RTI Security Plugins

Release Notes

Version 6.1.0

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 6.1.0 is supported on the following platforms.

Note: POSIX®-compliant architectures that end with "FACE_GP" are not supported.

Table 1.1 Supported Platforms

Operating System	Version
Android™ *Available on demand*	All platforms listed in the *RTI Connext® DDS Core Libraries Release Notes* for the same version number, except SUSE® Linux Enterprise Server.
Linux®	All platforms listed in the *RTI Connext DDS Core Libraries Release Notes* for the same version number, except SUSE® Linux Enterprise Server.
macOS®	
QNX®	All QNX Neutrino® 6.5 and higher platforms listed in the *RTI Connext DDS Core Libraries Release Notes* for the same version number.
VxWorks	VxWorks 7.0 SR 0630 (Dynamic loading of *Security Plugins* not supported in kernel mode. Tested with OpenSSL from VxWorks 7.)
Windows®	All platforms listed in the *RTI Connext DDS Core Libraries Release Notes* for the same version number.

See the *RTI Connext DDS Core Libraries Platform Notes* for more information.

The *Security Plugins* are also supported on the platforms in Table 1.2 Custom Supported Platforms; these are target platforms for which RTI offers custom support. If you are interested in these platforms, please contact your local RTI representative or email sales@rti.com.

Table 1.2 Custom Supported Platforms

Operating System	Version	CPU	RTI Architecture Abbreviation
Linux	RedHawk™ Linux 6.5	x86	i86RedHawk6.5gcc4.9.2
		x64	x64RedHawk6.5gcc4.9.2
	Wind River® Linux 8	Arm v7	armv7aWRLinux8gcc5.2.0
	Yocto Project® 2.5	Arm v8	armv8Linux4gcc7.3.0
QNX	QNX 6.6	x86	i86QNX6.6qcc_cpp4.7.3
		Arm v7	armv7aQNX6.6.0qcc_cpp4.7.3
	QNX 7.0.4	Arm v7	armv7QNX7.0.0qcc_cxx5.4.0

Chapter 2 Compatibility

Security Plugins 6.1.0 is interoperable with 5.2.7 and higher versions of *Security Plugins*.

This release of *Security Plugins* includes partial support for the DDS Security specification from the Object Management Group (OMG)[1].

Security Plugins 6.1.0 is API-compatible with OpenSSL® versions 1.1.0a through 1.1.1k. It is not API-compatible with versions earlier than OpenSSL® 1.1.0a. Note that *Security Plugins* 6.1.0 has only been tested by RTI using OpenSSL 1.1.1k. If you need Security Plugins 6.1.0 to run against older versions of OpenSSL®, please contact support@rti.com.

Persistence Service databases secured with Security Plugins 6.1.0 are incompatible with databases generated by older versions of *Persistence Service*.

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

[1]http://www.omg.org/spec/DDS-SECURITY/1.1/

Chapter 3 What's New in 6.1.0

Release 6.1.0 is a General Access Release based on release 6.0.1. This section describes what's new.

3.1 New Platforms

This release adds support for these platforms:

- macOS 10.15 (x64)
- QNX Neutrino 7.0.4 (x64 and Arm v8)
- QNX Neutrino 7.0.4 (Arm v7) (Custom-supported target platform)
- Red Hat® Enterprise Linux 7.6 (x64)
- Ubuntu® 18.04 LTS (Arm v7 and v8)
- Ubuntu 20.04 LTS (x64)
- VxWorks 7.0.0 SR0630 (x64) (Dynamic loading of *Security Plugins* is not supported in kernel mode. Tested with OpenSSL from VxWorks 7.)
- Yocto Project® 2.5 (Arm v8) (Custom-supported target platform)

3.2 Removed Platforms

The following platforms are no longer supported:

- Android 5.0, 5.1
- iOS®
- macOS 10.12
- Ubuntu 12.04 LTS
- Wind River Linux 7

3.3 Improvements to Security Plugins Documentation

This release features a brand new *Security Plugins User's Manual*, which improves multiple aspects with respect to the previous manual:

- Easier to follow: it contains detailed information and is shared in a way that is accessible for *Connext DDS* users with some notions on security.
- Self-contained: it no longer depends on the OMG DDS Security specification.
- Includes information for advanced users: there are two full chapters on Design Considerations and Best Practices.

Additionally, the new *Security Plugins Getting Started Guide*, which was already available on the RTI Community portal, is now integrated with the *Security Plugins* installation. Also, new ECDSA examples have been added along with the existing RSA examples.

3.4 Changes Related to OpenSSL

3.4.1 Updated OpenSSL version

This release uses OpenSSL 1.1.1k (instead of 1.1.1d).

3.4.2 Target OpenSSL bundles distributed as .rtipkg files

Target OpenSSL bundles are now distributed as **.rtipkg** files. Once installed, the OpenSSL files are available in **<installation_folder>/third_party**.

3.4.3 Changes to OpenSSL static library names

The OpenSSL static library names no longer have a "z" suffix. **libcryptoz** has been renamed to **libcrypto**, and **libsslz** has been renamed to **libssl**. When including the static libraries in a makefile, we recommend including the whole path to the OpenSSL static libraries in order to avoid confusion with the dynamic libraries. Here is an example:

```
gcc -o myApp myApp.o -L$NDDSHOME/lib/$ARCH -lnddssecurityz -lnddscz -lnddscorez $RTI_
OPENSSLHOME/$ARCH/release/lib/libcrypto.a
```

In addition, the Android static library **librtisslsupportz** has been removed. You may use **libcrypto** instead.

3.5 Changes Related to New Supported Products

3.5.1 Added support for RTI Real-Time WAN Transport

This release adds support in the *Security Plugins* for the new *RTI Real-Time WAN Transport*. In particular, this release supports protecting the Real-Time WAN Transport Binding Ping messages.

When using HMAC-Only Mode, the value of **hmac_only.cryptography.key** is now used to protect Real-Time WAN Transport Binding Ping messages. When not using HMAC-Only Mode, you may now set a new optional property, **cryptography.rtps_protection_key**, to specify a pre-shared key that is used to protect Real-Time WAN Transport Binding Ping messages.

3.5.2 Added limited support for RTI Cloud Discovery Service

This release supports using *RTI Cloud Discovery Service* in combination with the *Security Plugins*. In particular, this release supports protecting the participant announcement messages emitted by *Cloud Discovery Service* against tampering and replay.

To protect *Cloud Discovery Service* messages against tampering and replay, you can protect the exchange of participant announcements between *Cloud Discovery Service* and the *DomainParticipants* through a pre-shared key (the Participant Discovery Protection Key) that must be set in all *DomainParticipants* and *Cloud Discovery Service*.

This release adds a new property, **authentication.participant_discovery_protection_key**, to configure this behavior. Please refer to the section on 'Security Considerations when Using Cloud Discovery Service' in the *RTI Security Plugins User's Manual* for more information.

3.6 Changes Related to Observability

3.6.1 Added local contextual information to security events logged through Connext DDS builtin logging system

This release changes the format of security events that are logged through the *Connext DDS* builtin logging system (NDDS_Config_Logger) to include all the metadata that was already included when the log messages were distributed over DDS (by setting the **logging.mode_mask** property to include the SECURITY_TOPIC flag).

Specifically, security events logged through the *Connext DDS* builtin logging system now use a JSON format, and they look similar to the following:

```
{"DDS:Security:LogTopic":{"f":"10","s":"6","t":
{"s":"1589283680","n":"554941999"},"h":"vmhyrule-u1804-x64","i":"0.0.0.0","a":"RTI Secure DDS
Application","p":"22689","k":"security","m":"successfully created encryption key","x":[{"DDS":
[{"domain_id":"<unknown>"},{"guid":"<unknown>"},{"plugin_class":"Cryptography"},{"plugin_
method":"RTI_Security_Cryptography_register_matched_remote_participant"}]}]}}

{"DDS:Security:LogTopic":{"f":"10","s":"6","t":
{"s":"1589283684","n":"266673000"},"h":"vmhyrule-u1804-
x64","i":"0.0.0.0","a":"MyAppTestName","p":"22689","k":"security","m":"successfully registered
endpoint","x":[{"DDS":[{"domain_id":"77"},{"guid":"b3339b76.d20fbe2d.6206b40f.1c1"},{"plugin_
class":"Cryptography"},{"plugin_method":"RTI_Security_Cryptography_register_matched_remote_
endpoint"}]}]}}
```

The mapping from the above format to the fields in the **DDS:Security:LogTopic** type is as follows:

- "f": facility
- "s": severity
- "t": timestamp (within this field, "s" refers to seconds, and "n" to nanoseconds)
- "h": hostname
- "i": hostip
- "a": appname
- "p": procid
- "k": msgid
- "m": message
- "x": structured_data

3.6.2 Improved builtin logging verbosity of security messages with informational severity

Security-related log messages with severity DDS_LOGGING_INFORMATIONAL_LEVEL were logged with the NDDS_CONFIG_LOG_LEVEL_STATUS_REMOTE log level when going through the Connext DDS builtin logging system. These log messages could have gone unnoticed, depending on the verbosity configured. Now, these messages are logged with NDDS_CONFIG_LOG_VERBOSITY_STATUS_LOCAL log level.

3.6.3 Identity certificate and identity CA certificate values now included in log messages

Previously, if verification of the identity certificate failed, the resulting error log message did not include the certificate values. Observability has been improved by logging the identity certificate (**dds.sec.auth.identity_certificate**) and the identity CA certificate (**dds.sec.auth.identity_ca**).

This can be helpful information if the certificates being read are not the ones the user expects (for example, if a different security profile is loaded).

3.6.4 Issuer and subject of invalid certificate now included in log messages

Previously, if verification of an identity certificate failed, the resulting log message did not include the certificate subject or issuer and went through the NDDS_Config_Logger. Observability has been improved by logging both the issuer and subject of invalid certificates, for both locally created and remotely discovered participants. This message now goes through the Security Logging Plugin. This information can be used to audit attempts to join a secure domain.

3.6.5 Enhanced message logged when trying to load a non-existent CA identity file

This feature was added in release 6.0.1, but did not get documented at that time.

If the **dds.sec.auth.identity_ca** property pointed to a non-existent file, *Security Plugins* logged an error message including the following text:

```
Error opening CA certificate
failed to load certificate authority cert in file
```

That text was misleading, since it implied there could be a problem processing the file (when the problem was in fact that the file was not found). This release replaces the above text with a more meaningful error:

```
No such file or directory
failed to lookup dds.sec.auth.identity_ca
```

(This improvement was made while in the course of making **dds.data_writer.history.key_material_key** mandatory; see 'Property key_material_key now required for Secure Persistence Service' in 'What's New' in the 6.0.1 *Security Plugins Release Notes*.)

3.6.6 Message now logged when authentication and authorization complete

The following (or a similar) log message is now generated when a *DomainParticipant* authenticates and authorizes another *DomainParticipant* after completing an authentication handshake:

```
PRESParticipant_processHandshake:[Local Participant: 12345678 12345678 12345678] [Remote
Participant: 87654321 87654321 87654321] security: authentication and authorization completed
```

The specific numbers will vary depending on the D*omainParticipants*' GUIDs.

3.6.7 Replaced cryptic error message in recoverable authentication scenario

During authentication, a *DomainParticipant* may have rarely generated this message at ERROR verbosity:

```
RTI_Security_Authentication_process_initial_handshake_message:failed to get identity
certificate binary property
```

This message indicated that the *DomainParticipant* was in a recoverable authentication scenario. Instead of seeing this ERROR message, you will now see this message at NOTICE verbosity:

```
RTI_Security_Authentication_begin_handshake_reply:received unexpected handshake message,
probably from a participant that this one lost liveliness with before ongoing authentication
completed. Once this participant sends an authentication request, communication should be
restored.
```

3.6.8 Error is now logged if OpenSSL is finalized prematurely

In previous releases, if a DDS application prematurely finalized the OpenSSL library or certain OpenSSL resources (for example, by calling the function **EVP_cleanup**) during *Security Plugins* execution, this triggered a crash upon discovery of a remote secure *DataWriter* or *DataReader*.

While finalizing the OpenSSL library after initialization is not supported (and may lead to undefined behavior), the *Security Plugins* will now handle this particular situation gracefully, and the application will fail with the following error message instead of crashing:

```
!get cipher: this may be caused by a premature openSSL library finalization
```

3.7 Changes Related to Scalability

3.7.1 Endpoint state transition improvements for key distribution

This release incorporates a set of internal improvements to the way the *Security Plugins* handle key distribution interactions with endpoint state transitions.

In particular, the *Security Plugins* now treat secure endpoints whose **key material has not been fully exchanged** (i.e., both the *DataWriter* and the *DataReader* have successfully delivered their keys to their match) as **unmatched endpoints**. Consequently, *Connext DDS* will not start sending secure endpoint traffic until it has confirmed the remote endpoint can successfully decode it, which will save unnecessary traffic during initial endpoint discovery. Another result of this change is that *Connext DDS* will not start applying liveliness/activity timeouts until the two involved endpoints are ready to successfully exchange RTPS messages, which will save the application from unexpected liveliness/activity events during/immediately after initial endpoint discovery.

Note that beyond saving traffic and avoiding unexpected transitions, the behavior observable by *Connext DDS* applications has not changed.

3.7.2 Automatic removal of human-readable part of propagated Identity Certificates

The human-readable part of Identity Certificates is now stripped by default. Identity Certificates are in PEM format, which, in addition to the Base64 encoded certificate, may contain a human-readable component. If present, the human-readable part is now removed before transmitting the Identity Certificate over the network. You can control this behavior with the boolean property **authentication.propagate_simplified_identity_certificate**. See the *Security Plugins User's Manual* for more details.

3.7.3 Disabled multicast on Authentication and Key Exchange builtin endpoints

This release disables multicast for the builtin endpoints of Authentication (ParticipantStatelessMessage) and Key Exchange (ParticipantVolatileMessageSecure). The *DataWriters* of these topics send messages directly to individual *DataReaders*. Multicast is therefore unnecessary. In previous releases, the *DataReaders* of these topics inherited the multicast locators that were specified in the DiscoveryQosPolicy's **multicast_receive_addresses** field. In this release, this inheritance does not occur; the *DataReaders* of these topics never use any multicast locators.

3.7.4 Improved discovery scalability when using BuiltinQosLib::Generic.Security profile

In previous releases, the **BuiltinQosLib::Generic.Security** profile set the **fast_heartbeat_period** and **late_joiner_heartbeat_period** to 100ms. This may have resulted in excessive traffic during discovery, which may have affected scalability if your system had a considerable number of *DomainParticipants*.

In this release, both the **fast_heartbeat_period** and **late_joiner_heartbeat_period** settings have been changed to 1s. As a result, using **BuiltinQosLib::Generic.Security** as your base profile will provide shorter discovery times without compromising your system's scalability.

3.7.5 Changed default value of max_heartbeat_retries for secure volatile channel to UNLIMITED

A *DataReader* is marked as inactive when it does not respond within the **max_heartbeat_retries** number of periodic heartbeats. This means that the *DataWriter* will not wait for the *DataReader* to send an ACK/NACK before removing DDS samples.

The default value for **max_heartbeat_retries** was changed to UNLIMITED for the secure volatile channel because, if a pending volatile sample is removed and never resent, the system enters into an unrecoverable situation (unless participant liveliness expires).

For more information, see 'Configuring Reliability Protocol Settings of the Key Exchange Topic' in the *Security Plugins User's Manual* and the RELIABILITY QosPolicy in the *Connext DDS Core Libraries User's Manual.*

3.7.6 Changed default fast_heartbeat_period for secure volatile channel

The default value of the **fast_heartbeat_period** for the secure volatile channel has been changed to 0.25 seconds. Previously, it was 8 milliseconds, which was more aggressive. The new value should result in less traffic and a better default experience.

For more information, see 'Configuring Reliability Protocol Settings of the Key Exchange Topic' in the *Security Plugins User's Manual* and the RELIABILITY QosPolicy in the *Connext DDS Core Libraries User's Manual*.

3.8 Changes Related to Shipped Examples

3.8.1 New Shapes Demo XML examples

This release includes new XML files to configure security:

- **RTI_SHAPES_DEMO_GOVERNANCE_RTPS_ENCRYPT_WITH_ORIGIN_ AUTHENTICATION.xml**
- **signed/RTI_SHAPES_DEMO_GOVERNANCE_RTPS_ENCRYPT_WITH_ORIGIN_ AUTHENTICATION.p7s**

And a new profile:

- **Security::SecureRtpsEncryptWithOriginAuthentication**
  This profile provides maximum security for RTPS messages. It protects outgoing messages from being tainted or viewed, and protects outgoing messages from being replayed by a subscriber masquerading as a publisher.

In addition, the *Shapes Demo User's Manual* includes a new example that illustrates the contents of RTPS packets when using maximum protection for RTPS messages.

3.8.2 Changes in shipped example certificates and OpensSSL configuration files

This release changes the way the shipped example certificates and OpensSSL configuration files are structured. In particular, the following structure applies:

- cert
    - <pkiName_description>
        - ca
            - private
            - database
            - newCerts
        - identities

3.8.3 Removed libssl library from hello_security makefiles

The *Security Plugins* do not depend on the OpenSSL library **libssl**. They only depend on **libcrypto**. Therefore, **libssl** has been removed from the **hello_security** example makefiles and project files.

3.9 Changes Related to Usability

3.9.1 Ability to configure reliability protocol settings of the Key Exchange builtin topic

This release adds the ability to configure the reliability protocol settings of the Key Exchange topic (ParticipantVolatileMessageSecure), when security is enabled. Now you can modify the reliability protocol and data lifecycle settings of the Key Exchange builtin topic by changing the DiscoveryConfigQosPolicy's **secure_volatile_reader** and **secure_volatile_writer**.

For more information, see 'Configuring reliability protocol settings of the Key Exchange Topic' in the *Security Plugins User’s Manual* and the RELIABILITY QosPolicy in the *Connext DDS Core Libraries User's Manual*.

3.9.2 Introduced bitmask to configure the logging methods in use

This release changes the way to configure which logging methods to use, by using a bitmask. The property **logging.mode_mask** now configures whether to use the Connext DDS builtin logging system, the Builtin Secure Logging Topic as defined in the DDS Security specification, or both.

The **logging.mode_mask** property is now the only way to enable a logging method, deprecating the **logging.distribute_enable** property.

Distributing the security log over DDS now requires setting the **logging.mode_mask** to include **the SECURITY_TOPIC** flag. For consistency, properties for configuring security logging distributed over DDS have been renamed to start with **logging.security_topic**. For more information, see 3.9.3 New property names to configure security logging distribution over DDS on the next page.

Redirecting the security messages to a file with the **logging.log_file** property is no longer possible, and using this property will result in a *DomainParticipant* creation failure. You can still redirect the security log to a file by enabling the BUILTIN flag in the **logging.mode_mask** property (enabled by default) and configuring the Connext DDS Builtin Logging System to use a log file or an output device.

3.9.3 New property names to configure security logging distribution over DDS

In previous releases, properties for configuring security logging distributed over DDS had a name starting with **logging.distribute**. This release renames these properties to start with **logging.security_topic**.

Properties with the former names are deprecated and will be removed in a future release. If you set a property using both the **logging.distribute** and the **logging.security_topic** forms, the latter will take effect, and the former will be ignored.

3.9.4 Changes related to logging plugin configuration

The Logging Plugin properties **logging.distribute.writer_history_depth** and **logging.distribute.writer_timeout** have been deprecated. Setting either of them will not affect the logging *DataWriter's* QoS and will result in a WARNING-level log message similar to:

```
Ignoring logging.distribute.writer_history_depth, which is now a deprecated property.
Use the logging.security_topic.profileproperty to specify the writer QoS.
```

For the **logging.security_topic.profile** property, the documented behavior did not match the actual behavior. For example, the Logging Plugin actually hard-coded the durability.kind, **history.kind**, **publish_mode.kind**, and **reliability.kind**, regardless of what was in the profile. This problem has been fixed. These values are no longer hard-coded. See the *Security Plugins User's Manual* for details.

3.9.5 Improved handling of messages that exceed logging queue message_size_max

If the **LogTopic***DataWriter* failed to write a log message because of the **logging.security_topic.queue.message_size_max limit**, the error message was:

```
REDAConcurrentQueue_startWriteEA:!precondition: msgSize > q->_desc._messageSizeMax
```

This behavior has improved. If it's possible to send any part of the message, the message will be truncated to fit within the limit, and *Connext DDS's* own builtin logging system will generate a WARNING message. If it's not possible, the write will fail, and *Connext DDS's* own builtin logging system will generate an ERROR message. To guarantee space for the shortest possible log message, the minimum allowed value of **logging.security_topic.queue.message_size_max** is now 27.

3.9.6 The "file:" prefix can now be used for alternative files

In release 6.0.0, the properties **com.rti.serv.secure.authentication.alternative_ca_files** and **com.rti.serv.secure.access_control.alternative_permissions_authority_files** did not accept a "file:" prefix in their values, unlike all other properties that accepted a file name as a value. (Release 6.0.0 introduced the "file:"

prefix to the other properties.) These properties now accept an optional "file:" prefix in front of any of the filenames in the list.

3.9.7 Added support to provide a Certificate Revocation List as a string

You may now specify the Certificate Revocation List as document contents instead of a file name. The **authentication.crl_file** property has been deprecated and replaced by **authentication.crl**, which requires a "**file:**" or "**data:,**" prefix.

3.9.8 Provided Certificate Revocation List may include CRLs from intermediate CAs

The **authentication.crl** property value may now contain CRLs signed by intermediate CAs from an identity certificate chain.

Consider this scenario:

- rootCa signed
    - intermediateCa
    - identityCert1
    - rootCrl, which revoked
        - identityCert1
- intermediateCa signed
    - identityCert2
    - intermediateCrl, which revoked
        - identityCert2

Consider the following configuration:

- identity_ca = rootCa
- crl = rootCrl concatenated with intermediateCrl
- identity_certificate = identityCert2 concatenated with intermediateCa

In previous releases, certificate verification would have succeeded. Now, certificate verification fails because intermediateCa revoked identityCert2.

3.9.9 Disallow multiple private keys

If the value of **dds.sec.auth.private_key** contained multiple private keys concatenated to each other, *DomainParticipant* creation succeeded and the *DomainParticipant* used the first private key that appeared in the value.

This behavior was error-prone and has been improved. *DomainParticipant* creation now fails in this situation, with this log message:

```
there must be exactly one private key in this URI
```

3.9.10 Ability to load multiple OpenSSL engines

You may now use the **openssl_engine** property to load multiple engines. You may specify a semicolon-separated list of dynamic libraries that each implement an OpenSSL engine. Each engine may implement a different set of security functions. For example, one engine may implement certificate management, while another engine may implement cryptographic operations. If the **authentication.keyform** property value is **engine**, the private key must be successfully loaded by exactly one of the engines in this list.

3.9.11 New property to disable RSA PSS padding

The kind of padding used when signing and verifying documents can be now controlled using the property **<prefix>.authentication.rsa_pss_pad**. The value is a boolean:

- TRUE [default]: Use RSA PSS padding (RSA_PKCS1_PSS_PADDING) as specified in the DDS Security specification.
- FALSE: Use standard RSA padding (RSA_PKCS1_PADDING).

This property takes effect only on certificate authorities that use RSA. All of the *DomainParticipants* in the system must set this property to the same value in order to communicate with each other.

This allows you to use the shipped OpenSSL configuration files to regenerate the certificates without needing to create new directories (database files still need to be initialized).

Chapter 4 What's Fixed in 6.1.0

4.1 Fixes Related to Access Control

4.1.1 Permissions document incorrectly required subject names to have attributes in a certain order

If the order of the <subject_name> attributes in the signed permissions file (i.e., the order of the C, L, CN, and ST elements) was not in forward or reverse order relative to the Subject field in the identity_certificate, then the <subject_name> would not be considered a match with the identity_ certificate, and you would see the following error, followed by a *DomainParticipant* creation failure:

```
[CREATE Participant]RTI_Security_AccessControl_get_grant_from_certificate:XML file
doesn't contain a grant for subject name
```

The requirement of a certain order does not align with X.509 certificate standards. This problem has been fixed by allowing the attributes to appear in any order.

[RTI Issue ID SEC-1000]

4.1.2 Wrong permissions validity date if date is a leap year

According to the DDS Security specification, the Permissions Document contains a <validity> element, which contains <not_before> and <not_after> elements. Each of the latter two elements contains a date and time. If you specified a leap year as the date, the *Security Plugins* incorrectly added one day to the date. For example, Security Plugins incorrectly interpreted "2020-01-08T00:00:00" as "2020-01-09T00:00:00". As a result, if you set the <not_before> value to less than a day before the current time, and the day was within a leap year, you would incorrectly get this error and fail DomainParticipant creation:

```
RTI_Security_PermissionsGrant_isValidTime:now is before not_before of permissions file
```

This problem has been fixed. Leap years in the Permissions Document are now interpreted correctly.

[RTI Issue ID SEC-1056]

4.1.3 Incorrect rule matching when there were multiple rules for the same topic

In a Permissions Document, if an allow or deny rule contained multiple publish or subscribe rules for the same topic, and the first publish or subscribe rule was not applicable because of partitions or data tags but the second rule was applicable, then the second rule was incorrectly not applied.

Here is an example scenario:

```
<deny_rule>
    <subscribe>
        <topics>
            <topic>SEC1241Topic</topic>
        </topics>
        <data_tags>
            <tag>
                <name>tag1</name>
                <value>value1</value>
            </tag>
        </data_tags>
    </subscribe>
    <subscribe>
        <topics>
            <topic>SEC1241Topic</topic>
        </topics>
        <data_tags>
            <tag>
                <name>tag2</name>
                <value>value2</value>
            </tag>
        </data_tags>
    </subscribe>
</deny_rule>
<default>ALLOW</default>
```

If you tried to create a *DataReader* of SEC1241Topic with a data tag of {tag2, value2}, the *DataReader* would incorrectly be created because of <default>ALLOW</default>. This problem has been fixed. This *DataReader* will now fail to be created because of the second subscribe rule.

[RTI Issue ID SEC-1241]

4.1.4 Incorrect 'allow rule' matching for partitions with regular expression patterns

In the Permissions Document, an <allow_rule> with a pattern partition (e.g., "P[!1]") incorrectly allowed creation of an entity whose PartitionQosPolicy contained a regular expression pattern that was not a subset of that <allow_rule> (e.g., "P*". "P1" is included in "P*" but not included in "P[!1]", so "P*" should not be allowed).

This problem has been fixed. An entity with that partition pattern will no longer be allowed in this scenario. Now, the entity's partitions must consist of only "P[!1]" or concrete partitions included by "P[!1]". In order for an <allow_rule> to allow an entity, any pattern partitions in the entity must have a corresponding exact match in the <allow_rule>.

Addressing this issue requires introducing a behavior (see 5.6 'Allow Rule' Patterns Incorrectly do not Allow Subset Patterns in QoS on page 30) that has been determined to cause less user friction. Previously, if "P*" was allowed, then "P1*" was also allowed. Now, "P1*" is incorrectly no longer allowed because "P1*" is not an exact match with "P*".

SEC-1242 is also described in the *Migration Guide on the RTI Community Portal (https://-community.rti.com/documentation)*.

[RTI Issue ID SEC-1228]

4.1.5 'Allow rule' with no partitions incorrectly allowed an entity with an empty partition followed by a non-empty partition

An <allow_rule> with no partitions incorrectly allowed creation of an entity whose PartitionQosPolicy contained the empty partition followed by a non-empty partition (e.g., "" followed by "A"). This problem has been fixed. This entity is no longer allowed because its PartitionQosPolicy contains "A", which is not empty.

[RTI Issue ID SEC-1245]

4.1.6 Incorrect 'deny rule' matching for partitions with regular expression patterns

In the Permissions Document, a <deny_rule> that had concrete non-empty partitions (e.g., "partitionA") incorrectly did not prevent creation of an entity whose PartitionQosPolicy contained a regular expression pattern that intersected with that <deny_rule> (e.g., "partition*"). This problem has been fixed. A <deny_rule> with "partitionA" will now deny a PartitionQosPolicy with "partition*" in order to prevent the entity from reading or writing data on partitionA. Note that "partition[!A]" would be allowed (since !A means "any character except A").

[RTI Issue ID SEC-1248]

4.2 Fixes Related to Cryptography

4.2.1 Unexpected 'DecryptFinal' or precondition failure during participant key exchange may have prevented communication

There were certain scenarios that may have led to issues during *DomainParticipants*' key exchange when enabling security in your *Connext DDS* application:

- With release libraries, you would have seen "DecryptFinal failed. Possible GCM authentication failure" when **logging.verbosity** (or **logging.log_level**) was set to DEBUG. (Note this error can happen for other, expected reasons, too.)
- With debug libraries, the behavior depended on which release you were running:
  - With *Connext DDS* 5.3.0.20 and 6.0.1.3 (which included SEC-1061) you would have seen the following precondition failure at the function **RTI_Security_Cryptography_encode_ submessage()**: "!precondition: (remote_endpoint_crypto_list)->_size != 1". This precondition should never occur, and when it triggers it may prevent any further communications with certain *DomainParticipants*.
  - With all other releases, you would have seen "DecryptFinal failed. Possible GCM authentication failure" when **logging.verbosity** (or **logging.log_level**) was set to DEBUG. (Note this error can happen for other, expected reasons, too.)

In particular, one of the scenarios leading to these issues was triggered if you created and destroyed a *DomainParticipant*, A1, and then created a new *DomainParticipant*, A2, on the same machine and with the same configuration as A1. In this scenario, a secure *DomainParticipant*, B, that remained alive during the creation of A1 and A2 may have produced the aforementioned errors. Additionally, when the precondition "!precondition: (remote_endpoint_crypto_list)->_size != 1" was logged, B may have entered into a state that prevented any further communication with A2.

This problem has been resolved. The issues leading to the errors above have been fixed. Moreover, in addition to the existing precondition, the following error message has been added to both the release and debug libraries to identify if this unexpected situation (or a similar one) happens in the future (which should never be the case):

```
"Only one remote endpoint crypto handle is expected for the Secure Volatile channel, instead
there are 2."
```

[RTI Issue ID SEC-1141]

4.2.2 Unexpected "DecryptFinal" failures may have resulted in unnecessary increased or dropped traffic

There was an issue with receiver-specific MACs validation for ACK and GAP submessages that resulted in network traffic being incorrectly dropped and possibly increased. When this issue triggered, the following message was logged at DEBUG logging.verbosity (or the equivalent logging.log_level in 6.0.1 and below):

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

This problem has been fixed. Now all protected GAP/ACK messages are properly validated and the log message included above should no longer appear because of this particular issue. Note that the log message mentioned above may still be logged in the following two cases:

- At ERROR verbosity in case of real submessage tampering/corruption.
- At DEBUG verbosity if communicating with DDS Security implementations that do not disable multicast for the Secure Volatile Channel (this might be the case when interoperating with Security Plugins prior to 6.1.0 or other vendor plugins).

[RTI Issue ID SEC-1061]

4.2.3 Potential unexpected protection of certain submessages

In certain cases, a *DataWriter* setting <metadata_protection_kind> to a value other than NONE may have protected submessages that should not be protected according to the DDS Security specification.

Specifically, this may have happened if the buffer the *DataWriter* uses to compose RTPS messages became full with submessages while building an RTPS message, and thus the *DataWriter* needed to split the submessages into two different RTPS messages. In this scenario, the second RTPS message may have incorrectly contained protected INFO_DST or INFO_TS submessages (by the OMG DDS Security specification, INFO_DST and INFO_TS should not be protected).

While this would have not prevented communication, it was not compliant with the OMG DDS Security specification, and it might have provoked the second RTPS message to be dropped by other DDS implementations, potentially generating unnecessary traffic.

[RTI Issue ID SEC-1144]

4.2.4 Potential unexpected protection of certain submessages within the same RTPS message

In certain cases, a *DataWriter* setting <metadata_protection_kind> to a value other than NONE may have protected submessages that should not be protected according to the DDS Security specification.

For example, this was the case when adding multiple samples (with different timestamps) as part of the same RTPS message. Another case was composing an RTPS message that was directed to multiple destinations. In these scenarios, the resulting RTPS message may have incorrectly contained a mix of protected and unprotected INFO_DST or INFO_TS submessages (by the OMG DDS Security specification, INFO_DST and INFO_TS should not be protected).

While this would not have prevented communication, it was not compliant with the OMG DDS Security specification, and it may have provoked the second RTPS message to be dropped by other DDS implementations, potentially generating unnecessary traffic.

[RTI Issue ID SEC-1168]

4.2.5 INFO_DST destination information ignored when parsing secure submessages or enabling CRC

An INFO_DST submessage contains the GUID Prefix of the destination reader for the submessages coming after it (until the end of the RTPS message or a new INFO_DST).

Previously, some types of submessages skipped this validation. As a result, some submessages (for example, secure submessages) were incorrectly processed after processing an INFO_DST indicating that the destination reader was not the one receiving the submessages. Decryption may have then failed (because the reader trying to decode the submessages was not the reader the messages were addressed to), and *Connext DDS* would have logged errors.

This problem has been resolved. *Connext DDS* now properly enforces the destination restrictions derived from received INFO_DST submessages in all cases, and submessages whose destination don't match the receiving reader will be silently dropped.

[RTI Issue ID SEC-1109]

4.2.6 Incorrect certificate verification failure when CRL was not applicable

Certificate verification incorrectly failed when the *DomainParticipant* was configured to have a **crl_file** that was signed by a different CA than the one that signed the **identity_certificate**. For example, consider the following scenario:

- **identity_ca** has signed the crl_file.
- One of the **alternative_ca_files** has signed the **identity_certificate**.
- Certificate verification would fail with error 8: CRL signature failure.

This verification failure resulted in a failure to either create or complete discovery with a *DomainParticipant*. This problem has been fixed. As long as no element in the **identity_certificate** chain has been revoked by its signer, certificate verification will now succeed.

[RTI Issue ID SEC-1117]

4.3 Fixes Related to Discovery and Entity Matching

4.3.1 get_matched APIs incorrectly included endpoints with pending key material

If a *DataWriter* and a matching *DataReader* had **metadata_protection_kind** or **data_protection_kind** equal to SIGN or ENCRYPT, then **DDS_DataWriter_get_matched_subscription_data()** incorrectly returned DDS_RETCODE_OK if the *DataWriter* had not yet received the *DataReader's* key material. **DDS_DataReader_get_matched_publication_data()** had a similar problem.

Also, **DDS_DataWriter_get_matched_subscriptions()** and **DDS_DataReader_get_matched_publications()** incorrectly included subscriptions and publications whose key material had not yet been received. Consequently, if you were relying on those functions to determine when to start writing samples, it was possible to start writing protected samples before a *DataReader* was able to decode them or before a *DataWriter* was able to decode ACKNACKs. Those samples or ACKNACKs would have been dropped.

All four functions have been fixed. **DDS_DataWriter_get_matched_subscription_data()** and **DDS_DataReader_get_matched_publication_data()** now return DDS_RETCODE_PRECONDITION_NOT_MET in this situation. **DDS_DataWriter_get_matched_subscriptions()** and **DDS_DataReader_get_matched_publications()** no longer include endpoints whose key material is pending reception.

[RTI Issue ID SEC-1064]

4.3.2 DataWriter may have reported unexpected PUBLICATION_MATCHED_STATUS for a DataReader

Under the following sequence of events, a *DataWriter* setting <metadata_protection_kind> or <data_protection_kind> to a value other than NONE would have reported an unexpected PUBLICATION_MATCHED_STATUS for a *DataReader*:

1. The *DataWriter* discovered the *DataReader*, but the key material for the *DataReader* was not available yet. In this case, the *DataWriter* did not report PUBLICATION_MATCHED_STATUS yet because the *DataReader* was not fully matched. This is expected.
2. The *DataReader* left the system.
3. The *DataWriter* received key material from the *DataReader* immediately after the *DataReader* left the system.

In this scenario, the *DataWriter* should not have reported any PUBLICATION_MATCHED_STATUS, since the *DataReader* was never fully matched nor reported to the user. However, the *DataWriter* incorrectly reported a PUBLICATION_MATCHED_STATUS for the *DataReader* with a **current_count_change** of -1.

This problem has been resolved. The *DataWriter* will no longer report an unexpected PUBLICATION_MATCHED_STATUS for a *DataReader* that had never been fully matched.

[RTI Issue ID SEC-1151]

4.4 Fixes Related to Interoperability with Other Vendors

4.4.1 Lack of communication with DomainParticipant from different DDS vendor

A *Connext DDSDomainParticipant* with security enabled may not have communicated with a DomainParticipant from a different DDS vendor.

This only occurred when any of these conditions were met:

- The other DDS vendor sent directed writes on the DCPSParticipantVolatileMessageSecure or DCPSParticipantStatelessMessage builtin Topic.
- The other DDS vendor parsed the directed write (PID_DIRECTED_WRITE) inline QoS parameter that was sent by the *Connext DDS DomainParticipant*.

[RTI Issue ID SEC-1074]

4.4.2 Diffie-Hellman public key did not match DDS Security specification

The DDS Security specification states that if the key agreement algorithm is "DH+MODP-2048-256", then the Diffie-Hellman public key shall be according to IETF RFC 5114. Previous releases incorrectly conformed to IETF RFC 3526 when the property **authentication.shared_secret_algorithm** was set to "dh". This behavior prevented the *Security Plugins* from interoperating with correctly implemented security plugins from other DDS vendors when using Diffie-Hellman.

Starting with this release, the Diffie-Hellman key agreement conforms with the specification. To avoid breaking backward compatibility with previous versions of the *Security Plugins*, a DH public key according to IETF RFC 3526 is generated for key agreement with remote participants of older versions of the *Security Plugins.*

[RTI Issue ID SEC-1214]

4.5 Fixes Related to Observability

4.5.1 Wrong function name in heap allocation-related error messages

If an error occurred during heap allocation within the *Security Plugins*, the log message included the name of the parent function, which is inconsistent with the rest of *Connext DDS* heap-related errors.

This problem has been resolved. Now the log message will refer to the name of allocating function (not its parent).

[RTI Issue ID SEC-991]

4.5.2 Wrong logging distribution property names

In previous releases, the documented names of the properties to configure the logging thread threshold did not match the ones the Security Plugins expected. In particular, the following three property names were not correctly parsed by the Security Plugins:

- logging.distribute.thread.message_threshold
- logging.distribute.thread.plugin_method_threshold
- logging.distribute.thread.plugin_class_threshold

Instead, the plugins were expecting the following, wrong names:

- logging.distribute.queue.thread.message_threshold
- logging.distribute.queue.thread.plugin_method_threshold
- logging.distribute.queue.thread.plugin_class_threshold

The names for the properties that configure the logging thread thresholds have been updated.

Old Property Name	New Property Name
logging.distribute.queue.thread.message_threshold	logging.security_topic.thread.message_threshold
logging.distribute.queue.thread.plugin_method_threshold	logging.security_topic.thread.plugin_method_threshold
logging.distribute.queue.thread.plugin_class_threshold	logging.security_topic.thread.plugin_class_threshold

You must update the property names if you were using the ones including “queue” in their name. Attempting to use the “queue” properties will now fail during property validation.

No code changes are required if you were already using the ones without “queue” in their name. With that said, we recommend updating your properties to use the new properties, which include “security_topic” instead of “distribute”, because a future release may stop supporting the “distribute” version of the properties.

[RTI Issue ID SEC-1162]

4.6 Fixes Related to Scalability

4.6.1 Dynamically loaded Security Plugins library was never unloaded

When loading a *Security Plugins* library (e.g., **nddssecurity**) dynamically, the library was never unloaded. This problem has been fixed by unloading the library when the last *DomainParticipant* in the DomainParticipantFactory is deleted.

Note: You may still see "still reachable" memory leaks in **dlopen** and **dlclose**. These leaks are a result of a bug in Valgrind™ (https://bugs.launchpad.net/ubuntu/+source/valgrind/+bug/1160352).

[RTI Issue ID SEC-1026]

4.6.2 Memory leak when setting logging.security_topic.profile (formerly logging.distribute.profile)

Setting the property **logging.security_topic.profile** (formerly **logging.distribute.profile**) resulted in a memory leak in the function **RTI_Security_Logging_set_log_options()**, which is called during *DomainParticipant* creation. This memory leak has been fixed.

[RTI Issue ID SEC-1085]

4.6.3 Memory leak if certificate chain failed to be verified against CA

A memory leak occurred in the function **PEM_read_bio_X509_AUX()** when an **identity_certificate** containing a chain of at least two certificates failed to be verified against the **identity_ca** or one of the **alternative_ca_files**. This leak occurred while either creating or discovering a *DomainParticipant*. This problem has been fixed.

[RTI Issue ID SEC-1169]

4.6.4 Potentially very long recovery times for timed-out authentications

An issue may have caused two *DomainParticipants* to take a very long time to recover from an unsuccessful authentication negotiation. Specifically, the issue may have occurred if the following three conditions were met:

- One of the *DomainParticipants* completed authentication within the authentication timeout (configured through the **dds.participant.trust_plugins.authentication_timeout.sec** property).
- The other *DomainParticipant* timed out the authentication.
- The authentication request delay (configured through the **dds.participant.trust_plugins.authentication_request_delay.sec** property) was set to a value lower than the authentication time out (i.e., the authentication request recovery mechanism was enabled).

Under these circumstances, the two involved participants may have entered into a state where the first participant remained in an authenticated state, while the second participant continuously started new authentication negotiations and failed them after the configured authentication timeout.

This problem has been resolved. Two *DomainParticipants* will no longer enter into this state, and they will complete authentication in a timely manner. In addition, the following property has been added to provide more control over the authentication negotiation: **dds.participant.trust_plugins.authentication_request_timeout.sec**.

This property determines the timeout (in seconds) for authentication negotiations started from an authentication request message (authentication request is a DDS Security 1.1 mechanism to securely recover from an asymmetric liveliness loss). The default value is 20 seconds. If this property is set to a value greater than **dds.participant.trust_plugins.authentication_timeout.sec**, then the value in **dds.participant.trust_plugins.authentication_timeout.sec** will be used instead.

To minimize authentication negotiation times during system startup, follow these guidelines:

- Set **dds.participant.trust_plugins.authentication_timeout.sec** to a value that is twice the time it takes to authenticate all of the *DomainParticipants* during the system startup. A too short value will trigger additional authentication negotiations, generating additional CPU load and network traffic, and generally slowing down the system startup.
- Set **dds.participant.trust_plugins.authentication_request_delay.sec** to a value that is higher than the time it takes to authenticate all of the *DomainParticipants* during the system startup. A too short value will generate additional traffic and potentially additional unnecessary authentication negotiations.
- Set **dds.participant.trust_plugins.authentication_request_timeout.sec** to the average time it takes to authenticate a *DomainParticipant* in your system at startup.
- Make sure that the sum of the configured values for **authentication_request_delay** and **authentication_request_timeout** is lower than the **authentication_timeout**.

[RTI Issue ID SEC-1203]

4.6.5 Inefficient decoding of samples in keyed DataReaders

For a DataReader setting <data_protection_kind> to a value other than NONE, when **dds.data_reader.history.memory_manager.fast_pool.pool_buffer_max_size** is set to a value other than UNBOUNDED, *Connext DDS* creates a decoding buffer that is reused by the *DataReader* to decode all of the received samples with a serialized sample size smaller than the value configured in **dds.data_reader.history.memory_manager.fast_pool.pool_buffer_max_size**. For samples with a bigger size than **dds.data_reader.history.memory_manager.fast_pool.pool_buffer_max_size**, the *DataReader* dynamically allocates and releases a temporary buffer.

In previous releases, there was an issue that prevented the decoding buffer from being created, making the *DataReader* allocate and free a temporary buffer for all of the received samples.

This problem has been resolved. *DataReaders* now correctly create the decoding buffer when **dds.data_reader.history.memory_manager.fast_pool.pool_buffer_max_size** is configured.

[RTI Issue ID SEC-1128]

4.6.6 Key Exchange (Secure Volatile) DataWriter was inefficient when repairing samples in some cases

The Key Exchange (Secure Volatile) *DataWriter* was inefficient when repairing samples in some cases. Specifically, this inefficiency was triggered when repairing more than 32 samples at the same time, resulting in additional traffic.

This problem has been resolved. The Secure Volatile *DataWriter* no longer generates additional traffic in this scenario.

[RTI Issue ID SEC-1143]

4.7 Fixes Related to Security Plugins SDK

4.7.1 SDK compilation failure after "make clean"

The *Security Plugins* SDK compilation would fail if the command "make clean" was executed. This problem has been fixed.

[RTI Issue ID SEC-769]

4.8 Other Fixes

4.8.1 Segmentation fault when using DomainParticipantListener with Security Plugins

While enabling a *DomainParticipant*, a segmentation fault would occur in the function **DDSDomainParticipantListener_forward_onPublicationMatched()** when using a DomainParticipantListener with the *Security Plugins* enabled. This issue has been fixed.

[RTI Issue ID SEC-1029]

4.8.2 Participant creation failed when using authentication.keyform property in Security Plugins SDK

If you tried to set the property **com.rti.serv.secure.authentication.keyform**, participant creation failed with the following error:

```
DDS_PropertyQosPolicy_validatePropertyNames:Unexpected property:
com.rti.serv.secure.authentication.keyform. Closest valid property:
com.rti.serv.secure.authentication.crl_file
```

This problem has been resolved.

[RTI Issue ID SEC-1072]

4.8.3 Possible crash if Security Logging Plugin deleted while it was logging a message

There was an unexpected scenario where the Security Logging Plugin was logging a message while another thread was deleting the plugin. This problem could lead to a crash in the logging operation.

The problem has been resolved. Now deletion of the Security Logging Plugin will be synchronized with the logging operation. So the Logging Plugin can only be deleted if there are no threads using it. We will wait for a timeout, so any thread using the plugin has the opportunity to finish. In the (unexpected) case there are still threads using the plugin after the timeout, the deletion will be skipped to prevent a crash.

[RTI Issue ID SEC-1255]

4.8.4 remove_peer() caused delete_participant() to hang

When using security, the function **DDS_DomainParticipant_remove_peer()** did not work. With debug libraries, **DDS_DomainParticipant_remove_peer()** generated a precondition error while calling the internal function **PRESParticipant_getRemoteParticipantInterceptorHandleNodePt()**, and the peer would fail to be removed. With release libraries, a later call to the function **DDS_DomainParticipant_add_peer()** would be ineffective, and a later call to the function **DDS_DomainParticipantFactory_delete_participant()** would hang.

This problem has been resolved. **DDS_DomainParticipant_remove_peer()** now works when using security.

[RTI Issue ID SEC-1261]

Chapter 5 Known Issues

5.1 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
```

5.2 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]

5.3 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 DDS* 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]

5.4 relay_only parameter in check_remote_datareader is not Populated

When calling **check_remote_datareader**, *Connext DDS* 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]

5.5 Possible Valgrind Still-Reachable Leaks when Loading Dynamic Libraries

If you load any dynamic libraries, you may see "still reachable" memory leaks in **dlopen** and **dlclose**. These leaks are a result of a bug in Valgrind (https://bugs.launchpad.net/ubuntu/+source/valgrind/+bug/1160352).

[RTI Issue ID SEC-1026]

5.6 '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]

5.6.1 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 DDS 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 DDS 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]