RTI Security Plugins

Release Notes

Version 7.1.0

Trademarks

Copy and Use Restrictions

Notices

Deprecations and Removals

Any deprecations or removals noted in this document serve as notice under the Real-Time Innovations, Inc. Maintenance Policy #4220 and/or any other agreements by and between RTI and customer regarding maintenance and support of RTI’s software.

Deprecated means that the item is still supported in the release, but will be removed in a future release. *Removed* means that the item is discontinued or no longer supported. By specifying that an item is deprecated in a release, RTI hereby provides customer notice that RTI reserves the right after one year from the date of such release and, with or without further notice, to immediately terminate maintenance (including without limitation, providing updates and upgrades) for the item, and no longer support the item, in a future release.

Early Access Software

"Real-Time Innovations, Inc. ("RTI") licenses this Early Access release software ("Software") to you subject to your agreement to all of the following conditions:

(1) you may reproduce and execute the Software only for your internal business purposes, solely with other RTI software licensed to you by RTI under applicable agreements by and between you and RTI, and solely in a non-production environment;

(2) you acknowledge that the Software has not gone through all of RTI's standard commercial testing, and is not maintained by RTI's support team;

(3) the Software is provided to you on an "AS IS" basis, and RTI disclaims, to the maximum extent permitted by applicable law, all express and implied representations, warranties and guarantees, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, satisfactory quality, and non-infringement of third party rights;

(4) any such suggestions or ideas you provide regarding the Software (collectively , "Feedback"), may be used and exploited in any and every way by RTI (including without limitation, by granting sublicenses), on a non-exclusive, perpetual, irrevocable, transferable, and worldwide basis, without any compensation, without any obligation to report on such use, and without any other restriction or obligation to you; and

(5) TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT WILL RTI BE LIABLE TO YOU FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, EXEMPLARY OR PUNITIVE OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR FOR LOST PROFITS, LOST DATA, LOST REPUTATION, OR COST OF COVER, REGARDLESS OF THE FORM OF ACTION WHETHER IN CONTRACT, TORT (INCLUDING WITHOUT LIMITATION, NEGLIGENCE), STRICT PRODUCT LIABILITY OR OTHERWISE, WHETHER ARISING OUT OF OR RELATING TO THE USE OR INABILITY TO USE THE SOFTWARE, EVEN IF RTI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES."

Technical Support

Real-Time Innovations, Inc.
232 E. Java Drive
Sunnyvale, CA 94089
Phone: (408) 990-7444
Email: support@rti.com
Website: https://support.rti.com/

Contents

Chapter 1 Supported Platforms

RTI® Security Plugins 7.1.0 is a feature release based on release 7.0.0.

See the columns for *Security Plugins* in the table of Supported Platforms for Platform-Dependent Products, in the RTI Connext Core Libraries Release Notes.

Chapter 2 Compatibility

This release of the *Security Plugins* includes partial support for the DDS Security specification from the Object Management Group (OMG).

The *Security Plugins* 7.1.0 are interoperable with the *Security Plugins* 5.2.7 and higher.

Persistence Service databases secured with the *Security Plugins* 7.1.0 are incompatible with databases generated by versions of *Persistence Service* older than 7.0.0.

When using the *Security Plugins SDK*, the required minimum version of CMake is 3.12 if linking dynamically and 3.13 if linking statically.

In release 7.1.0, the *Security Plugins* are available for use with OpenSSL® 1.1.1, OpenSSL 3.0, and wolfSSL® 5.5. There are separate installation packages for each of these options.

> For more information about other backward compatibility issues, see the *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation).

2.1 Compatibility with OpenSSL 1.1.1 and 3.0

The *Security Plugins* 7.1.0 for OpenSSL are API-compatible with OpenSSL 1.1.0 - 1.1.1t, and OpenSSL 3.0. The *Security Plugins 7.1.0* have only been tested by RTI using OpenSSL 1.1.1t and 3.0.8.

The *Security Plugins* SDK has been tested with OpenSSL 1.1.1t and 3.0.8.

Limitations when using OpenSSL

The *Security Plugins* for OpenSSL 3.0 do not support the OpenSSL Provider API, which is the OpenSSL 3.0 replacement for OpenSSL 1.1's Engine API. OpenSSL 3.0 no longer supports the Engine API.

2.2 Compatibility with wolfSSL 5.5

The *Security Plugins* 7.1.0 for wolfSSL have been tested with wolfSSL 5.5.1 on following target platform:

- QNX® Neutrino® 7.1 systems on Arm® v8 CPUs (RTI architecture: armv8QNX7.1qcc_gpp8.3.0)

Limitations when using wolfSSL

The *Security Plugins* for wolfSSL are interoperable with the *Security Plugins* for OpenSSL in most configurations. However, there are some features that are not supported by the *Security Plugins* for wolfSSL:

- Diffie-Hellman: The *Security Plugins* for wolfSSL only support the ECDHE-CEUM+P256 and ECDHE-CEUM+P384 Elliptic Curve Diffie-Hellman (ECDHE) key establishment algorithms.
- RSASSA-PSS-MGF1SHA256+2048+SHA256: The *Security Plugins* for wolfSSL support for digital signatures is limited to the RSASSA-PKCS1-V1_5+2048+SHA256, ECDSA-P256+SHA256, and ECDSA-P384+SHA384 algorithms.
- X.509 v3 key usage extensions: Enforcing the presence of the X.509 v3 extension keyUsage is not supported by the *Security Plugins* for wolfSSL.
- OpenSSL engines are not supported.

Chapter 3 What's New in 7.1.0

This section describes what's new, compared to the *RTI Security Plugins* 7.0.0.

3.1 Changes Related to Dynamic Participant Renewal, Revocation, and Expiration

3.1.1 Support for notification when the Local Participant's own certificate is about to expire

During *DomainParticipant* creation, the *Security Plugins* check that the desired certificate is currently valid, as described in Verifying the certificate validity on the current date and time, in the RTI Security Plugins User's Manual. When the certificate is about to expire, you may want to be notified so that you can replace the certificate with one that expires later. In this release, the *Security Plugins* do not yet support replacing the certificate, but they do support the notification mechanism, which is a DomainParticipantListener callback function combined with a property that configures how much advance notice you want.

For more information, see Dynamic Certificate Expiration of the Local DomainParticipant, in the Security Plugins User's Manual.

3.1.2 Support for kicking Remote Participants off a system because of an expired certificate

As described in Verifying the certificate validity on the current date and time, in the RTI Security Plugins User's Manual, when mutually authenticating with a remote *DomainParticipant*, the local *DomainParticipant* checks that the remote *DomainParticipant's* certificate is currently valid.

If the certificate is currently valid but later becomes expired, the local *DomainParticipant* may want to stop communicating with the remote *DomainParticipant*. In this release, the *Security Plugins* now support this behavior.

When the certificate expires, the local *DomainParticipant* will automatically and immediately remove the remote *DomainParticipant* and, if Key Revisions are enabled, it will regenerate and redistribute key material.

For more information, see Dynamic Certificate Expiration of Remote DomainParticipants, in the Security Plugins User's Manual.

3.2 Changes Related to Cryptography

3.2.1 Pre-Shared Key-based RTPS protection mechanism

This release introduces a new Pre-Shared Key-based RTPS protection mechanism, "RTPS PSK Protection." This is a Cryptography Plugin mechanism that supports basic communication protection, based on a pre-shared key that is distributed out-of-band to *DomainParticipants*.

RTPS PSK Protection does not require authentication. Consequently, it does not support more sophisticated security features such as granular-security and topic permissions enforcement. RTPS PSK Protection offers metadata and data protection on the wire and restricts communication to only participants holding the pre-shared, user-configurable key.

RTPS PSK Protection can be leveraged in two different ways:

- As part of the *RTI Security Plugins*: RTPS PSK Protection works alongside existing Security Plugins features and secures the communication that occurs before two participants authenticate each other.
- As part of *RTI Lightweight Security*: In this case, all traditional DDS Security mechanisms are disabled and the entire communication is protected with RTPS PSK Protection.

3.2.2 Support for Additional Authenticated Data (AAD) when using RTPS protection

If AAD is enabled, the RTPS Header and Header Extension (if present) submessages are passed as additional authenticated data to the encode AES operations. This means that the Security Plugins will check the integrity of those headers. In previous releases, the Security Plugins checked for the integrity of the RTPS Header in a different way. The main benefit of enabling AAD is that it reduces the size of the RTPS messages that we send on the wire.

If AAD is disabled, the Security Plugins behave as previously. The plugins include an INFO_SRC submessage (20 Bytes) right after the Header of the RTPS message. This submessage is protected (along with the others) using the algorithm given by the **com.rti.serv.secure.cryptography.encryption_algorithm** property. Doing so protects the integrity of the header data, at the expense of a few extra bytes on the wire.

AAD is disabled by default. You can enable it with the **com.rti.serv.secure.cryptography.enable_additional_authenticated_data** boolean property. The property must be TRUE if you are enabling the RTPS 2.5 Header Extension.

3.3 Changes Related to Performance and Scalability

3.3.1 RTI Lightweight Security

This release of the *Security Plugins* introduces Lightweight Security, a lightweight solution that uses a pre-shared key (distributed out-of-band) to protect the information. This new feature can be used with the OpenSSL 1, OpenSSL 3, and wolfSSL crypto libraries. The new library, **nddslightweightsecurity**, is included with the *Security Plugins* bundles.

Using pre-shared key protection, we can protect the confidentiality or integrity of the communication, without the overhead of authentication, key exchange, and enforcing permissions. Therefore, the RTI Lightweight Security library can be useful in resource-constrained scenarios.

The Lightweight Security library does not use the most demanding (CPU and memory wise) DDS Security mechanisms like authentication or access control. As a consequence of this, RTI Lightweight Security does not support more sophisticated security features like granular-security and topic permissions enforcement: it only protects against spoofing, tampering, and information disclosure from actors not holding the pre-shared, user-configurable key.

In this version of the *Security Plugins*, secure *DomainParticipants* skip authentication and access control. Instead, security is based on a per-participant, pre-shared key that protects all messages (including discovery). The *Security Plugins* derive the per-participant pre-shared key based on a seed that the user must set consistently across the whole system. The property for configuring the seed is **com.rti.serv.secure.cryptography.rtps_protection_preshared_key**. The entire communication is protected by default using the AES256+GCM cryptographic algorithm. You can choose another algorithm with the **com.rti.serv.secure.cryptography.rtps_protection_preshared_key_algorithm property**. The available options are AES128+GCM, AES256+GCM, AES128+GMAC, and AES256+GMAC.

Note that *DomainParticipants* from the Lightweight Security library are not interoperable with those from the full *Security Plugins* (**nddssecurity**).

For more information, see Lightweight Security, in the Security Plugins User's Manual

3.4 Changes Related to APIs

3.4.1 Information from the Trust Plugins added to builtin topic data in Java API

The ParticipantBuiltinTopicData, PublicationBuiltinTopicData, and SubscriptionBuiltinTopicData entities contain two new fields with data from the Trust Plugins:

- **trust_algorithm_info** has the algorithms associated with the discovered *DomainParticipant*.
- **trust_protection_info** has data that is dependent on the Trust Plugins implementation.

Connext 7.0.0 introduced these two fields in the C, traditional C++, and modern C+ APIs. *Connext* 7.1.0 added these fields to the Java API.

For more information, see Relevant Connext APIs, in the Security Plugins User Manual. The section on the discovered_participant_data API describes these types and includes some relevant links.

3.4.2 Changes to Trust APIs to match future DDS Security specification with respect to Security Algorithm Info and Security Protection Info

This release updates several of the types introduced in *Connext* 7.0.0, to match the future DDS Security specification. In particular, the wire representation and user-level API types associated with Cryptographic Algorithms configuration (Trust Algorithms Info, Security Algorithm Info) have been updated. The user-level API types associated with the *Security Plugins* configuration (Trust Protection Info, Security Protection Info) have also been updated.

For more information, see:

- The API Reference documentation
- Relevant Types for the Governance Document, in the Security Plugins User's Manual
- Relevant Types for the Security Algorithms, in the Security Plugins User's Manual

3.5 Changes Related to Debuggability

3.5.1 Adjusted verbosity of several security event logged messages

This release updates the verbosity level of several security event logged messages. In particular, security event logged messages now follow this schema:

- DDS_LOGGING_EMERGENCY_LEVEL: Used to log fatal error conditions that prevent RTI Security Plugins from continuing to run properly.
- DDS_LOGGING_ALERT_LEVEL: Used to log security alerts. Usually derived from a remote peer not being properly configured or being malicious.

- DDS_LOGGING_CRITICAL_LEVEL: Used to log critical, unexpected errors. In most cases, these errors will be triggered by the local host running out of resources. While the *Security Plugins* can continue operating, it is likely that new errors will continue to be triggered.
- DDS_LOGGING_ERROR_LEVEL: Used to log error conditions.
- Higher verbosity levels: Used to log non-error conditions, from warnings to informative messages.

3.6 Changes Related to Third-Party Software

3.6.1 Upgraded OpenSSL to versions 1.1.1t and 3.0.8

The following third-party software, used by the *Security Plugins*, has been upgraded:

Third-party Tool	Old Version	New Version
OpenSSL	1.1.1n	1.1.1t 3.0.8

The *Security Plugins* now support the latest LTS version of OpenSSL (OpenSSL 3.0). In this release, the *Security Plugins* are available as both a set of **nddssecurity** libraries built against OpenSSL 1.1.1t (supported until September 2023) and a set of **nddssecurity** libraries built against OpenSSL 3.0.8 (supported until September 2026).

OpenSSL 3.0 has replaced the Engine API with the Provider API (see https://www.openssl.org/docs/man3.0/man7/migration_guide.html and search for the 'Engines and "METHOD" APIs' section).

If you are using OpenSSL Engines (see https://community.rti.com/static/documentation/connext-dds/7.1.0/doc/manuals/connext_dds_secure/users_manual/p3_advanced/openssl_engines.html), please note that the *Security Plugins* do not support providers (see https://www.openssl.org/docs/man3.0/man7/provider.html).

See the *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation) for migration issues related to this upgrade.

3.6.2 Upgraded to wolfSSL 5.5.1

The *Security Plugins* for wolfSSL are now based on, and API-compatible with, wolfSSL version 5.5.1 (no earlier versions).

For this release, the *Security Plugins* for wolfSSL have only been tested by RTI using wolfSSL 5.5.1.

3.7 Changes Related to System Extensibility and Configurability

3.7.1 Deprecated properties related to the propagation of security info

Communication between *DataWriters* and *DataReaders* using inconsistent Governance Topic-Level Rules is not compliant with the DDS Security Specification.

Likewise, configuring *DomainParticipants* within the same domain with inconsistent Governance Domain-Level Rules is also not compliant with the DDS Security Specification.

Both of these scenarios can make the system more vulnerable to attackers. Therefore the following properties have been deprecated:

- **dds.participant.discovery_config.disable_endpoint_security_info_propagation**
- **dds.participant.discovery_config.disable_participant_security_info_propagation**

Support for these properties may be removed in future versions of the *Security Plugins*. Using these properties is highly discouraged.

3.8 Changes Related to Supported Platforms

3.8.1 New Platforms

This release adds support for this platform:

- Red Hat® Enterprise Linux® 9 on x64 (x64Linux4gcc7.3.0)

3.8.2 Removed Platforms

The following platforms are no longer supported:

- macOS® 10.13, 10.14, 10.15
- VxWorks® 21.11

3.9 Changes Related to Shipped Examples

3.9.1 Support building shipped examples using different crypto libraries

This release adds supports for compiling the security shipped examples (the C, C++ and Java hello_ security examples, and the hello_banish C example) using any of the available crypto libraries (OpenSSL 3.0, OpenSSL 1.1.1, or wolfSSL 5.5). Use the crypto library matching your installation of the *Security Plugins*.

The examples for Windows® systems now include new build modes, so that you choose the crypto library.

On Linux and macOS systems, you can indicate the crypto library as a parameter of the **make** command when compiling the example. Please see the **hello_security READ_ME.txt** files for more details.

3.9.2 Security examples now support secp384r1 curve

The hello_security examples now accept "p384" as the third command-line argument, whereas they previously only accepted the "rsa" value. The publisher or subscriber application will create a *DomainParticipant* that uses ECDHE-CEUM+P384 for key establishment and ECDSA+P384+SHA384 for digital signatures. For examples of commands to generate ECDSA secp384r1 certificates, see the Getting Started Guide, Hands-on 4.

3.9.3 New example for banishing participants

There is a new C example that demonstrates how to use:

- **DDS_DomainParticipant_get_discovered_participant_subject_name()**
- **DDS_DomainParticipant_get_discovered_participants_from_subject_name()**
- **DDS_DomainParticipant_banish_ignored_participants()**

You can find the example in **<path to examples>/connext_dds/c/hello_banish**. See Relevant Connext APIs, in the Security Plugins User Manual for more information.

3.10 Other Changes

3.10.1 Redaction of sensitive properties when logging DDS Entities' PropertyQos configuration

Connext has the ability to log the DDS Entity QoS configuration when a DDS Entity is created and when the DDS Entity QoS is set. The logged information includes all the Entity's PropertyQos properties that have non-default values.

This release now redacts the values of sensitive properties (for example, those containing cryptographic keys) before they are output to the log. For example, logging the **dds.sec.auth.private_key property** will result in the following output:

```
...
<element>
    <name>dds.sec.auth.private_key</name>
    <value>[redacted]</value>
</element>
...
```

Connext considers as sensitive any property that ends with any of the following suffixes:

- ".cryptography.key"
- ".internal_license_string"
- ".internal_license_validation"
- ".key_material_key"
- ".license_file"
- ".license_string"
- ".participant_discovery_protection_key"
- ".password"
- ".private_key"
- ".private_key_file"
- ".private_key_password"
- ".rsa_private_key"
- ".rsa_private_key_file"
- ".rtps_protection_key"

Chapter 4 What's Fixed in 7.1.0

4.1 Fixes Related to Discovery and Authentication

4.1.1 Rare 'copy failure' error while getting participant details before Discovery completed

If you called the API function **get_discovered_participant_data()** or **get_discovered_participant_subject_name()** on a *DomainParticipant* while it was in the process of discovering other *DomainParticipants* or their endpoints, then in rare cases, the *DomainParticipant* failed to discover other *DomainParticipants* or their endpoints. An accompanying error message referred to **PRESParticipant_onSecurityChannelWriteEvent** or **PRESParticipant_processMatchedRemoteEndpointSecurity** and a failure to copy a remoteParticipant table.

This problem, which might have prevented communication between the two involved *DomainParticipants* for one or more of their *Topics*, has been fixed. This error message will no longer occur, and discovery will no longer fail due to this error message.

[RTI Issue ID SEC-1779]

4.1.2 Missing security information in the Participant Builtin Topic data

Connext 7.0.0 introduced relevant security information as part of the *DomainParticipant's* builtin topic data. The 7.0.0 release added two new fields: **trust_algorithms** (renamed in 7.1.0 to **trust_algorithm_info**) and **trust_info** (renamed in 7.1.0 to **trust_protection_info**). In 7.0.0, you were able to retrieve this information using the **discovered_participant_data()** API.

The security information should also have been available through the samples of the *DomainParticipant's* builtin *Subscriber*. This was not the case in *Connext* 7.0.0.

This problem has been resolved. Now you can get the *DomainParticipant's* builtin Topic data using the **on_data_available()** callbacks for its builtin *Subcriber*, and the **trust_algorithm_info** and **trust_protection_info** fields will be correctly populated.

[RTI Issue ID SEC-1871]

4.1.3 Unbounded memory growth and 'deadlock risk' error when deleting a DataWriter matched with a DataReader on same DomainParticipant

This problem applied to *DataWriters* that were created with a Governance Document whose **metadata_protection_kind** or **data_protection_kind** for the *DataWriter's* topic was a value other than NONE.

If you deleted a *DataWriter* matched with a *DataReader* on the same *DomainParticipant*, and the PublisherQos of the *DataWriter'sPublisher* did not have **exclusive_area.use_shared_exclusive_area** set to true, it was possible to see a 'deadlock risk' error about failing to enter level 20 from level 30. This error indicated a failure to free memory, and continuing to create and delete *DataWriters* could have led to unbounded memory growth. This problem was more likely to occur if the *DataWriter* and *DataReader* had compatible types and matching topics, but had some other kind of incompatibility. This problem has been resolved.

[RTI Issue ID SEC-1883]

4.2 Fixes Related to Cryptography

4.2.1 Session keys renewed half as frequently as they should have been

The *Security Plugins* update the session keys after protecting some message blocks. The **cryptography.max_blocks_per_session** property determines how many message blocks can be encrypted using the same session key.

However, the **cryptography.max_blocks_per_session's** effective value depended on the **cryptography.encryption_algorithm** property. In the case of AES256+GCM, the effective value was double the property value. In the case of AES192+GCM, the effective value was 1.5 times the property value. The issue did not affect AES128+GCM. This problem occurred for all protection types. See 4.2.4 Session keys were not renewed as often as they should when using RTPS SIGN protection on the next page for further overuse of session keys affecting only RTPS SIGN protection.

The issue has been fixed.

[RTI Issue ID SEC-1231]

4.2.2 data_protection_kind = SIGN was sometimes treated as ENCRYPT

For a given topic, if the Governance Document tag **data_protection_kind** had a value of SIGN and either of the following conditions was true, the serialized payload was mistakenly encrypted:

- The Governance Document tag **metadata_protection_kind** had a value of ENCRYPT.
- **metadata_protection_kind** had a value of SIGN and the *DomainParticipant's* PropertyQosPolicy **cryptography.share_key_for_metadata_and_data_protection** had a value of FALSE.

This problem has been fixed. The serialized payload is now unencrypted (protected with AES-GMAC) in the above scenarios.

[RTI Issue ID SEC-1773]

4.2.3 Possible crash when disable_endpoint_security_info_propagation was true

A *DataReader* may have crashed due to a race condition when the following conditions were met:

- **dds.participant.discovery_config.disable_endpoint_security_info_propagation** was set to true (see the RTI Connext Migration Guide).
- A *DataReader's DomainParticipant's* Governance Document had **metadata_protection_kind** set to NONE
- A matched *DataWriter's DomainParticipant's* Governance Document had **metadata_protection_kind** set to something other than NONE (which is a configuration allowed by this version of *Connext* but that is not compliant with OMG DDS Security specification, and therefore discouraged).

A memory checking tool such as Valgrind™ would have reported invalid reads in a function due to accessing an address freed by a different function. This problem has been fixed.

[RTI Issue ID SEC-1747]

4.2.4 Session keys were not renewed as often as they should when using RTPS SIGN protection

The Security Plugins update the session keys after protecting some message blocks. The **cryptography.max_blocks_per_session** property determines how many message blocks can be encrypted using the same session key.

However, **cryptography.max_blocks_per_session** had an effective value larger than the property value when using RTPS SIGN (or SIGN_WITH_ORIGIN_AUTHENTICATION) protection. The problem led to slightly overused session keys in some scenarios. This issue only affected *Security Plugins* 7.0.0 and has been fixed.

[RTI Issue ID SEC-1786]

4.2.5 Value AES192+GCM for cryptography.encryption_algorithm did not work

Connext 7.0.0 introduced the following values for the **cryptography.encryption_algorithm** property: AES128+GCM, AES192+GCM, and AES256+GCM. These new values are meant to replace but still coexist with the legacy ones: aes-128-gcm, aes-192-gcm, and aes-256-gcm.

However, the AES192+GCM choice did not work correctly. The workaround for setting the AES-192 symmetric cipher algorithm was to use the aes-192-gcm legacy value. This issue has been fixed.

[RTI Issue ID SEC-1806]

4.2.6 Setting wrong value for symmetric cipher algorithm failed silently

In release 7.0.0, configuring the **cryptography.encryption_algorithm** property with a wrong value failed silently. In these cases, the final value of the property was AES256+GCM (the default). This problem has been resolved. Now if the property is set to a wrong value, there will be a failure during *DomainParticipant* creation.

[RTI Issue ID SEC-1807]

4.2.7 Race conditions related to banish_ignored_participants may have caused crashes or decoding errors

The **banish_ignored_participants()** API (introduced in *Security Plugins* 7.0.0) had several concurrency problems that led to potential crashes or decoding errors. These problems may have occurred during deletion of a local *DataWriter* or *DataReader,* or when a *DataWriter's* key material for Submessage Protection was different from its key material for Serialized Data Protection (see share key for metadata and data protection, in Design Considerations, in the Security Plugins User's Manual).

These problems have been resolved.

[RTI Issue ID SEC-1825]

4.2.8 Communication failure when using origin authentication and banish_ignored_participants

If you set the Governance document tag **rtps_protection_kind** or **metadata_protection_kind** to SIGN_WITH_ORIGIN_AUTHENTICATION or ENCRYPT_WITH_ORIGIN_AUTHENTICATION, and you successfully called the API **banish_ignored_participants()**, you would have experienced a persistent communication failure. This failure was accompanied by this error message:

```
RTI_Security_Cryptography_verifyReceiverSpecificMac:
OpenSSL function EVP_DecryptFinal_ex (GMAC) failed with error:
(error details not available).
```

This problem has been resolved.

[RTI Issue ID SEC-1862]

4.2.9 Communication failure when using origin authentication and max_blocks_per_session

If you set the Governance document tag **rtps_protection_kind** or **metadata_protection_kind** to SIGN_WITH_ORIGIN_AUTHENTICATION or ENCRYPT_WITH_ORIGIN_AUTHENTICATION, you would have experienced a persistent communication failure when the Session Keys were changed due to the property **cryptography.max_blocks_per_session** (see Limiting the usage of a specific session key, in the Security Plugins User's Manual). This failure was accompanied by an error message such as:

```
DecryptFinal failed. Possible GCM authentication failure
```

This problem has been resolved.

[RTI Issue ID SEC-1863]

4.2.10 Potential invalid read while decoding encrypted messages

In previous releases, receiving a malformed, protected RTPS message may have resulted in invalid memory reads or, in very rare crashes, in a crash. This issue, which did not impact the confidentiality or integrity of *Connext* applications, has been fixed

[RTI Issue ID SEC-1892]

4.3 Fixes Related to Reliability Protocol and Wire Representation

4.3.1 Unexpected error 'Fragment data not supported by this writer'

In *Connext* 7.0.0, you may have seen the following error when trying to run an application that had set the **dds.participant.protocol.rtps_overhead** property and it was using the *Security Plugins*. The same configuration did not fail in previous releases.

```
{noformat}ERROR COMMENDFacade_canSampleBeSent:NOT SUPPORTED | Fragment data not supported by
this writer.{noformat}
```

To workaround the issue, you could have removed the property **dds.participant.protocol.rtps_overhead** from the Participant's configuration. This is also the recommended configuration starting with 7.0.0, as the overhead is automatically calculated by the middleware. This problem has been resolved.

[RTI Issue ID SEC-1813]

4.4 Fixes Related to Entities

4.4.1 Error creating participant with specific local GUID prefixes using security

An error occurred if a participant had a hostId, appId, or instanceId set to zero.

[RTI Issue ID SEC-1835]

4.5 Fixes Related to Shipped Examples

4.5.1 DomainParticipantQoS in C hello_security was not finalized

When using static libraries in the C **hello_security** example, the DomainParticipantQoS was not being finalized. This caused memory leaks for the QoS. This issue has been fixed by properly finalizing the DomainParticipantQoS at the end of execution.

[RTI Issue ID SEC-1699]

4.6 Fixes Related to Vulnerabilities

4.6.1 Submessage Protection was ineffective at protecting against submessage tampering

Submessage Protection was ineffective at protecting against submessage tampering. This problem has been resolved.

A vulnerability in the *Connext* application could have resulted in the following:

- An attacker was able to bypass Submessage Protection authentication (enabled with metadata_protection_kind set to SIGN, ENCRYPT, SIGN_WITH_ORIGIN_AUTHENTICATION, or ENCRYPT_WITH_ORIGIN_AUTHENTICATION) and inject untrusted submessages to the system.
- Still, an attacker was not able bypass Submessage Protection encryption (enabled with metadata_protection_kind set to ENCRYPT or ENCRYPT_WITH_ORIGIN_AUTHENTICATION) to read protected submessages.
- Remotely exploitable only if rtps_protection_kind was set to NONE or if a trusted Domain Participant was already compromised by a previous attack.
- Potential impact on integrity of *Connext* application.
- CVSS Base Score: 9.1 CRITICAL
- CVSS v3.1 Vector: AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:H

Note that this vulnerability is only fixed as long as the (now deprecated and discouraged to be used) participant's property **disable_endpoint_security_info_propagation** is set to FALSE, which is the default value.

[RTI Issue ID SEC-1887]

4.6.2 Potential Denial of Service when using OpenSSL 1.1.1 due to a vulnerability in OpenSSL 1.1.1

The *Security Plugins* had a third-party dependency on OpenSSL 1.1.1, which is known to be affected by a number of publicly disclosed vulnerabilities. These vulnerabilities have been fixed by upgrading OpenSSL to the latest stable version, 1.1.1t.

User Impact without Security: No impact.

User Impact with Security: The impact to *Security Plugins* applications when using the previous version was as follows:

- Exploitable by triggering the parsing of malicious certificates that need to be checked against a CRL obtained from a CRL distribution point.
- The application could hang.
- CVSS Base Score: 7.5 HIGH
- CVSS v3.1 Vector: AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

This issue has been fixed.

[RTI Issue ID SEC-1955 THIRDPARTY-70]

4.6.3 Authentication handshake did not effectively protect against GUID impersonation

The authentication handshake was ineffective at protecting against GUID impersonation. This problem has been resolved.

User Impact without Security: No impact. This issue is only applicable when using Security.

User Impact with Security: A vulnerability in the *Connext* application could have allowed an attacker to bypass any user-level dynamic access control built around GUIDs. As a result, other *DomainParticipants* would have accepted an attacker using the wrong GUID. The user impact was as follows:

- Remotely exploitable
- Potential impact on integrity of *Connext* application
- CVSS Base Score: 9.8 CRITICAL
- CVSS v3.1 Vector: AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

[RTI Issue ID SEC-1988]

4.7 Fixes Related to Security Plugins SDK

4.7.1 Instructions on statically building Security Plugins SDK referred to the wrong cmake version

In release 6.1.1, the *Security Plugins* SDK introduced a buildable test suite, which allows you to validate the *Security Plugins* source code. In order to build the test suite statically (`-DBUILD_SHARED_LIBS=OFF`), a cmake version of at least 3.13 is required.

However, the documentation previously stated that the minimum cmake version was 3.12. This documentation issue is now fixed. The requirements now specify that if you are compiling the SDK statically, the minimum cmake version is 3.13.

[RTI Issue ID SEC-1884]

4.8 Fixes Related to Crashes

4.8.1 Potential crash while decoding protected submessages

Release 6.1.1 introduced several performance optimizations to Submessage Protection decoding. There was an issue with one of these optimizations, potentially resulting in a rare crash on the receiver (*DataWriter* or *DataReader*) while decoding a protected submessage.

In particular, this issue was triggerable if any of the following were true for at least one *DataWriter-/DataReader* pair:

- **metadata_protection_kind** set to a value different from NONE
- **discovery_protection_kind** set to a value different from NONE and **enable_discovery_protection** is TRUE
- **liveliness_protection_kind** set to a value different from NONE and **enable_liveliness_protection** is TRUE

This issue was more likely to trigger when the sender's *DomainParticipant* was deleting all of its endpoints. This issue has been fixed; decoding protected submessages no longer results in a crash.

[RTI Issue ID SEC-1960]

Chapter 5 Previous Release

5.1 What's New in 7.0.0

This section describes what's new, compared to the *RTI Security Plugins* 6.1.1.

This release adds a set of new features and improvements that will enable your *Connext Secure* applications with two key capabilities:

- **Seamlessly Regenerate and Redistribute Key Material**

  The *Security Plugins* now support a mechanism to regenerate and redistribute the Key Material without needing to recreate the involved *DomainParticipants* or lose liveliness. This mechanism enables securely kicking *DomainParticipants* out of a system. Future releases will add additional ways to trigger key regeneration and redistribution. The specific new features related to this are described in 5.1.1 Changes Related to Dynamic Participant Renewal, Revocation, and Expiration on the next page.

- **Meet Commercial National Security Algorithm (CNSA) Suite TOP-SECRET Level Requirements**

  The *Security Plugins* can now operate at CNSA Suite TOP-SECRET level. In particular, this release adds support for secp384r1 key-establishment and digital-signature algorithms. The extended algorithm support is complemented with:

  - A new mechanism for early detection of cryptographic algorithms compatibility during the discovery phase.
  - A new Governance Document-based mechanism to restrict which cryptographic algorithms are authorized to be used within a DDS system.

  The specific new features related to this are described in 5.1.2 Changes Related to Cryptographic Algorithms on page 22 and 5.1.3 Changes Related to System Extensibility and Configurability on page 24.

This section includes descriptions of products, features, and platforms that are *deprecated* or *removed* starting in release 7.0.0.

Deprecated means that the item is still supported in this release, but will be removed in a future release. *Removed* means that the item is discontinued or no longer supported. By specifying that an item is deprecated in this release, RTI is hereby providing customer notice that RTI reserves the right after one year from the date of this release and, with or without further notice, to immediately terminate maintenance (including without limitation, providing updates and upgrades) for the item, and no longer support the item, in a future release.

This section serves as notice under the Real-Time Innovations, Inc. Maintenance Policy #4220 and/or any other agreements by and between RTI and customer regarding maintenance and support of RTI's software.

5.1.1 Changes Related to Dynamic Participant Renewal, Revocation, and Expiration

5.1.1.1 Support for kicking Participants off a system

As described in Limiting the Usage of Specific Key Material (Section 50.1.1), in the *RTI Security Plugins User's Manual*, the **cryptography.max_blocks_per_session** property is not useful for kicking participants off the system, because the original Key Material stays the same.

In this release, the *Security Plugins* now support a mechanism to regenerate and redistribute the Key Material without needing to recreate the involved *DomainParticipants* or losing liveliness. During a key regeneration and redistribution event, information to derive new Key Material is propagated over the Secure Key Exchange Channel to all currently legitimate remote *DomainParticipants*. When those *DomainParticipants* acknowledge this information, the old Key Material will no longer be used to encode new content, thus banishing formerly legitimate remote *DomainParticipants*, without negatively impacting communication with trusted *DomainParticipants*.

In this first release of this feature, key regeneration and redistribution can be triggered by calling the new *DomainParticipant* function, **banish_ignored_participants()** (see 5.1.1.2 New API for kicking Participants off a system on the next page). Future releases will add other ways to trigger key regeneration and redistribution.

This feature introduces new properties:

- **dds.participant.trust_plugins.key_revision_max_history_depth**
- **dds.participant.trust_plugins.max_key_redistribution_delay.sec**

To enable this feature, you must set the property **dds.participant.trust_plugins.key_revision_max_history_depth** to a non-zero value. A *DomainParticipant* that sets this property to a non-zero value will

not communicate with a *DomainParticipant* that sets this property to 0, or with a *DomainParticipant* of a release older than *Security Plugins* 7.0.0.

See Limiting the Usage of Specific Key Material (Section 50.1.1), in the *RTI Security Plugins User's Manual* for more information.

5.1.1.2 New API for kicking Participants off a system

This release adds a new API to kick *DomainParticipants* off a system, **DomainParticipant::banish_ignored_participants()**. This API complements **DomainParticipant::ignore_participant()**, which prevents the local *DomainParticipant* from processing traffic from the remote *DomainParticipant*. This new method prevents already ignored remote *DomainParticipants* from processing traffic from the local *DomainParticipant*.

You can use **DomainParticipant::banish_ignored_participants()** in combination with the key regeneration and redistribution capabilities of the *Security Plugins*. See Limiting the Usage of Specific Key Material (Section 50.1.1), in the *RTI Security Plugins User's Manual* for more information.

5.1.2 Changes Related to Cryptographic Algorithms

5.1.2.1 Support for secp384r1 key-establishment and digital-signature algorithms

This release introduces support for new key-establishment and digital-signature algorithms. The supported key-establishment algorithms now include Elliptic Curve Diffie-Hellman in Ephemeral mode with secp384r1 as its curve (**ECDHE-CEUM+P384**). There is also support for digital signatures using ECDSA secp384r1 key-pairs with SHA-384 (**ECDSA+P384+SHA384**). Note that these algorithms are still not part of the DDS Security Specification.

5.1.2.2 Changes to property that configures key-establishment algorithm

The property **authentication.shared_secret_algorithm** has been renamed to **authentication.key_establishment_algorithm**. (The former name still works, but is now deprecated and may be removed in a future release). The previously supported values (**dh** and **ecdh**) are also deprecated. See below for replacement values.

The new property, **authentication.key_establishment_algorithm**, supports these values:

- **DHE+MODP-2048-256:** Replaces **dh**.
- **ECDHE-CEUM+P256:** Replaces **ecdh**.
- **ECDHE-CEUM+P384:** The key establishment algorithm is Elliptic Curve Diffie-Hellman in Ephemeral mode with secp384r1 as its curve.

- **AUTO:** The *Security Plugins* will detect the algorithm from the Identity's private key. If the private key is Elliptic, with a NIST P-384 curve, the algorithm is set to **ECDHE-CEUM+P384**; otherwise, the algorithm is set to **ECDHE-CEUM+P256**.

5.1.2.3 Removed support for Digital Signature Algorithm (DSA)

In previous releases, Digital Signature Algorithm (DSA) support was deprecated. In this release, the DSA support is removed from the *Security Plugins*. As a result, the *Security Plugins* now require replacing DSA with one of the supported algorithms (see Cryptographic Algorithms Used for Digital Signatures for more information).

5.1.2.4 Added experimental support for ED25519, ED448, X25519, and X448

This release adds experimental support for two new digital signature algorithms (EDDSA+ED25519+SHA512, EDDSA+ED448+SHAKE256) and two key establishment algorithms (ECDHE-CEUM+X25519, ECDHE-CEUM+X448). Support for these new algorithms is disabled by default; it can be enabled through the following new property:

- **com.rti.serv.secure.authentication.enable_custom_algorithms**

This new property configures whether to enable custom cryptographic algorithms for the Authentication plugin. When enabled (not by default) the *Security Plugins* will enable additional digital signature and key establishment algorithms that are not part of the DDS Security specification (EDDSA+ED25519+SHA512, EDDSA+ED448+SHAKE256, ECDHE-CEUM+X25519, ECDHE-CEUM+X448).

This property is currently only supported in combination with OpenSSL; it will have no effect when used in combination with wolfSSL.

5.1.2.5 Changed default symmetric cipher algorithm to AES256+GCM

The AES symmetric keys used by the Cryptography Plugin to protect the confidentiality, integrity, and authenticity of messages are now, by default, 256-bits long (AES256+GCM). The previous default length for these keys was 128 bits. The change in the default behavior does not affect compatibility with previous releases: *DomainParticipants* using different size AES symmetric keys interoperate with no issues. You can modify the length of the keys to use 128 bits by setting the **cryptography.encryption_algorithm** property to AES128+GCM.

5.1.3 Changes Related to System Extensibility and Configurability

5.1.3.1 Information about supported and used cryptographic algorithms propagated in discovery

Secure *DomainParticipants* now propagate information about their supported and used cryptographic algorithms during discovery. This information is used to determine matching between different *DomainParticipants*, matching between different Endpoints, and for early detection of configuration issues.

DomainParticipants propagate the following information:

- PID_PARTICIPANT_SECURITY_DIGITAL_SIGNATURE_ALGO: Supported and used identity trust chain and authentication algorithms
- PID_PARTICIPANT_SECURITY_KEY_ESTABLISHMENT_ALGO: Supported and preferred key establishment algorithms
- PID_PARTICIPANT_SECURITY_SYMMETRIC_CIPHER_ALGO: Supported and used symmetric cipher algorithms for builtin endpoints traffic and key exchange
- PID_ENDPOINT_SECURITY_SYMMETRIC_CIPHER_ALGO: Symmetric cipher algorithm used by an endpoint to encode its traffic

If any of the PIDs values are set to defaults, or if security is disabled, they are not propagated. The defaults are compatible with previous *Security Plugins* releases: communication with earlier releases is not impacted.

Compatibility Rules

The following rules determine if two *DomainParticipants*, PA and PB, are compatible with respect to these cryptographic algorithms:

- Identity trust chain digital signature algorithms
  - PA’s supported algorithms intersect with any bit from PB’s used algorithm, and
  - PB’s supported algorithms intersect with any bit from PA’s used algorithm.
- Authentication digital signature algorithms
  - PA’s supported algorithms intersect with PB’s used algorithm, and
  - PB’s supported algorithms intersect with PA’s used algorithm.
- Key establishment algorithms
  - PA’s supported algorithms intersect with PB’s preferred algorithm, and
  - PB’s supported algorithms intersect with PA’s preferred algorithm.

- Symmetric cipher algorithms
  - PA's supported algorithm intersects with PB's used algorithm, and
  - PB's supported algorithm intersects with PA's used algorithm, and
  - PA's builtin endpoint key exchange algorithm is equal to PB's builtin endpoint key exchange algorithm.
- Two endpoints, EPA and EPB, are compatible if:
  - PA's supported symmetric cipher algorithms intersect with EPB's used algorithm, and
  - PB's supported symmetric cipher algorithms intersect with EPA's used algorithm.

5.1.3.2 Ability to configure system-wide allowed security algorithms

There is a new XML element in the Governance Document: **<allowed_security_algorithms>**. This element determines the security algorithms that are allowed in your system. There are four families of algorithms. You can specify the list of approved system-wide algorithms for each of the families:

- **<digital_signature>**

  Configures the Digital signature algorithms that *DomainParticipants* can use for generating and validating signatures during the authentication process. Unless **<digital_signature_identity_trust_chain>** is set, **<digital_signature>** also configures the Digital signature algorithms that *DomainParticipants* can use in the context of the identity trust chain. These are the algorithms that are allowed when verifying the Identity Certificate (local or remote) against the Identity Certificate Authority.

  Possible values:
  - RSASSA-PSS-MGF1SHA256+2048+SHA256
  - RSASSA-PKCS1-V1_5+2048+SHA256
  - ECDSA+P256+SHA256
  - ECDSA+P384+SHA384

- **<digital_signature_identity_trust_chain>**

  If set, overwrites the configuration of **<digital_signature>** for configuring the Digital signature algorithms that *DomainParticipants* can use in the context of the identity trust chain. These are the algorithms that are allowed when verifying the Identity Certificate (local or remote) against the Identity Certificate Authority.

  Possible values:

- RSASSA-PSS-MGF1SHA256+2048+SHA256
- RSASSA-PKCS1-V1_5+2048+SHA256
- ECDSA+P256+SHA256
- ECDSA+P384+SHA384

- **<key_establishment>**

  Algorithms that *DomainParticipants* can use for key establishment.

  Possible values:
  - DHE+MODP-2048-256
  - ECDHE-CEUM+P256
  - ECDHE-CEUM+P384

- **<symmetric_cipher>**

  Algorithms that *DomainParticipants* and their endpoints can use for symmetric cipher operations.

  Possible values:
  - AES128+GCM
  - AES256+GCM

Secure *DomainParticipants* propagate their list of *supported+approved* algorithms during discovery. Two *DomainParticipants* will match or not, depending on their *supported+approved* algorithms. They will try to authenticate each other only if they match.

To allow *DomainParticipants* in your system to use any supported security algorithm, do *not* add the **<allowed_security_algorithms>** XML element to the Governance Document. In that case, the only restriction comes from the implementation of the *Security Plugins*. For example, a particular crypto library may not support some algorithms. The *Security Plugins* will internally populate the supported algorithms and let other *DomainParticipants* know about them during discovery.

5.1.3.3 New XML attribute to improve version compatibility of Governance and Permissions Documents

This release introduces support for the **must_interpret** XML attribute. This attribute improves the backward and forward compatibility of the Governance and Permissions Documents.

XML elements that have the **must_interpret** attribute set to false will not trigger a validation failure of the XML parser. Add **must_interpret="false"** to the elements of your Governance or Permissions Document that are not supported in other *Connext* releases. Only the versions of the *Security Plugins*

that understand these elements will interpret them. Others will ignore the elements when parsing the XML file.

If **must_interpret** is not specified, its default value is "true"—the XML parser validates the element as in previous releases.

Using **must_interpret** in your Governance of Permissions Document breaks compatibility with versions of the *Security Plugins* before 7.0.0. For more information, see:
—The *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation),
—How the Governance File is Interpreted, Section 3.3.5 in the *RTI Security Plugins User's Manual*, and
—How the XML is Validated, Section 50.9, in the *RTI Connext Core Libraries User's Manual*.

5.1.4 Changes Related to Performance and Scalability

5.1.4.1 Improved throughput when batching protected data

When enabling batching and data protection, the data protection is now applied to the entire batch instead of to the individual samples within the batch. This change introduces two improvements:

- The combination of compression, batching, and data protection is now supported. First the batch will be compressed, then the compressed batch will be protected.
- The throughput of batching and data protection has been improved because the overhead of data protection only appears once per batch.

5.1.4.2 Added optional custom allocator for Security Plugins for OpenSSL

This release adds the ability to set custom allocators for the *Security Plugins* loaded crypto library. In particular, this release adds a new custom allocator for *Security Plugins* for OpenSSL. This feature can be enabled through the new **com.rti.serv.secure.authentication.enable_custom_allocators** property.

com.rti.serv.secure.authentication.enable_custom_allocators configures whether to set custom crypto library (e.g., OpenSSL) allocators. When enabled (not by default), the *Security Plugins* will configure custom allocator functions (alloc, realloc, free) to the loaded crypto library with the goal of reducing memory fragmentation at the cost of a minimum performance impact. This is currently only supported in combination with OpenSSL.

This property is only effective the first time a *DomainParticipant* loads the *Security Plugins* within the same process: subsequent *DomainParticipant* creations will ignore this property and leave the existing configuration unchanged. Moreover, this property is only effective if no allocation has been done with the crypto library builtin allocators before the *Security Plugins* have been loaded, otherwise a warning will be logged and no change will be made.

Important: Since the allocator functions live within the *Security Plugins* library, your application must not make any calls to the crypto library once the *Security Plugins* have been unloaded from memory.

5.1.5 Changes Related to Usability

5.1.5.1 "file:" prefix is now optional when specifying filename properties

You may now specify a filename property value without using the prefix "**file:**". If there is no "**data:,**" prefix and the **openssl_engine** property is not set, then the value is assumed to be a filename.

For example, "**file:../../../dds_security/cert/ecdsa01/identities/ecdsa01Peer01Cert.pem**" is now equivalent to "**../../../dds_security/cert/ecdsa01/identities/ecdsa01Peer01Cert.pem**".

5.1.5.2 Updated naming convention for email addresses, common names, and subject names of shipped example certificates

This release changes the naming convention used for the email addresses, common names, and subject names of the shipped example certificates. This change has an impact on the resulting subject name of these certificates and therefore this release also updates the shipped example Permission documents accordingly.

5.1.5.3 New APIs to identify DomainParticipants by subject name

When using the *Security Plugins*, it is natural to identify *DomainParticipants* by their Distinguished Names (subject names). Subject names appear in the Identity Certificate (see Identity Certificates, in Section 3.2 Public Key Infrastructure, in the *RTI Security Plugins User's Manual*) and the Permissions File (see Permissions File, in Section 3.4 in the *RTI Security Plugins User's Manual*).

But many of the current *DomainParticipant* APIs (such as **DomainParticipant::ignore_participant()**) identify *DomainParticipants* by their InstanceHandle_t. In this release, we bridge the gap between InstanceHandle_t and subject names. If you know the subject name of the *DomainParticipant* that you want to ignore, and you need to get the associated InstanceHandle_t, then you can use a new API, **DomainParticipant::get_discovered_participants_from_subject_name()**. You pass it a subject name string, and it outputs an InstanceHandleSeq of *DomainParticipants* that have this subject name.

In addition, if you know the InstanceHandle_t of a *DomainParticipant* for which you want to get the subject name, you can use another new API, **DomainParticipant::get_discovered_participant_subject_name()**. See the Relevant Connext APIs, Chapter 15 in the *RTI Security Plugins User's Manual*.

5.1.5.4 Ability to dynamically load Monitoring Library and Security Plugins on VxWorks systems

Connext has the capability to enable the Monitoring Library and *Security Plugins* using QoS settings, without the need to recompile an application. This release adds support for these features on VxWorks systems.

See Method 1 - Change the Participant QoS to Automatically Load the Dynamic Monitoring Library, Section 50.1.1, in the *RTI Connext Core Libraries User's Manual*
and Dynamic linking in Section 9.1.1, Linking Applications with the Security Plugins, in the *RTI Security Plugins User's Manual*
for details on the QoS properties used to enable these features.

5.1.6 Changes Related to Debuggability

5.1.6.1 Improved message content in case of permissions validation failure

Previously, if validation failed for a permission or governance document, only a high-level message was logged, suggesting that you check the configured permission authorities. This message has been improved. Now it includes a list of the permission authorities in the configuration that failed to sign the document.

5.1.6.2 Messages logged with Security Logging Plugin are now part of SECURITY category

All security events and messages logged with the Security Logging Plugin are now part of the SECURITY logging category (NDDS_CONFIG_LOG_CATEGORY_SECURITY). This has several implications for security-related messages, regardless of whether they come from the *Security Plugins* or *Connext*:

- The Logging Plugin will log a message if its log level is less than or equal to the verbosity of either the *Security Plugins* or the SECURITY category.
- *Connext* will log a security-related message if its log level is less than or equal to the SECURITY category verbosity.
- Setting the verbosity of the *Security Plugins* also configures the verbosity for the SECURITY category, which will affect any security-related message (including those logged from *Connext*) logged from any *DomainParticipant* within the same application.

For more on the interactions between Security Plugins and the SECURITY category verbosities, see Advanced Logging Concepts, in Chapter 7 of the *RTI Security Plugins User's Manual*.

5.1.6.3 Increased logging in case of identity validation failure

Previously, when identity validation failed, the user received only a high-level message informing about the fact and advising to check on configured identity authorities. Now, this message is followed by the list of all authorities listed in the configuration to sign the identity but failing to do so.

5.1.7 Changes Related to the Security Plugins SDK

5.1.7.1 New functions in SDK test infrastructure

There are several new functions you can use for testing:

- **RTITest_waitForEqualsIntExt()**: Wait a certain time for a value to be equal to the expected one. Execute an action every 10ms (which can be useful for updating the value before checking if it matches the one we expect).
- **DDSCTestContext_getMatchingPublicationsLength()**: Get the number of matching publications associated with a *DataReader*.
- **DDSCPubSubDataReaderListenerData_reset()**: Reset the values in the DataReader Listener Data.
- **DDSCTesterHelperLoggerDeviceData_initialize()** and **DDSCTesterHelperLoggerDeviceData_finalize():** Initialize and finalize a semaphore that protects the counter for found messages. The semaphore is required when multiple *DomainParticipants* are concurrently producing the expected log message.
- Functions for positioning a stream:
  - **DDSCTestHelper_positionStreamToBinaryProperty()**
  - **DDSCTestHelper_positionStreamToPid()**
  - **DDSCTestHelper_positionStreamToNextPid()**
- Functions associated with a DDSCPubSubTestContext:
  - **DDSCPubSubTestContext_initializeListener()**
  - **DDSCPubSubTestContext_createPubParticipantWithTypeConfig()**
  - **DDSCPubSubTestContext_createSubParticipantWithTypeConfig()**

5.1.7.2 New '-verbosity' argument for SDK testers

You can now change the verbosity for the access control and cryptography testers using the **-verbosity <int>** argument. It accepts a number between 0 (SILENT) and 6 (STATUS_ALL). The default value is 2: print fatal errors and exceptions. See the output of **-help** for more information about verbosity levels.

5.1.7.3 More meaningful return types for SDK tests

The access control and cryptography testers run a battery of tests. These tests previously returned only two values: RTI_FALSE (0) when a test failed and RTI_TRUE (1) when a test passed. A test can now return an RTITestReturnCode, which allows more possibilities:

- **RTI_TEST_RETCODE_FAILED**: The test failed. This value is equivalent to the previous RTI_FALSE.
- **RTI_TEST_RETCODE_PASSED**: The test passed. This value is equivalent to the previous RTI_TRUE.
- **RTI_TEST_RETCODE_UNSUPPORTED**: The test is not supported. A test won't be supported when it depends on a feature unavailable for the current crypto library or architecture. The testing infrastructure doesn't report unsupported tests as errors. Instead, unsupported tests pass when running.

More return types may be added in the future.

5.1.8 Changes Related to Supported Platforms

5.1.8.1 New Platforms

This release adds support for these platforms:

- macOS 12 on x64 and Arm v8 (SDK only supported on x64)
- Ubuntu 22.04 LTS on x64 and Arm v8 (SDK only supported on x64)
- VxWorks 21.11 on x64 (SDK not supported)
- Windows 11 on x64 with Visual Studio 2022

5.1.8.2 Removed Platforms

The following platforms were supported in *Security Plugins* 6.1.1, but are not supported in release 7.0.0.

- Android
- These Linux platforms:
  - CentOS 6.x
  - NI Linux 3
  - Red Hat Enterprise Linux 6.x
  - Ubuntu 18.04 LTS on Arm v7

- QNX Neutrino 6.x, 7.0.4
- VxWorks 7.x

5.2 What's Fixed in 7.0.0

5.2.1 Fixes Related to Discovery and Authentication

5.2.1.1 Reader incorrectly lost liveliness with writer when using enable_liveliness_protection

A *DataReader* incorrectly reported that a *DataWriter* lost liveliness at the **max_liveliness_loss_detection_period** when using **enable_liveliness_protection**, if the *DataWriter's* (**lease_duration**)/(**assertions_per_lease_duration**) was greater than the **max_liveliness_loss_detection_period**—even if the full **lease_duration** had not passed. This problem has been resolved.

[RTI Issue ID SEC-1630]

5.2.1.2 Key agreement did not use ephemeral key pairs as required by DDS Security specification

The DDS Security 1.1 specification states that dh/ecdh key pairs used for Key Agreement should be used *only once* (i.e., the key should be ephemeral). Previous *Security Plugins* releases were not compliant with this.

As a result, every *DomainParticipant* reused the same Key Agreement public/private key pair for performing Key Establishment with other *DomainParticipants*. Note that recreating the *DomainParticipant* resulted in new keys. The keys were recreated upon *DomainParticipant* recreation, not upon every *DomainParticipant*-to-*DomainParticipant* Key Establishment process.

This non-compliant behavior increased the impact of a hypothetical successful attack where the attacker already took over a *DomainParticipant's* dh/ecdh keys:

- In previous releases, taking over a *DomainParticipant's* temporary dh/ecdh private key (which is reused during the *DomainParticipant's* lifetime) would have resulted in being able to access any communications involving this *DomainParticipant* (with any other *DomainParticipant*).
- Starting with this release (7.0.0), the impact of taking over a *DomainParticipant's* temporary dh/ecdh private key (which is now only used during one Key Establishment process with a specific *DomainParticipant*) is reduced. Now it will result in only being able to access any communications involving the two *DomainParticipants* involved in the authentication (as opposed to all communications from the compromised *DomainParticipant*).

[RTI Issue ID SEC-1676]

5.2.2 Fixes Related to Cryptography

5.2.2.1 Data protection kind did not protect serialized keys sent with dispose samples

If you set **DataWriterQos.protocol.serialize_key_with_dispose** to true, and you set the Governance document tag **data_protection_kind** to a value other than NONE, then the key that was serialized with a dispose sample was, incorrectly, not protected.

To protect this key, you had to set a Governance document tag, **metadata_protection_kind** or **rtps_protection_kind**, to a value other than NONE. This problem has been fixed.

The fix affects backward interoperability with *Security Plugins* 6.1.1 and below. Please see the *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation) for details.

[RTI Issue ID SEC-627]

5.2.3 Fixes Related to Access Control

5.2.3.1 When parsing domain rules from a Permissions document, Security Plugins applied an incorrect order-of-precedence

In 6.1.1, the Security Plugins used an incorrect order-of-precedence when parsing conflicting domain rules from a Permissions document. This problem would have prevented a *Connext* 6.1.1 or higher application from communicating with a *Connext* 7.0.0 (or higher) application.

For example, in the following Permissions Document snippet, a 7.0.0 *DomainParticipant* on domain 12 (let's call this *DomainParticipant* P1) should be allowed to exist:

```
<allow_rule>
    <domains>
        <id>12</id>
    </domains>
</allow_rule>
<deny_rule>
    <domains>
        <id>12</id>
    </domains>
</deny_rule>
```

But when another *DomainParticipant*, P2, discovered P1, P2 incorrectly denied P1 from communicating with it because P2 applied the deny rule instead of the allow rule.

The fix for this issue "future-proofs" compatibility between applications based on *Connext* 7.0.0 and higher. If you have a 6.1.1-based application that needs to communicate with a 7.0.0-based application, you will need a 6.1.1 patch; please contact RTI Support. Further information is in the *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation).

This problem was one scenario within the scope of the problem described in SEC-850, which was described as fixed in *Security Plugins* 6.1.1 but was missing the 7.0.0-related fix described here.

[RTI Issue ID SEC-1687]

5.2.4 Fixes Related to Interoperability with Other Vendors

5.2.4.1 Could not detect participant discovery changes from DomainParticipants using non-RTI Security Plugins

If a *DomainParticipant* using *RTI Security Plugins* was communicating with more than four *DomainParticipants* that were using DDS Security and that were changing their QoS at any time, then there were two problems:

- The following warning would have incorrectly been logged when receiving a ParticipantBuiltinTopicData sample indicating a QoS change from any *DomainParticipant* beyond the fourth one:

  ```
  PRESCstReaderCollator_addInstanceEntry:exceeded ResourceLimitsQosPolicy.max_instances
  ```

- This warning was benign if it was logged upon receiving a ParticipantBuiltinTopicData sample from a *DomainParticipant* that was created using the *RTI Security Plugins*. But if the *DomainParticipant* was not created using a different implementation of DDS Security, then its QoS change would have gone undetected.

This problem, which only affected *Security Plugins* 6.0.0 and above, has been fixed.

[RTI Issue ID SEC-1639]

5.2.4.2 Incorrect key agreement algorithm sent by replier DomainParticipant

Version 1.1 of the DDS Security Specification mandates that the replier *DomainParticipant* must set the key agreement algorithm in the authentication handshake equal to the value received from the initiator *DomainParticipant*.

In previous releases, the replier *DomainParticipant* incorrectly set the value in the handshake to its own key agreement algorithm, when it should have used the initiator's. This did not impact *Connext Secure* or *Connext Micro*, but it might have caused interoperability issues with other vendors. The issue has been fixed.

[RTI Issue ID SEC-1674]

5.2.5 Fixes Related to Debuggability

5.2.5.1 Debug messages not logged when Logging Plugin used Connext Builtin Logging System

If the Security Logging Plugin was configured to use the Connext Builtin Logging System, debug-level messages (DDS_LOGGING_DEBUG_LEVEL) were not logged, even if the **logging.verbosity** property was set to DEBUG.

This was due to a mismatch in the translation between the Logging Plugin and the Connext log levels. This problem has been resolved.

[RTI Issue ID SEC-1640]

5.2.5.2 Verbosity was not per application when Logging Plugin used Connext Builtin Logging System

For messages logged through the Connext Builtin Logging System (either directly or by the Logging Plugin), the verbosity is supposed to be per application, meaning that, if a *DomainParticipant* has configured the verbosity, it will update it for all *DomainParticipants* within the application. This did not always happen, however, when the messages came from the Logging Plugin.

When messages came from the Logging Plugin, they were filtered out twice: at the Logging Plugin level (using the verbosity that was configured for the *DomainParticipant* upon creation) and at the Connext level (using the verbosity specified by the last *DomainParticipant* created in the application). As a result, the threshold used for determining if a message should be logged or not was the lower of the two verbosity levels.

Because of this, if a second *DomainParticipant* specified a greater verbosity level than the first one, the verbosity of the first one was not changed, because messages were being discarded anyway at the Logging Plugin level.

Now, the Security verbosity is always per application, regardless of whether the messages come from the Logging Plugin, and regardless of whether the Logging Plugin is configured to use the Connext Builtin Logging System. The last *DomainParticipant* to configure the Security verbosity will apply that setting to all the *DomainParticipants* within the application.

[RTI Issue ID SEC-1648]

5.2.5.3 Validation of boolean properties did not treat non-boolean values as errors

In previous releases, specifying a non-boolean value was treated as not specifying any value at all, and *Connext* silently used the default value. This problem has been resolved. Now, specifying a non-boolean value for a boolean property will result in an error containing "is not a boolean value", followed by entity creation failure.

[RTI Issue ID SEC-1653]

5.2.5.4 Obscure error messages when failing to verify Identity Certificate in debug libraries of Security Plugins for wolfSSL

The *Security Plugins* for wolfSSL logged a message similar to the following if verification of the Identity Certificate failed:

```
RTI_Security_CryptoLibAdapterWolfSSL_logging_cb:!wolfSSL error occurred, error = 162
line:40816 file:wolfssl-4.7.0-commercial/src
```

The right message was also logged:

```
RTI_Security_Authentication_getCertificate:{"DDS:Security:LogTopic":{"f":"10","s":"3","t":
{"s":"1656525802","n":"388092999"},"h":"bld-ubuntu1804","i":"0.0.0.0","a":"RTI Secure DDS
Application","p":"12300","k":"security","x":[{"DDS":[{"domain_id":"12"},
{"guid":"9d69955f.b83e6145.974e667f.1c1"},{"plugin_class":"Authentication"},{"plugin_
method":"RTI_Security_Authentication_getCertificate"}]}],"m":"Identity verification failed.
Make sure it was signed by the right authority."}}
```

The wolfSSL error message made the error from the *Security Plugins* less noticeable. This issue, which only affected the debug libraries, has been fixed.

[RTI Issue ID SEC-1710]

5.2.6 Fixes Related to the Security Plugins SDK

5.2.6.1 Certificate Revocation Lists expired after 30 days

In previous releases of the Security Plugins SDK, a subset of the tests started failing after 30 days because Certificates Revocation Lists expired. The 30 days started counting from the moment RTI generated the CRLs. Therefore, users may have found that some SDK tests never passed. This issue has been fixed. The CRLs now have the same expiration time as the certificates: 5 years.

[RTI Issue ID SEC-1677]

5.2.7 Fixes Related to Shipped Examples

5.2.7.1 Secure Hello World example always linked OpenSSL dynamically

The C and traditional C++ **hello_security** examples always linked OpenSSL dynamically, even if the user wanted to use static linking. This issue has been fixed. Now, when linking on a Windows system with Visual Studio, the OpenSSL and crypt32 libraries are linked statically, unless you choose Debug DLL or Release DLL from the configuration pull-down menu of the provided projects. Or, when using a makefile, OpenSSL is now linked statically, unless you use pass the **SHAREDLIB=1** argument to the **make** command.

[RTI Issue ID SEC-880]

Chapter 6 Known Issues

Note: For an updated list of critical known issues, see the Critical Issues List on the RTI Customer Portal at https://support.rti.com.

6.1 No Support for ECDSA-ECDH with Static OpenSSL Libraries and Certicom Security Builder

If you are using the Certicom® Security Builder® engine, you cannot use the ecdsa-ecdh shared secret algorithm together with static OpenSSL libraries. If you want to use ecdsa-ecdh with Certicom Security Builder, you must use dynamic OpenSSL libraries. Attempting to use ecdsa-ecdh with static OpenSSL libraries and Certicom Security Builder will cause the following errors during participant discovery:

```
Authentication_compute_sharedsecret:failed to provide remote DP public key
Authentication_process_handshake:key generation fail
Authentication_get_shared_secret:empty secret
PRESParticipant_authorizeRemoteParticipant:!security function get_shared_secret
```

6.2 No Support for Writing >65kB Unfragmented Samples Using Metadata or RTPS Message Protection

The following use case is not supported:

- **metadata_protection_kind** = SIGN or ENCRYPT or **rtps_protection_kind** = SIGN or ENCRYPT
- **message_size_max** > 65536. This is possible when using the TCP transport.
- The user is writing unfragmented samples of size greater than 65kB but less than **message_size_max**.

In order to write the large sample, you must set **message_size_max** to be smaller than the message size, so the sample can be put in fragments smaller than 65 kB.

[RTI Issue ID SEC-768]

6.3 subscription_data and publication_data in check_local_datawriter_match / check_local_datareader_match are not Populated

When calling **check_local_datawriter_match** / **check_local_datareader_match**, *Connext* does not set the **subscription_data** and **publication_data** parameters. While this issue has no impact on the DDS Security builtin plugins, it could affect a custom plugin relying on those parameters.

[RTI Issue ID SEC-758]

6.4 relay_only parameter in check_remote_datareader is not Populated

When calling **check_remote_datareader**, *Connext* does not set the relay_only parameter. While this issue has no impact on the DDS Security builtin plugins, it could affect a custom plugin relying on this parameter.

[RTI Issue ID SEC-852]

6.5 'Allow Rule' Patterns Incorrectly do not Allow Subset Patterns in QoS

In the Permissions Document, an <allow_rule> that has a pattern partition other than * (e.g., P*) incorrectly does not allow creation of an entity whose PartitionQosPolicy contains a regular expression pattern that is a subset of that <allow_rule> (e.g., P1*). This problem only affects *Security Plugins* 6.1.0 and above.

The workaround is to change the <allow_rule>'s pattern partition to exactly match the pattern partition in the QoS (e.g., change P* to P1*).

[RTI Issue ID SEC-1242]

6.6 FlatData in Combination with Payload Encryption and/or Compression will not Save Copies

RTI FlatData™ language binding offers a reduced number of end-to-end copies when sending a sample (from four to two), providing improved latency for large data samples. (See the "FlatData Language Binding" section in the *RTI Connext Core Libraries User's Manual*.) When used with payload encryption and/or payload compression, however, there are no savings in the number of copies. (See the section "Interactions with *RTI Security Plugins* and Compression" in the "Using FlatData Language Binding" section of the *RTI Connext Core Libraries User's Manual*). In future releases, other copies currently being made can potentially be optimized out in order to reduce the number of copies when using FlatData in combination with security and compression.

[RTI Issue ID CORE-11262]

6.7 Warning that Crypto Library Adapter File has no Symbols when Statically Building Security Plugins SDK on macOS Systems

Statically building the *Security Plugins SDK* on macOS systems will result in warning messages similar to:

```
file: libnddssecurityz.a (CryptoLibAdapterWolfSSL.c.o) has no symbols
file: libnddssecurityz.a (CryptoLibAdapterWolfSSL47.c.o) has no symbols
```

These warnings occur because the SDK is linking against all the crypto library adapter files. These warnings are not harmful, since the SDK does not use the empty files mentioned in the warnings. This problem will be resolved in a future release.

[RTI Issue ID SEC-1984]