	event_queue	0
queue_size_warning	event_queue	0 (not monitored)
resolution_no_source_notification_threshold	receiver	0 (do not notify)
resolution_number_of_sources_query_threshold	receiver	10000000 (10 million)

Configuration Option	Scope	Default Value
resolver_active_source_interval	context	1000 (1 second)
resolver_active_threshold	context	60
resolver_maximum_advertisements	context	0 (all topics)
resolver_maximum_queries	context	0 (all topics with no source)
resolver_multicast_ttl	context	16
resolver_query_interval	context	100 (0.1 seconds)
resolver_query_max_interval	wildcard_receiver	0 (do not query)

## 4.39. Options (Callbacks) That Cannot Be Set From a UM Configuration File

This section lists options that require function pointers as their value and cannot be set in a UM Configuration File. These options must be set with API functions.

**Table 4-6. Options That Cannot Be Set From a UM Configuration File**

Configuration Option	Scope	Default Value
immediate_message_receiver_function	context	NULL
immediate_message_topic_receiver_function	context	NULL
mim_unrecoverable_loss_function	context	NULL
pattern_callback	wildcard_receiver	NULL
receiver_create_callback	wildcard_receiver	NULL
receiver_delete_callback	wildcard_receiver	NULL
resolver_source_notification_function	context	NULL
resolver_string_hash_function_ex	context	NULL
source_cost_evaluation_function	context	NULL
source_event_function	context	NULL
source_notification_function	receiver	NULL
ume_force_reclaim_function	source	NULL
ume_recovery_sequence_number_information	function	NULL
ume_registration_extended_function	receiver	NULL
ume_registration_function	receiver	NULL

# Index

## A

### Address

- mim\_address (context), 132
- mim\_incoming\_address (context), 132
- mim\_outgoing\_address (context), 133
- resolver\_multicast\_address (context), 83
- resolver\_multicast\_incoming\_address (context), 83
- resolver\_multicast\_outgoing\_address (context), 84
- resolver\_unicast\_address (context), 222
- transport\_lbtrm\_multicast\_address (source), 96
- transport\_lbtrm\_multicast\_address\_high (context), 97
- transport\_lbtrm\_multicast\_address\_low (context), 97
- ume\_primary\_store\_address (source), 225
- ume\_secondary\_store\_address (source), 226
- ume\_tertiary\_store\_address (source), 226

## B

### Buffer, Network

- resolver\_multicast\_receiver\_socket\_buffer (context), 85
- resolver\_unicast\_receiver\_socket\_buffer (context), 88
- response\_session\_sender\_socket\_buffer (context), 151
- transport\_lbtrm\_receiver\_socket\_buffer (context), 103
- transport\_lbtrm\_source\_socket\_buffer (context), 104
- transport\_lbtru\_receiver\_socket\_buffer (context), 114
- transport\_lbtru\_source\_socket\_buffer (context), 114
- transport\_tcp\_receiver\_socket\_buffer (context), 94
- transport\_tcp\_sender\_socket\_buffer (source), 95

## C

channel\_map\_tablesz, 156

client\_id, 206

### ConnectionFactory Scope Options

- client\_id, 206
- context\_index, 206
- create\_queue\_browser\_context, 206
- debug, 207
- default\_message\_type, 207
- default\_temp\_topic\_type, 207

- default\_topic\_type, 208
- queue\_browser\_creation\_delay, 209
- queue\_browser\_timeout, 209
- RECEIVER\_CREATION\_DELAY, 209
- SOURCE\_CREATION\_DELAY, 210
- SOURCE\_REGISTRATION\_TIMEOUT, 210
- use\_app\_header, 211
- use\_index\_queuing, 212
- use\_ump\_session\_ids, 212
- wait\_for\_source\_registration, 212

### context Scope Options

- context\_event\_function, 60
- context\_name, 60
- dbl\_lbtrm\_acceleration, 129
- dbl\_lbtru\_acceleration, 129
- dbl\_mim\_acceleration, 130
- dbl\_resolver\_acceleration, 130
- delivery\_control\_maximum\_total\_map\_entries, 157
- disable\_extended\_topic\_resolution\_message\_options, 70
- fd\_management\_type, 60
- immediate\_message\_receiver\_function, 136
- immediate\_message\_topic\_receiver\_function, 136
- mim\_activity\_timeout, 137
- mim\_address, 132
- mim\_delivery\_control\_activity\_check\_interval, 137
- mim\_delivery\_control\_activity\_timeout, 137
- mim\_delivery\_control\_loss\_check\_interval, 158
- mim\_delivery\_control\_order\_tablesz, 137
- mim\_destination\_port, 132
- mim\_ignore\_interval, 134
- mim\_implicit\_batching\_interval, 138
- mim\_implicit\_batching\_minimum\_length, 138
- mim\_incoming\_address, 132
- mim\_incoming\_destination\_port, 133
- mim\_nak\_backoff\_interval, 134
- mim\_nak\_generation\_interval, 134
- mim\_nak\_initial\_backoff\_interval, 134
- mim\_nak\_suppress\_interval, 135
- mim\_ordered\_delivery, 138
- mim\_outgoing\_address, 133
- mim\_outgoing\_destination\_port, 133
- mim\_send\_naks, 135
- mim\_sm\_maximum\_interval, 139
- mim\_sm\_minimum\_interval, 139
- mim\_sqn\_window\_increment, 139
- mim\_sqn\_window\_size, 140
- mim\_src\_deletion\_timeout, 140
- mim\_tgsz, 140

mim\_transmission\_window\_limit, 135  
 mim\_transmission\_window\_size, 135  
 mim\_unrecoverable\_loss\_function, 140  
 monitor\_appid, 217  
 monitor\_interval, 218  
 monitor\_transport, 218  
 monitor\_transport\_opts, 219  
 network\_compatibility\_mode, 61  
 operational\_mode, 62  
 receive\_thread\_pool\_size, 64  
 request\_tcp\_bind\_request\_port, 148  
 request\_tcp\_exclusiveaddr, 150  
 request\_tcp\_interface, 149  
 request\_tcp\_listen\_backlog, 150  
 request\_tcp\_port, 149  
 request\_tcp\_port\_high, 149  
 request\_tcp\_port\_low, 150  
 request\_tcp\_reuseaddr, 151  
 resolver\_active\_source\_interval, 220  
 resolver\_active\_threshold, 220  
 resolver\_cache, 73  
 resolver\_context\_advertisement\_interval, 64  
 resolver\_datagram\_max\_size, 73  
 resolver\_initial\_advertisements\_per\_second, 74  
 resolver\_initial\_advertisement\_bps, 74  
 resolver\_initial\_queries\_per\_second, 74  
 resolver\_initial\_query\_bps, 75  
 resolver\_maximum\_advertisements, 221  
 resolver\_maximum\_queries, 221  
 resolver\_multicast\_address, 83  
 resolver\_multicast\_incoming\_address, 83  
 resolver\_multicast\_incoming\_port, 83  
 resolver\_multicast\_interface, 83  
 resolver\_multicast\_outgoing\_address, 84  
 resolver\_multicast\_outgoing\_port, 84  
 resolver\_multicast\_port, 84  
 resolver\_multicast\_receiver\_socket\_buffer, 85  
 resolver\_multicast\_ttl, 85  
 resolver\_query\_interval, 221  
 resolver\_receiver\_map\_tablesz, 77  
 resolver\_source\_map\_tablesz, 77  
 resolver\_source\_notification\_function, 64  
 resolver\_string\_hash\_function, 78  
 resolver\_string\_hash\_function\_ex, 79  
 resolver\_sustain\_advertisements\_per\_second, 79  
 resolver\_sustain\_advertisement\_bps, 79  
 resolver\_sustain\_queries\_per\_second, 80  
 resolver\_sustain\_query\_bps, 80  
 resolver\_ud\_acceleration, 130  
 resolver\_unicast\_activity\_timeout, 80  
 resolver\_unicast\_address, 222  
 resolver\_unicast\_change\_interval, 81  
 resolver\_unicast\_check\_interval, 81  
 resolver\_unicast\_daemon, 86  
 resolver\_unicast\_destination\_port, 223  
 resolver\_unicast\_force\_alive, 81  
 resolver\_unicast\_interface, 87  
 resolver\_unicast\_keepalive\_interval, 82  
 resolver\_unicast\_port, 223  
 resolver\_unicast\_port\_high, 87  
 resolver\_unicast\_port\_low, 87  
 resolver\_unicast\_receiver\_socket\_buffer, 88  
 resolver\_wildcard\_queries\_per\_second, 162  
 resolver\_wildcard\_query\_bps, 163  
 resolver\_wildcard\_receiver\_map\_tablesz, 163  
 response\_session\_maximum\_buffer, 151  
 response\_session\_sender\_socket\_buffer, 151  
 response\_tcp\_deletion\_timeout, 152  
 response\_tcp\_interface, 152  
 response\_tcp\_nodelay, 152  
 source\_cost\_evaluation\_function, 64  
 source\_event\_function, 65  
 source\_includes\_topic\_index, 65  
 transport\_datagram\_max\_size, 224  
 transport\_lbtipec\_datagram\_max\_size, 122  
 transport\_lbtipec\_id\_high, 123  
 transport\_lbtipec\_id\_low, 123  
 transport\_lbtipec\_receiver\_operational\_mode, 124  
 transport\_lbtipec\_receiver\_thread\_behavior, 124  
 transport\_lbtrdma\_datagram\_max\_size, 126  
 transport\_lbtrdma\_maximum\_ports, 126  
 transport\_lbtrdma\_port\_high, 127  
 transport\_lbtrdma\_port\_low, 127  
 transport\_lbtrdma\_receiver\_thread\_behavior, 128  
 transport\_lbtrm\_datagram\_max\_size, 107  
 transport\_lbtrm\_data\_rate\_limit, 107  
 transport\_lbtrm\_multicast\_address\_high, 97  
 transport\_lbtrm\_multicast\_address\_low, 97  
 transport\_lbtrm\_rate\_interval, 108  
 transport\_lbtrm\_receiver\_socket\_buffer, 103  
 transport\_lbtrm\_retransmit\_rate\_limit, 108  
 transport\_lbtrm\_source\_port\_high, 97  
 transport\_lbtrm\_source\_port\_low, 97  
 transport\_lbtrm\_source\_socket\_buffer, 104  
 transport\_lbtru\_datagram\_max\_size, 118  
 transport\_lbtru\_data\_rate\_limit, 118

- transport\_lbtru\_maximum\_ports, 111
- transport\_lbtru\_port\_high, 111
- transport\_lbtru\_port\_low, 112
- transport\_lbtru\_rate\_interval, 119
- transport\_lbtru\_receiver\_socket\_buffer, 114
- transport\_lbtru\_retransmit\_rate\_limit, 120
- transport\_lbtru\_source\_socket\_buffer, 114
- transport\_session\_multiple\_sending\_threads, 66
- transport\_tcp\_datagram\_max\_size, 92
- transport\_tcp\_maximum\_ports, 89
- transport\_tcp\_port\_high, 90
- transport\_tcp\_port\_low, 90
- transport\_tcp\_receiver\_socket\_buffer, 94
- ud\_acceleration, 131
- ume\_ack\_batching\_interval, 166
- ume\_receiver\_liveness\_interval, 173
- ume\_session\_id, 180
- ume\_source\_liveness\_timeout, 182
- ume\_user\_receiver\_registration\_id, 186
- umq\_command\_interval, 187
- umq\_command\_outstanding\_maximum, 187
- umq\_flight\_size, 188
- umq\_flight\_size\_behavior, 188
- umq\_message\_retransmission\_interval, 190
- umq\_message\_stability\_notification, 190
- umq\_msg\_total\_lifetime, 191
- umq\_queue\_activity\_timeout, 192
- umq\_queue\_check\_interval, 192
- umq\_queue\_query\_interval, 194
- umq\_queue\_registration\_id, 194
- umq\_require\_queue\_authentication, 195
- umq\_retention\_intergroup\_stability\_behavior, 195
- umq\_retention\_intragroup\_stability\_behavior, 196
- umq\_session\_id, 198
- context\_event\_function, 60
- context\_index, 206
- context\_name, 60
- create\_queue\_browser\_context, 206

## D

- dbl\_lbtrm\_acceleration, 129
- dbl\_lbtru\_acceleration, 129
- dbl\_mim\_acceleration, 130
- dbl\_resolver\_acceleration, 130
- debug, 207
- default\_message\_type, 207

- default\_temp\_topic\_type, 207
- default\_topic\_type, 208
- delivery\_control\_loss\_check\_interval, 156, 213
- delivery\_control\_loss\_tablesz, 157
- delivery\_control\_maximum\_burst\_loss, 157, 214
- delivery\_control\_maximum\_total\_map\_entries, 157, 214
- delivery\_control\_max\_delay, 214
- delivery\_control\_order\_tablesz, 157
- Destination Scope Options
  - DestType, 208
  - regid, 210
  - Topic, 210
  - Type, 211
  - Wildcard, 213
- DestType, 208
- disable\_extended\_topic\_resolution\_message\_options, 70
- duplicate\_delivery, 215
- Dynamic
  - implicit\_batching\_interval (source), 153
  - implicit\_batching\_minimum\_length (source), 153
  - implicit\_batching\_type (source), 153
  - queue\_age\_enabled (event\_queue), 163
  - queue\_count\_enabled (event\_queue), 164
  - queue\_delay\_warning (event\_queue), 165
  - queue\_enqueue\_notification (event\_queue), 165
  - queue\_service\_time\_enabled (event\_queue), 166
  - queue\_size\_warning (event\_queue), 166
  - resolution\_no\_source\_notification\_threshold (receiver), 70
  - resolution\_number\_of\_sources\_query\_threshold (receiver), 71
  - resolver\_active\_source\_interval (context), 220
  - resolver\_active\_threshold (context), 220
  - resolver\_maximum\_advertisements (context), 221
  - resolver\_maximum\_queries (context), 221
  - resolver\_multicast\_ttl (context), 85
  - resolver\_query\_interval (context), 221
  - resolver\_query\_max\_interval (wildcard\_receiver), 222

## E

### event\_queue Scope Options

- event\_queue\_name, 163
- monitor\_appid, 217
- monitor\_interval, 218
- monitor\_transport, 219
- monitor\_transport\_opts, 219
- queue\_age\_enabled, 163
- queue\_cancellation\_callbacks\_enabled, 164
- queue\_count\_enabled, 164
- queue\_delay\_warning, 165
- queue\_enqueue\_notification, 165
- queue\_objects\_purged\_on\_close, 165
- queue\_service\_time\_enabled, 166
- queue\_size\_warning, 166

event\_queue\_name, 163

## F

fd\_management\_type, 60

## H

### hfx Scope Options

- delivery\_control\_loss\_check\_interval, 213
- delivery\_control\_maximum\_burst\_loss, 214
- delivery\_control\_maximum\_total\_map\_entries, 214
- delivery\_control\_max\_delay, 214
- duplicate\_delivery, 215
- ordered\_delivery, 216

hf\_duplicate\_delivery, 215

hf\_optional\_messages, 215

hf\_receiver, 216

## I

immediate\_message\_receiver\_function, 136

immediate\_message\_topic\_receiver\_function, 136

implicit\_batching\_interval, 153

implicit\_batching\_minimum\_length, 153

implicit\_batching\_type, 153

### Interface

request\_tcp\_interface (context), 149

resolver\_multicast\_interface (context), 83

resolver\_unicast\_interface (context), 87

response\_tcp\_interface (context), 152

transport\_lbtrdma\_interface (source), 126

transport\_lbtru\_interface (receiver), 110

transport\_lbtru\_interface (source), 111

transport\_tcp\_interface (receiver), 89

transport\_tcp\_interface (source), 89

## L

late\_join, 142

## M

message\_selector, 61

mim\_activity\_timeout, 137

mim\_address, 132

mim\_delivery\_control\_activity\_check\_interval, 137

mim\_delivery\_control\_activity\_timeout, 137

mim\_delivery\_control\_loss\_check\_interval, 158

mim\_delivery\_control\_order\_tablesz, 137

mim\_destination\_port, 132

mim\_ignore\_interval, 134

mim\_implicit\_batching\_interval, 138

mim\_implicit\_batching\_minimum\_length, 138

mim\_incoming\_address, 132

mim\_incoming\_destination\_port, 133

mim\_nak\_backoff\_interval, 134

mim\_nak\_generation\_interval, 134

mim\_nak\_initial\_backoff\_interval, 134

mim\_nak\_suppress\_interval, 135

mim\_ordered\_delivery, 138

mim\_outgoing\_address, 133

mim\_outgoing\_destination\_port, 133

mim\_send\_naks, 135

mim\_sm\_maximum\_interval, 139

mim\_sm\_minimum\_interval, 139

mim\_sqn\_window\_increment, 139

mim\_sqn\_window\_size, 140

mim\_src\_deletion\_timeout, 140

mim\_tgsz, 140

mim\_transmission\_window\_limit, 135

mim\_transmission\_window\_size, 135

mim\_unrecoverable\_loss\_function, 140

monitor\_appid, 217, 217  
monitor\_interval, 218, 218  
monitor\_transport, 218, 219  
monitor\_transport\_opts, 219, 219

#### Multicast address

mim\_address (context), 132  
mim\_incoming\_address (context), 132  
mim\_outgoing\_address (context), 133  
resolver\_multicast\_address (context), 83  
resolver\_multicast\_incoming\_address (context), 83  
resolver\_multicast\_outgoing\_address (context), 84  
transport\_lbtrm\_multicast\_address (source), 96  
transport\_lbtrm\_multicast\_address\_high (context), 97  
transport\_lbtrm\_multicast\_address\_low (context), 97

## N

#### Network buffer

resolver\_multicast\_receiver\_socket\_buffer (context), 85  
resolver\_unicast\_receiver\_socket\_buffer (context), 88  
response\_session\_sender\_socket\_buffer (context), 151  
transport\_lbtrm\_receiver\_socket\_buffer (context), 103  
transport\_lbtrm\_source\_socket\_buffer (context), 104  
transport\_lbtru\_receiver\_socket\_buffer (context), 114  
transport\_lbtru\_source\_socket\_buffer (context), 114  
transport\_tcp\_receiver\_socket\_buffer (context), 94  
transport\_tcp\_sender\_socket\_buffer (source), 95

network\_compatibility\_mode, 61

#### no-config-file

immediate\_message\_receiver\_function (context), 136  
immediate\_message\_topic\_receiver\_function (context), 136  
mim\_unrecoverable\_loss\_function (context), 140  
pattern\_callback (wildcard\_receiver), 159  
receiver\_create\_callback (wildcard\_receiver), 160  
receiver\_delete\_callback (wildcard\_receiver), 161  
resolver\_source\_notification\_function (context), 64  
resolver\_string\_hash\_function\_ex (context), 79  
source\_cost\_evaluation\_function (context), 64  
source\_event\_function (context), 65  
source\_notification\_function (receiver), 159  
ume\_force\_reclaim\_function (source), 171  
ume\_recovery\_sequence\_number\_info\_function (receiver), 174  
ume\_registration\_extended\_function (receiver), 174  
ume\_registration\_function (receiver), 175

null\_channel\_behavior, 158

## O

operational\_mode, 62  
ordered\_delivery, 62, 216  
otr\_request\_duration, 146  
otr\_request\_initial\_delay, 146  
otr\_request\_log\_alert\_cooldown, 147  
otr\_request\_maximum\_interval, 147  
otr\_request\_minimum\_interval, 147  
otr\_request\_outstanding\_maximum, 148

## P

pattern\_callback, 159

pattern\_type, 160

#### Port

mim\_destination\_port (context), 132  
mim\_incoming\_destination\_port (context), 133  
mim\_outgoing\_destination\_port (context), 133  
request\_tcp\_port (context), 149  
request\_tcp\_port\_high (context), 149  
request\_tcp\_port\_low (context), 150  
resolver\_multicast\_incoming\_port (context), 83  
resolver\_multicast\_outgoing\_port (context), 84  
resolver\_multicast\_port (context), 84  
resolver\_unicast\_destination\_port (context), 223  
resolver\_unicast\_port (context), 223  
resolver\_unicast\_port\_high (context), 87  
resolver\_unicast\_port\_low (context), 87  
transport\_lbtrdma\_maximum\_ports (context), 126  
transport\_lbtrm\_destination\_port (source), 96  
transport\_lbtrm\_source\_port\_high (context), 97  
transport\_lbtrm\_source\_port\_low (context), 97  
transport\_lbtru\_maximum\_ports (context), 111  
transport\_lbtru\_port (source), 111  
transport\_lbtru\_port\_high (context), 111  
transport\_lbtru\_port\_high (receiver), 112  
transport\_lbtru\_port\_low (context), 112  
transport\_lbtru\_port\_low (receiver), 112  
transport\_tcp\_maximum\_ports (context), 89  
transport\_tcp\_port (source), 89  
transport\_tcp\_port\_high (context), 90  
transport\_tcp\_port\_low (context), 90  
ume\_primary\_store\_port (source), 225  
ume\_secondary\_store\_port (source), 226  
ume\_tertiary\_store\_port (source), 227

## Q

- queue\_age\_enabled, 163
- queue\_browser\_creation\_delay, 209
- queue\_browser\_timeout, 209
- queue\_cancellation\_callbacks\_enabled, 164
- queue\_count\_enabled, 164
- queue\_delay\_warning, 165
- queue\_enqueue\_notification, 165
- queue\_objects\_purged\_on\_close, 165
- queue\_service\_time\_enabled, 166
- queue\_size\_warning, 166

## R

- rcv\_sync\_cache, 63
- rcv\_sync\_cache\_timeout, 63
- receiver Scope Options
  - channel\_map\_tablesz, 156
  - delivery\_control\_loss\_check\_interval, 156
  - delivery\_control\_loss\_tablesz, 157
  - delivery\_control\_maximum\_burst\_loss, 157
  - delivery\_control\_order\_tablesz, 157
  - hf\_duplicate\_delivery, 215
  - hf\_optional\_messages, 215
  - message\_selector, 61
  - null\_channel\_behavior, 158
  - ordered\_delivery, 62
  - otr\_request\_duration, 146
  - otr\_request\_initial\_delay, 146
  - otr\_request\_log\_alert\_cooldown, 147
  - otr\_request\_maximum\_interval, 147
  - otr\_request\_minimum\_interval, 147
  - otr\_request\_outstanding\_maximum, 148
- rcv\_sync\_cache, 63
- rcv\_sync\_cache\_timeout, 63
- resolution\_no\_source\_notification\_threshold, 70
- resolution\_number\_of\_sources\_query\_threshold, 71
- resolver\_query\_maximum\_initial\_interval, 75
- resolver\_query\_minimum\_initial\_duration, 75
- resolver\_query\_minimum\_initial\_interval, 76
- resolver\_query\_minimum\_sustain\_duration, 76
- resolver\_query\_sustain\_interval, 76
- retransmit\_initial\_sequence\_number\_request, 142
- retransmit\_message\_caching\_proximity, 143
- retransmit\_request\_generation\_interval, 143
- retransmit\_request\_interval, 143

- retransmit\_request\_maximum, 144
- retransmit\_request\_outstanding\_maximum, 144
- source\_notification\_function, 159
- transport\_demux\_tablesz, 66
- transport\_lbtpc\_acknowledgement\_interval, 224
- transport\_lbtpc\_activity\_timeout, 121
- transport\_lbtrm\_activity\_timeout, 106
- transport\_lbtrm\_nak\_backoff\_interval, 101
- transport\_lbtrm\_nak\_generation\_interval, 102
- transport\_lbtrm\_nak\_initial\_backoff\_interval, 102
- transport\_lbtrm\_nak\_suppress\_interval, 103
- transport\_lbtrm\_preactivity\_timeout, 107
- transport\_lbtrm\_send\_naks, 103
- transport\_lbtru\_acknowledgement\_interval, 116
- transport\_lbtru\_activity\_timeout, 116
- transport\_lbtru\_connect\_interval, 118
- transport\_lbtru\_interface, 110
- transport\_lbtru\_maximum\_connect\_attempts, 119
- transport\_lbtru\_nak\_backoff\_interval, 113
- transport\_lbtru\_nak\_generation\_interval, 113
- transport\_lbtru\_nak\_suppress\_interval, 114
- transport\_lbtru\_port\_high, 112
- transport\_lbtru\_port\_low, 112
- transport\_tcp\_activity\_method, 91
- transport\_tcp\_activity\_timeout, 91
- transport\_tcp\_interface, 89
- ume\_activity\_timeout, 167
- ume\_allow\_confirmed\_delivery, 167
- ume\_consensus\_sequence\_number\_behavior, 168
- ume\_explicit\_ack\_only, 169
- ume\_receiver\_paced\_persistence, 173
- ume\_recovery\_sequence\_number\_info\_function, 174
- ume\_registration\_extended\_function, 174
- ume\_registration\_function, 175
- ume\_registration\_interval, 175
- ume\_retransmit\_request\_generation\_interval, 179
- ume\_retransmit\_request\_interval, 180
- ume\_retransmit\_request\_maximum, 180
- ume\_retransmit\_request\_outstanding\_maximum, 180
- ume\_session\_id, 181
- ume\_state\_lifetime, 182
- ume\_use\_ack\_batching, 185
- ume\_use\_late\_join, 185
- ume\_use\_store, 186
- umq\_delayed\_consumption\_report\_interval, 187
- umq\_hold\_interval, 189
- umq\_index\_assignment\_eligibility\_default, 190
- umq\_queue\_participation, 193



- umq\_receiver\_type\_id, 194
- umq\_retransmit\_request\_interval, 197
- umq\_retransmit\_request\_outstanding\_maximum, 198
- umq\_ulb\_source\_activity\_timeout, 205
- umq\_ulb\_source\_check\_interval, 205
- unrecognized\_channel\_behavior, 159
- use\_late\_join, 145
- use\_otr, 148
- use\_transport\_thread, 68
- receiver\_create\_callback, 160
- RECEIVER\_CREATION\_DELAY, 209
- receiver\_delete\_callback, 161
- receive\_thread\_pool\_size, 64
- regid, 210
- request\_tcp\_bind\_request\_port, 148
- request\_tcp\_exclusiveaddr, 150
- request\_tcp\_interface, 149
- request\_tcp\_listen\_backlog, 150
- request\_tcp\_port, 149
- request\_tcp\_port\_high, 149
- request\_tcp\_port\_low, 150
- request\_tcp\_reuseaddr, 151
- resolution\_no\_source\_notification\_threshold, 70
- resolution\_number\_of\_sources\_query\_threshold, 71
- resolver\_active\_source\_interval, 220
- resolver\_active\_threshold, 220
- resolver\_advertisement\_maximum\_initial\_interval, 71
- resolver\_advertisement\_minimum\_initial\_duration, 71
- resolver\_advertisement\_minimum\_initial\_interval, 72
- resolver\_advertisement\_minimum\_sustain\_duration, 72
- resolver\_advertisement\_send\_immediate\_response, 72
- resolver\_advertisement\_sustain\_interval, 73
- resolver\_cache, 73
- resolver\_context\_advertisement\_interval, 64
- resolver\_datagram\_max\_size, 73
- resolver\_initial\_advertisements\_per\_second, 74
- resolver\_initial\_advertisement\_bps, 74
- resolver\_initial\_queries\_per\_second, 74
- resolver\_initial\_query\_bps, 75
- resolver\_maximum\_advertisements, 221
- resolver\_maximum\_queries, 221
- resolver\_multicast\_address, 83
- resolver\_multicast\_incoming\_address, 83
- resolver\_multicast\_incoming\_port, 83
- resolver\_multicast\_interface, 83
- resolver\_multicast\_outgoing\_address, 84
- resolver\_multicast\_outgoing\_port, 84
- resolver\_multicast\_port, 84
- resolver\_multicast\_receiver\_socket\_buffer, 85
- resolver\_multicast\_ttl, 85
- resolver\_no\_source\_linger\_timeout, 161
- resolver\_query\_interval, 221
- resolver\_query\_maximum\_initial\_interval, 75
- resolver\_query\_maximum\_interval, 161
- resolver\_query\_max\_interval, 222
- resolver\_query\_minimum\_duration, 162
- resolver\_query\_minimum\_initial\_duration, 75
- resolver\_query\_minimum\_initial\_interval, 76
- resolver\_query\_minimum\_interval, 162
- resolver\_query\_minimum\_sustain\_duration, 76
- resolver\_query\_sustain\_interval, 76
- resolver\_receiver\_map\_tablesz, 77
- resolver\_send\_initial\_advertisement, 77
- resolver\_source\_map\_tablesz, 77
- resolver\_source\_notification\_function, 64
- resolver\_string\_hash\_function, 78
- resolver\_string\_hash\_function\_ex, 79
- resolver\_sustain\_advertisements\_per\_second, 79
- resolver\_sustain\_advertisement\_bps, 79
- resolver\_sustain\_queries\_per\_second, 80
- resolver\_sustain\_query\_bps, 80
- resolver\_ud\_acceleration, 130
- resolver\_unicast\_activity\_timeout, 80
- resolver\_unicast\_address, 222
- resolver\_unicast\_change\_interval, 81
- resolver\_unicast\_check\_interval, 81
- resolver\_unicast\_daemon, 86
- resolver\_unicast\_destination\_port, 223
- resolver\_unicast\_force\_alive, 81
- resolver\_unicast\_interface, 87
- resolver\_unicast\_keepalive\_interval, 82
- resolver\_unicast\_port, 223
- resolver\_unicast\_port\_high, 87
- resolver\_unicast\_port\_low, 87
- resolver\_unicast\_receiver\_socket\_buffer, 88
- resolver\_wildcard\_queries\_per\_second, 162
- resolver\_wildcard\_query\_bps, 163
- resolver\_wildcard\_receiver\_map\_tablesz, 163
- response\_session\_maximum\_buffer, 151
- response\_session\_sender\_socket\_buffer, 151
- response\_tcp\_deletion\_timeout, 152
- response\_tcp\_interface, 152
- response\_tcp\_nodelay, 152
- retransmit\_initial\_sequence\_number\_request, 142
- retransmit\_message\_caching\_proximity, 143
- retransmit\_message\_map\_tablesz, 223

- retransmit\_request\_generation\_interval, 143
- retransmit\_request\_interval, 143
- retransmit\_request\_maximum, 144
- retransmit\_request\_outstanding\_maximum, 144
- retransmit\_retention\_age\_threshold, 144
- retransmit\_retention\_size\_limit, 145
- retransmit\_retention\_size\_threshold, 145

## S

### Socket option

- resolver\_multicast\_receiver\_socket\_buffer (context), 85
- resolver\_unicast\_receiver\_socket\_buffer (context), 88
- response\_session\_sender\_socket\_buffer (context), 151
- response\_tcp\_nodelay (context), 152
- transport\_lbtrm\_receiver\_socket\_buffer (context), 103
- transport\_lbtrm\_source\_socket\_buffer (context), 104
- transport\_lbtru\_receiver\_socket\_buffer (context), 114
- transport\_lbtru\_source\_socket\_buffer (context), 114
- transport\_tcp\_nodelay (source), 94
- transport\_tcp\_receiver\_socket\_buffer (context), 94
- transport\_tcp\_sender\_socket\_buffer (source), 95

### source Scope Options

- implicit\_batching\_interval, 153
- implicit\_batching\_minimum\_length, 153
- implicit\_batching\_type, 153
- late\_join, 142
- resolver\_advertisement\_maximum\_initial\_interval, 71
- resolver\_advertisement\_minimum\_initial\_duration, 71
- resolver\_advertisement\_minimum\_initial\_interval, 72
- resolver\_advertisement\_minimum\_sustain\_duration, 72
- resolver\_advertisement\_send\_immediate\_response, 72
- resolver\_advertisement\_sustain\_interval, 73
- resolver\_send\_initial\_advertisement, 77
- retransmit\_message\_map\_tablesz, 223
- retransmit\_retention\_age\_threshold, 144
- retransmit\_retention\_size\_limit, 145
- retransmit\_retention\_size\_threshold, 145
- transport, 66
- transport\_lbtiptc\_behavior, 122
- transport\_lbtiptc\_client\_activity\_timeout, 224
- transport\_lbtiptc\_id, 123
- transport\_lbtiptc\_maximum\_receivers\_per\_transport, 124
- transport\_lbtiptc\_sm\_interval, 125
- transport\_lbtiptc\_transmission\_window\_size, 125
- transport\_lbtrdma\_interface, 126
- transport\_lbtrdma\_port, 127

- transport\_lbtrdma\_transmission\_window\_size, 128
- transport\_lbtrm\_coalesce\_threshold, 106
- transport\_lbtrm\_destination\_port, 96
- transport\_lbtrm\_ignore\_interval, 101
- transport\_lbtrm\_multicast\_address, 96
- transport\_lbtrm\_sm\_maximum\_interval, 108
- transport\_lbtrm\_sm\_minimum\_interval, 109
- transport\_lbtrm\_tgsz, 109
- transport\_lbtrm\_transmission\_window\_limit, 104
- transport\_lbtrm\_transmission\_window\_size, 104
- transport\_lbtru\_client\_activity\_timeout, 117
- transport\_lbtru\_client\_map\_size, 117
- transport\_lbtru\_coalesce\_threshold, 117
- transport\_lbtru\_ignore\_interval, 113
- transport\_lbtru\_interface, 111
- transport\_lbtru\_port, 111
- transport\_lbtru\_sm\_maximum\_interval, 120
- transport\_lbtru\_sm\_minimum\_interval, 120
- transport\_lbtru\_transmission\_window\_limit, 115
- transport\_lbtru\_transmission\_window\_size, 115
- transport\_lbtru\_use\_session\_id, 121
- transport\_session\_maximum\_buffer, 90
- transport\_source\_side\_filtering\_behavior, 67
- transport\_tcp\_coalesce\_threshold, 91
- transport\_tcp\_exclusiveaddr, 92
- transport\_tcp\_interface, 89
- transport\_tcp\_listen\_backlog, 93
- transport\_tcp\_multiple\_receiver\_behavior, 93
- transport\_tcp\_multiple\_receiver\_send\_order, 93
- transport\_tcp\_nodelay, 94
- transport\_tcp\_port, 89
- transport\_tcp\_reuseaddr, 95
- transport\_tcp\_sender\_socket\_buffer, 95
- transport\_topic\_sequence\_number\_info\_active\_threshold, 67
- transport\_topic\_sequence\_number\_info\_interval, 68
- ume\_activity\_timeout, 167
- ume\_confirmed\_delivery\_notification, 168
- ume\_consensus\_sequence\_number\_behavior, 169
- ume\_flight\_size, 170
- ume\_flight\_size\_behavior, 170
- ume\_flight\_size\_bytes, 171
- ume\_force\_reclaim\_function, 171
- ume\_late\_join, 171
- ume\_message\_map\_tablesz, 225
- ume\_message\_stability\_notification, 172
- ume\_primary\_store\_address, 225
- ume\_primary\_store\_port, 225

- ume\_proxy\_source, 172
- ume\_receiver\_paced\_persistence, 173
- ume\_registration\_id, 225
- ume\_registration\_interval, 175
- ume\_repository\_ack\_on\_reception, 175
- ume\_repository\_disk\_file\_size\_limit, 176
- ume\_repository\_size\_limit, 176
- ume\_repository\_size\_threshold, 177
- ume\_retention\_intergroup\_stability\_behavior, 177
- ume\_retention\_intragroup\_stability\_behavior, 178
- ume\_retention\_size\_limit, 178
- ume\_retention\_size\_threshold, 179
- ume\_retention\_unique\_confirmations, 179
- ume\_secondary\_store\_address, 226
- ume\_secondary\_store\_port, 226
- ume\_session\_id, 181
- ume\_state\_lifetime, 182
- ume\_store, 182
- ume\_store\_activity\_timeout, 183
- ume\_store\_behavior, 183
- ume\_store\_check\_interval, 184
- ume\_store\_group, 184
- ume\_store\_name, 184
- ume\_tertiary\_store\_address, 226
- ume\_tertiary\_store\_port, 227
- ume\_write\_delay, 186
- umq\_flight\_size, 188
- umq\_flight\_size\_behavior, 189
- umq\_message\_stability\_notification, 191
- umq\_msg\_total\_lifetime, 191
- umq\_queue\_name, 192
- umq\_queue\_participants\_only, 193
- umq\_retention\_intergroup\_stability\_behavior, 196
- umq\_retention\_intragroup\_stability\_behavior, 197
- umq\_ulb\_application\_set, 198
- umq\_ulb\_application\_set\_assignment\_function, 199
- umq\_ulb\_application\_set\_events, 199
- umq\_ulb\_application\_set\_load\_factor\_behavior, 199
- umq\_ulb\_application\_set\_message\_lifetime, 200
- umq\_ulb\_application\_set\_message\_max\_reassignments, 200
- umq\_ulb\_application\_set\_message\_reassignment\_timeout, 201
- umq\_ulb\_application\_set\_receiver\_activity\_timeout, 201
- umq\_ulb\_application\_set\_receiver\_keepalive\_interval, 201
- umq\_ulb\_application\_set\_round\_robin\_bias, 202
- umq\_ulb\_check\_interval, 202

- umq\_ulb\_events, 202
- umq\_ulb\_flight\_size, 203
- umq\_ulb\_flight\_size\_behavior, 203
- umq\_ulb\_receiver\_events, 204
- umq\_ulb\_receiver\_portion, 204
- umq\_ulb\_receiver\_priority, 205
- use\_extended\_reclaim\_notifications, 68
- source\_cost\_evaluation\_function, 64
- SOURCE\_CREATION\_DELAY, 210
- source\_event\_function, 65
- source\_includes\_topic\_index, 65
- source\_notification\_function, 159
- SOURCE\_REGISTRATION\_TIMEOUT, 210

## T

### TCP Port

- request\_tcp\_port (context), 149
- request\_tcp\_port\_high (context), 149
- request\_tcp\_port\_low (context), 150
- transport\_tcp\_maximum\_ports (context), 89
- transport\_tcp\_port (source), 89
- transport\_tcp\_port\_high (context), 90
- transport\_tcp\_port\_low (context), 90
- ume\_primary\_store\_port (source), 225
- ume\_secondary\_store\_port (source), 226
- ume\_tertiary\_store\_port (source), 227

### Timer interval

- delivery\_control\_loss\_check\_interval (receiver), 156
- implicit\_batching\_interval (source), 153
- mim\_activity\_timeout (context), 137
- mim\_delivery\_control\_loss\_check\_interval (context), 158
- mim\_ignore\_interval (context), 134
- mim\_implicit\_batching\_interval (context), 138
- mim\_nak\_backoff\_interval (context), 134
- mim\_nak\_generation\_interval (context), 134
- mim\_nak\_initial\_backoff\_interval (context), 134
- mim\_nak\_suppress\_interval (context), 135
- mim\_sm\_maximum\_interval (context), 139
- mim\_sm\_minimum\_interval (context), 139
- mim\_src\_deletion\_timeout (context), 140
- rcv\_sync\_cache\_timeout (receiver), 63
- resolver\_active\_source\_interval (context), 220
- resolver\_advertisement\_maximum\_initial\_interval (source), 71

resolver\_advertisement\_minimum\_initial\_duration  
 (source), 71  
 resolver\_advertisement\_minimum\_initial\_interval  
 (source), 72  
 resolver\_advertisement\_minimum\_sustain\_duration  
 (source), 72  
 resolver\_advertisement\_sustain\_interval (source), 73  
 resolver\_context\_advertisement\_interval (context), 64  
 resolver\_no\_source\_linger\_timeout (wildcard\_receiver),  
 161  
 resolver\_query\_interval (context), 221  
 resolver\_query\_maximum\_initial\_interval (receiver), 75  
 resolver\_query\_maximum\_interval (wildcard\_receiver),  
 161  
 resolver\_query\_max\_interval (wildcard\_receiver), 222  
 resolver\_query\_minimum\_duration (wildcard\_receiver),  
 162  
 resolver\_query\_minimum\_initial\_duration (receiver), 75  
 resolver\_query\_minimum\_initial\_interval (receiver), 76  
 resolver\_query\_minimum\_interval (wildcard\_receiver),  
 162  
 resolver\_query\_minimum\_sustain\_duration (receiver),  
 76  
 resolver\_query\_sustain\_interval (receiver), 76  
 response\_tcp\_deletion\_timeout (context), 152  
 retransmit\_request\_generation\_interval (receiver), 143  
 retransmit\_request\_interval (receiver), 143  
 transport\_lbtpc\_acknowledgement\_interval (receiver),  
 224  
 transport\_lbtpc\_activity\_timeout (receiver), 121  
 transport\_lbtpc\_client\_activity\_timeout (source), 224  
 transport\_lbtpc\_sm\_interval (source), 125  
 transport\_lbtrm\_activity\_timeout (receiver), 106  
 transport\_lbtrm\_ignore\_interval (source), 101  
 transport\_lbtrm\_nak\_backoff\_interval (receiver), 101  
 transport\_lbtrm\_nak\_generation\_interval (receiver), 102  
 transport\_lbtrm\_nak\_initial\_backoff\_interval (receiver),  
 102  
 transport\_lbtrm\_nak\_suppress\_interval (receiver), 103  
 transport\_lbtrm\_preactivity\_timeout (receiver), 107  
 transport\_lbtrm\_rate\_interval (context), 108  
 transport\_lbtrm\_sm\_maximum\_interval (source), 108  
 transport\_lbtrm\_sm\_minimum\_interval (source), 109  
 transport\_lbtru\_acknowledgement\_interval (receiver),  
 116  
 transport\_lbtru\_activity\_timeout (receiver), 116  
 transport\_lbtru\_client\_activity\_timeout (source), 117  
 transport\_lbtru\_connect\_interval (receiver), 118

transport\_lbtru\_ignore\_interval (source), 113  
 transport\_lbtru\_nak\_backoff\_interval (receiver), 113  
 transport\_lbtru\_nak\_generation\_interval (receiver), 113  
 transport\_lbtru\_nak\_suppress\_interval (receiver), 114  
 transport\_lbtru\_rate\_interval (context), 119  
 transport\_lbtru\_sm\_maximum\_interval (source), 120  
 transport\_lbtru\_sm\_minimum\_interval (source), 120  
 transport\_tcp\_activity\_timeout (receiver), 91  
 transport\_topic\_sequence\_number\_info\_active\_threshold  
 (source), 67  
 transport\_topic\_sequence\_number\_info\_interval  
 (source), 68  
 ume\_ack\_batching\_interval (context), 166  
 ume\_activity\_timeout (receiver), 167  
 ume\_activity\_timeout (source), 167  
 ume\_receiver\_liveness\_interval (context), 173  
 ume\_registration\_interval (receiver), 175  
 ume\_registration\_interval (source), 175  
 ume\_retransmit\_request\_generation\_interval (receiver),  
 179  
 ume\_retransmit\_request\_interval (receiver), 180  
 ume\_source\_liveness\_timeout (context), 182  
 ume\_state\_lifetime (receiver), 182  
 ume\_state\_lifetime (source), 182  
 ume\_store\_activity\_timeout (source), 183  
 ume\_store\_check\_interval (source), 184  
 umq\_command\_interval (context), 187  
 umq\_delayed\_consumption\_report\_interval (receiver),  
 187  
 umq\_hold\_interval (receiver), 189  
 umq\_message\_retransmission\_interval (context), 190  
 umq\_msg\_total\_lifetime (context), 191  
 umq\_msg\_total\_lifetime (source), 191  
 umq\_queue\_activity\_timeout (context), 192  
 umq\_queue\_check\_interval (context), 192  
 umq\_queue\_query\_interval (context), 194  
 umq\_retransmit\_request\_interval (receiver), 197  
 umq\_ulb\_check\_interval (source), 202  
 umq\_ulb\_source\_activity\_timeout (receiver), 205  
 umq\_ulb\_source\_check\_interval (receiver), 205  
 Topic, 210  
 transport, 66  
 transport\_datagram\_max\_size, 224  
 transport\_demux\_tablesz, 66  
 transport\_lbtpc\_acknowledgement\_interval, 224  
 transport\_lbtpc\_activity\_timeout, 121  
 transport\_lbtpc\_behavior, 122  
 transport\_lbtpc\_client\_activity\_timeout, 224

transport\_lbtpc\_datagram\_max\_size, 122  
 transport\_lbtpc\_id, 123  
 transport\_lbtpc\_id\_high, 123  
 transport\_lbtpc\_id\_low, 123  
 transport\_lbtpc\_maximum\_receivers\_per\_transport, 124  
 transport\_lbtpc\_receiver\_operational\_mode, 124  
 transport\_lbtpc\_receiver\_thread\_behavior, 124  
 transport\_lbtpc\_sm\_interval, 125  
 transport\_lbtpc\_transmission\_window\_size, 125  
 transport\_lbtrdma\_datagram\_max\_size, 126  
 transport\_lbtrdma\_interface, 126  
 transport\_lbtrdma\_maximum\_ports, 126  
 transport\_lbtrdma\_port, 127  
 transport\_lbtrdma\_port\_high, 127  
 transport\_lbtrdma\_port\_low, 127  
 transport\_lbtrdma\_receiver\_thread\_behavior, 128  
 transport\_lbtrdma\_transmission\_window\_size, 128  
 transport\_lbtrm\_activity\_timeout, 106  
 transport\_lbtrm\_coalesce\_threshold, 106  
 transport\_lbtrm\_datagram\_max\_size, 107  
 transport\_lbtrm\_data\_rate\_limit, 107  
 transport\_lbtrm\_destination\_port, 96  
 transport\_lbtrm\_ignore\_interval, 101  
 transport\_lbtrm\_multicast\_address, 96  
 transport\_lbtrm\_multicast\_address\_high, 97  
 transport\_lbtrm\_multicast\_address\_low, 97  
 transport\_lbtrm\_nak\_backoff\_interval, 101  
 transport\_lbtrm\_nak\_generation\_interval, 102  
 transport\_lbtrm\_nak\_initial\_backoff\_interval, 102  
 transport\_lbtrm\_nak\_suppress\_interval, 103  
 transport\_lbtrm\_preactivity\_timeout, 107  
 transport\_lbtrm\_rate\_interval, 108  
 transport\_lbtrm\_receiver\_socket\_buffer, 103  
 transport\_lbtrm\_retransmit\_rate\_limit, 108  
 transport\_lbtrm\_send\_naks, 103  
 transport\_lbtrm\_sm\_maximum\_interval, 108  
 transport\_lbtrm\_sm\_minimum\_interval, 109  
 transport\_lbtrm\_source\_port\_high, 97  
 transport\_lbtrm\_source\_port\_low, 97  
 transport\_lbtrm\_source\_socket\_buffer, 104  
 transport\_lbtrm\_tgsz, 109  
 transport\_lbtrm\_transmission\_window\_limit, 104  
 transport\_lbtrm\_transmission\_window\_size, 104  
 transport\_lbtru\_acknowledgement\_interval, 116  
 transport\_lbtru\_activity\_timeout, 116  
 transport\_lbtru\_client\_activity\_timeout, 117  
 transport\_lbtru\_client\_map\_size, 117  
 transport\_lbtru\_coalesce\_threshold, 117  
 transport\_lbtru\_connect\_interval, 118  
 transport\_lbtru\_datagram\_max\_size, 118  
 transport\_lbtru\_data\_rate\_limit, 118  
 transport\_lbtru\_ignore\_interval, 113  
 transport\_lbtru\_interface, 110, 111  
 transport\_lbtru\_maximum\_connect\_attempts, 119  
 transport\_lbtru\_maximum\_ports, 110, 111  
 transport\_lbtru\_nak\_backoff\_interval, 113  
 transport\_lbtru\_nak\_generation\_interval, 113  
 transport\_lbtru\_nak\_suppress\_interval, 114  
 transport\_lbtru\_port, 111  
 transport\_lbtru\_port\_high, 110, 111, 112  
 transport\_lbtru\_port\_low, 110, 112, 112  
 transport\_lbtru\_rate\_interval, 119  
 transport\_lbtru\_receiver\_socket\_buffer, 114  
 transport\_lbtru\_retransmit\_rate\_limit, 120  
 transport\_lbtru\_sm\_maximum\_interval, 120  
 transport\_lbtru\_sm\_minimum\_interval, 120  
 transport\_lbtru\_source\_socket\_buffer, 114  
 transport\_lbtru\_transmission\_window\_limit, 115  
 transport\_lbtru\_transmission\_window\_size, 115  
 transport\_lbtru\_use\_session\_id, 121  
 transport\_session\_maximum\_buffer, 90  
 transport\_session\_multiple\_sending\_threads, 66  
 transport\_source\_side\_filtering\_behavior, 67  
 transport\_tcp\_activity\_method, 91  
 transport\_tcp\_activity\_timeout, 91  
 transport\_tcp\_coalesce\_threshold, 91  
 transport\_tcp\_datagram\_max\_size, 92  
 transport\_tcp\_exclusiveaddr, 92  
 transport\_tcp\_interface, 89, 89  
 transport\_tcp\_listen\_backlog, 93  
 transport\_tcp\_maximum\_ports, 88, 89  
 transport\_tcp\_multiple\_receiver\_behavior, 93  
 transport\_tcp\_multiple\_receiver\_send\_order, 93  
 transport\_tcp\_nodelay, 94  
 transport\_tcp\_port, 89  
 transport\_tcp\_port\_high, 88, 90  
 transport\_tcp\_port\_low, 88, 90  
 transport\_tcp\_receiver\_socket\_buffer, 94  
 transport\_tcp\_reuseaddr, 95  
 transport\_tcp\_sender\_socket\_buffer, 95  
 transport\_topic\_sequence\_number\_info\_active\_threshold, 67  
 transport\_topic\_sequence\_number\_info\_interval, 68  
 Type, 211

## U

### UDP Port

- mim\_destination\_port (context), 132
- mim\_incoming\_destination\_port (context), 133
- mim\_outgoing\_destination\_port (context), 133
- resolver\_multicast\_incoming\_port (context), 83
- resolver\_multicast\_outgoing\_port (context), 84
- resolver\_multicast\_port (context), 84
- resolver\_unicast\_destination\_port (context), 223
- resolver\_unicast\_port (context), 223
- resolver\_unicast\_port\_high (context), 87
- resolver\_unicast\_port\_low (context), 87
- transport\_lbtrm\_destination\_port (source), 96
- transport\_lbtrm\_source\_port\_high (context), 97
- transport\_lbtrm\_source\_port\_low (context), 97
- transport\_lbtru\_maximum\_ports (context), 111
- transport\_lbtru\_port (source), 111
- transport\_lbtru\_port\_high (context), 111
- transport\_lbtru\_port\_high (receiver), 112
- transport\_lbtru\_port\_low (context), 112
- transport\_lbtru\_port\_low (receiver), 112
- ud\_acceleration, 131
- ume\_ack\_batching\_interval, 166
- ume\_activity\_timeout, 167, 167
- ume\_allow\_confirmed\_delivery, 167
- ume\_confirmed\_delivery\_notification, 168
- ume\_consensus\_sequence\_number\_behavior, 168, 169
- ume\_explicit\_ack\_only, 169
- ume\_flight\_size, 170
- ume\_flight\_size\_behavior, 170
- ume\_flight\_size\_bytes, 171
- ume\_force\_reclaim\_function, 171
- ume\_late\_join, 171
- ume\_message\_map\_tablesz, 225
- ume\_message\_stability\_notification, 172
- ume\_primary\_store\_address, 225
- ume\_primary\_store\_port, 225
- ume\_proxy\_source, 172
- ume\_receiver\_liveness\_interval, 173
- ume\_receiver\_paced\_persistence, 173, 173
- ume\_recovery\_sequence\_number\_info\_function, 174
- ume\_registration\_extended\_function, 174
- ume\_registration\_function, 175
- ume\_registration\_id, 225
- ume\_registration\_interval, 175, 175
- ume\_repository\_ack\_on\_reception, 175
- ume\_repository\_disk\_file\_size\_limit, 176
- ume\_repository\_size\_limit, 176
- ume\_repository\_size\_threshold, 177
- ume\_retention\_intergroup\_stability\_behavior, 177
- ume\_retention\_intragroup\_stability\_behavior, 178
- ume\_retention\_size\_limit, 178
- ume\_retention\_size\_threshold, 179
- ume\_retention\_unique\_confirmations, 179
- ume\_retransmit\_request\_generation\_interval, 179
- ume\_retransmit\_request\_interval, 180
- ume\_retransmit\_request\_maximum, 180
- ume\_retransmit\_request\_outstanding\_maximum, 180
- ume\_secondary\_store\_address, 226
- ume\_secondary\_store\_port, 226
- ume\_session\_id, 180, 181, 181
- ume\_source\_liveness\_timeout, 182
- ume\_state\_lifetime, 182, 182
- ume\_store, 182
- ume\_store\_activity\_timeout, 183
- ume\_store\_behavior, 183
- ume\_store\_check\_interval, 184
- ume\_store\_group, 184
- ume\_store\_name, 184
- ume\_tertiary\_store\_address, 226
- ume\_tertiary\_store\_port, 227
- ume\_user\_receiver\_registration\_id, 186
- ume\_use\_ack\_batching, 185
- ume\_use\_late\_join, 185
- ume\_use\_store, 186
- ume\_write\_delay, 186
- umq\_command\_interval, 187
- umq\_command\_outstanding\_maximum, 187
- umq\_delayed\_consumption\_report\_interval, 187
- umq\_flight\_size, 188, 188
- umq\_flight\_size\_behavior, 188, 189
- umq\_hold\_interval, 189
- umq\_index\_assignment\_eligibility\_default, 190
- umq\_message\_retransmission\_interval, 190
- umq\_message\_stability\_notification, 190, 191
- umq\_msg\_total\_lifetime, 191, 191
- umq\_queue\_activity\_timeout, 192
- umq\_queue\_check\_interval, 192
- umq\_queue\_name, 192
- umq\_queue\_participants\_only, 193
- umq\_queue\_participation, 193
- umq\_queue\_query\_interval, 194
- umq\_queue\_registration\_id, 194
- umq\_receiver\_type\_id, 194
- umq\_require\_queue\_authentication, 195

- umq\_retention\_intergroup\_stability\_behavior, 195, 196
- umq\_retention\_intragroup\_stability\_behavior, 196, 197
- umq\_retransmit\_request\_interval, 197
- umq\_retransmit\_request\_outstanding\_maximum, 198
- umq\_session\_id, 198
- umq\_ulb\_application\_set, 198
- umq\_ulb\_application\_set\_assignment\_function, 199
- umq\_ulb\_application\_set\_events, 199
- umq\_ulb\_application\_set\_load\_factor\_behavior, 199
- umq\_ulb\_application\_set\_message\_lifetime, 200
- umq\_ulb\_application\_set\_message\_max\_reassignments, 200
- umq\_ulb\_application\_set\_message\_reassignment\_timeout, 201
- umq\_ulb\_application\_set\_receiver\_activity\_timeout, 201
- umq\_ulb\_application\_set\_receiver\_keepalive\_interval, 201
- umq\_ulb\_application\_set\_round\_robin\_bias, 202
- umq\_ulb\_check\_interval, 202
- umq\_ulb\_events, 202
- umq\_ulb\_flight\_size, 203
- umq\_ulb\_flight\_size\_behavior, 203
- umq\_ulb\_receiver\_events, 204
- umq\_ulb\_receiver\_portion, 204
- umq\_ulb\_receiver\_priority, 205
- umq\_ulb\_source\_activity\_timeout, 205
- umq\_ulb\_source\_check\_interval, 205
- unrecognized\_channel\_behavior, 159
- use\_app\_header, 211
- use\_extended\_reclaim\_notifications, 68
- use\_index\_queuing, 212
- use\_late\_join, 145
- use\_otr, 148
- use\_transport\_thread, 68
- use\_ump\_session\_ids, 212
- resolver\_query\_max\_interval, 222
- resolver\_query\_minimum\_duration, 162
- resolver\_query\_minimum\_interval, 162

## W

- wait\_for\_source\_registration, 212
- Wildcard, 213
- wildcard\_receiver Scope Options
  - hf\_receiver, 216
  - pattern\_callback, 159
  - pattern\_type, 160
  - receiver\_create\_callback, 160
  - receiver\_delete\_callback, 161
  - resolver\_no\_source\_linger\_timeout, 161
  - resolver\_query\_maximum\_interval, 161