RTI Security Plugins

Release Notes

Version 6.0.0

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 6.0.0 is supported on the following platforms.

Table 1.1 Supported Platformss

Operating System	Version
Android™	All platforms listed in the *RTI Connext® DDS Core Libraries Release Notes* for the same version number, except not supported on SUSE® platforms.
iOS®	
Linux®	
OS X®	
QNX®	All platforms listed in the *RTI Connext DDS Core Libraries Release Notes* for the same version number, except not supported on QNX Neutrino® 6.4.1.
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.

Security Plugins is 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
Linux	Debian® 7 RedHawk™ Linux 6.5 Ubuntu® 18.04 LTS Wind River® Linux 8
QNX	QNX Neutrino 6.6

Chapter 2 Compatibility

Security Plugins 6.0.0 is interoperable with 5.2.7 and higher versions of *Security Plugins*.

Security Plugins 6.0.0 is API-compatible with OpenSSL® versions 1.0.1c through 1.0.2o. It is not API-compatible with OpenSSL® 1.1.0a or above. Note that *Security Plugins* 6.0.0 has only been tested by RTI using OpenSSL 1.0.2o.

Persistence Service databases secured with Security Plugins 6.0.0 are incompatible with databases generated by older versions of *Persistence Service*.

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

For backward compatibility information between 6.0.0 and previous releases, 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 6.0.0

3.1 New Platforms

This release adds support for the following platforms:

Operating System	CPU	Compiler	RTI Architecture Abbreviation
Debian 7	ARMv7	gcc 4.9.3	armv7aLinux3.12gcc4.9.3cortex-a9 (Custom platform)
Ubuntu 18.04 LTS	x64	gcc 7.3.0	x64Linux4gcc7.3.0
	ARMv8 64-bit	gcc 7.3.0	armv8Linux4gcc7.3.0 (Custom platform)
Wind River Linux 8	ARMv7	gcc 5.2.0	armv7aWRLinux8gcc5.2.0 (Custom platform)

3.2 Updated OpenSSL Version

This release uses OpenSSL 1.0.2o (instead of 1.0.2n).

3.3 Changes Related to Specification Compliance

This release adds the following changes to be compliant with the Builtin Security Plugins defined in the DDS Security specification[1].

3.3.1 Authentication and Discovery

3.3.1.1 Option to specify private key password

This feature is related to the DDS Security specification properties used to configure the builtin Authentication plugin.

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

You may now encrypt the value of the property **dds.sec.auth.private_key** and specify its decryption password as the property **dds.sec.auth.password**. See the tables at the beginning of the "Authentication" chapter in the *RTI Security Plugins Getting Started Guide* for details.

3.3.1.2 Added support for new participant and endpoint security information

This release adds support for the new DDS Security 1.1 participant/endpoint security matching, which allows for early detection of incompatible security configurations.

To support this feature, participants will announce new parameters as part of the participant/endpoint discovery. These discovery parameters have the following attributes:

Discovery Type	Member Name	Member Type	Parameter ID Name	Parameter ID Value
Endpoint	security_info	EndpointSecurityInfo	PID_ENDPOINT_SECURITY_INFO	0x1004
Participant	security_info	ParticipantSecurityInfo	PID_PARTICIPANT_SECURITY_INFO	0x1005

Note that while these members are propagated and used for doing discovery matching, currently they are not exposed as part of any public API. Moreover, an incompatible security_info configuration is currently not reported as part of the OFFERED_INCOMPATIBLE_QOS/REQUESTED_INCOMPATIBLE_QOS statuses. Therefore, a security_info incompatibility will not trigger **on_offered_incompatible_qos()/on_requested_incompatible_qos()** callbacks.

The propagation of these parameters is enabled by default. Since their propagation introduces additional restrictions for participant and endpoint matching, the following new properties have been introduced to allow for keeping the old behavior:

Property Name[1]	Property Value Description
dds.participant.discovery_config.disable_endpoint_security_info_propagation	***Optional*** If set to FALSE, the endpoint's security_info is propagated and a *DataWriter*/*DataReader* pair needs to use the same security configuration to match. If set to TRUE in both participants, contained *DataWriters* and *DataReaders* may communicate for some combinations of inconsistent metadata/data protection kinds. Default: FALSE
dds.participant.discovery_config.disable_participant_security_info_propagation	***Optional*** If set to FALSE, the participant's security_info is propagated and a pair of participants needs to use the same Governance's RTPS, discovery, and liveliness configurations to match. If set to TRUE in both participants, participants may communicate for some combinations of inconsistent Governance's RTPS, discovery, and liveliness configurations. Default: FALSE

[1]These new properties do not need to be prefixed with 'com.rti.serv.secure.'

3.3.1.3 Added new DCPSParticipantsSecure builtin topic

This release adds a new builtin topic, “DCPSParticipantsSecure,” introduced by the DDS Security 1.1 specification. This topic is secured following the same rules as the Secure Endpoint Discovery topics (that is, it is configured through the Governance's **discovery_protection_kind** parameter).

To support this new topic, *Connext DDS* creates two new reliable endpoints (a *DataWriter* and a *DataReader*) when enabling security for a participant. The entity IDs for those endpoints are as follows:

Endpoint Name	Entity ID Definition	Entity ID Value
SPDPbuiltinParticipantsSecureWriter	ENTITYID_SPDP_BUILTIN_PARTICIPANTS_SECURE_WRITER	{{ff, 01, 01}, c2}
SPDPbuiltinParticipantsSecureReader	ENTITYID_SPDP_BUILTIN_PARTICIPANTS_SECURE_READER	{{ff, 01, 01}, c7}

When those endpoints are enabled, the following bits are set in the ParticipantBuiltinTopicData’s availableBuiltinEndpoints:

Builtin Endpoint	Bit in the ParticipantBuiltinTopicData's availableBuiltinEndpoints
SPDPbuiltinParticipantsSecureWriter	(0x00000001 << 26)
SPDPbuiltinParticipantsSecureReader	(0x00000001 << 27)

Starting in this release, once a remote participant is authenticated, any changes affecting the participant discovery data must be exchanged using the “DCPSParticipantsSecure” topic. To avoid breaking backwards compatibility with previous versions of *Connext DDS*, old remote participants will still rely on the existing TrustedState mechanism described in the section Protecting Participant Discovery, in the *RTI Security Plugins Getting Started Guide,* to propagate Participant discovery updates for authenticated Participants.

3.3.1.4 Support for DDS Security 1.1 AuthRequest

This release updates the *Security Plugins* re-authentication mechanism to be compliant with the new AuthRequest mechanism described in the DDS Security 1.1 specification. The changes are as follows:

- Changed GenericMessageClassId from "com.rti.sec.auth.request" to "dds.sec.auth_request".
- Changed Token’s class_id from "com.rti.sec.DDS:Auth:PKI-DH:1.0+AuthReq" to "DDS:Auth:PKI-DH:1.0+AuthReq".
- Updated **validate_remote_identity** to include AuthRequestMessageToken parameters.
- Removed **begin_auth_request** and **process_auth_request** APIs from the Authentication Plugin interface, because their logic is now part of the **validate_remote_identity** API. Note that *RTI Security Plugins SDK* still keeps those functions (now as private functions), since they are called from the **validate_remote_identity** function implementation.

Note that these changes **do not break backwards compatibility**: 6.0.0 participants will use either the old or the new re-authentication GenericMessageClassId and Token's class_id, depending on the detected remote participant's version (which is exchanged as part of participant discovery).

3.3.2 Access Control

3.3.2.1 Updated PermissionsToken class_id

The DDS Security 1.1 specification states that in the Builtin Access Control plugin, the PermissionsToken class_id shall be "DDS:Access:Permissions:1.0". *Security Plugins* has updated its PermissionsToken from "DDS:Access:Permissions" to "DDS:Access:Permissions:1.0". Note that this change does not affect compatibility because the specification states that if MajorVersion and MinorVersion are missing from the class_id, the class_id shall be interpreted as being MajorVersion 1 and MinorVersion 0. So the legacy class_id is equivalent to the new one.

3.3.2.2 Added check_remote_topic

The DDS Security specification describes the Access Control plugin operation **check_remote_topic()**. This function is now invoked and implemented. You will see no impact when using the builtin plugins. The function will not be invoked if either **enable_read_access_control** or **enable_write_access_control** is FALSE in the local DomainParticipant’s Governance document’s corresponding <topic_rule> tag. The function will return TRUE if the remote DomainParticipant's Permissions document allows a *DataWriter* or a *DataReader* of that topic-DomainParticipant combination.

3.3.2.3 Added functions to return security attributes

The DDS Security 1.1 specification introduces the Access Control plugin operations **return_participant_sec_attributes()**, **return_datawriter_sec_attributes()**, and **return_datareader_sec_attributes()**. These functions are now invoked and implemented. The builtin plugin implementation does nothing in these functions. If you implement a custom plugin that populates the PropertySeq within ParticipantSecurityAttributes or EndpointSecurityAttributes, then these functions should finalize the PropertySeq.

3.3.2.4 Data tagging

The DDS Security specification describes data tagging using the DataTagQosPolicy. *Security Plugins* now supports this policy and its usage in the Access Control plugin.

3.3.2.5 Support for DDS Security Topic Security Attributes

This release adds support for DDS Security 1.1 TopicSecurityAttributes and the associated **get_topic_security_attributes** API.

As part of this feature, EndpointSecurityAttributes now inherits from TopicSecurityAttributes, and some of the members of EndpointSecurityAttributes have been moved to TopicSecurityAttributes. For more information about these changes, see the *Migration Guide* on the RTI Community Portal (https://-community.rti.com/documentation).

3.3.2.6 Added Support for builtin topics to get_datawriter_sec_attributes and get_datareader_sec_attributes APIs

Previously, endpoint security attributes for builtin topic endpoints were hardcoded at the core libraries level; they were not modifiable by *Security Plugins* (or any custom security plugin).

Starting with this release, core libraries get the endpoint security attributes for the builtin topics using the **get_datawriter_sec_attributes** and **get_datareader_sec_attributes** APIs as described by the DDS Security 1.1 specification, Section 7.4.8, Securing the "Builtin Secure Endpoints."

To support this new mechanism, *Security Plugins's* **get_datawriter_sec_attributes** and **get_datareader_sec_attributes** APIs have been updated to support retrieving attributes for DDS and DDS Security builtin topics, as well as for RTI ("vendor-specific") builtin topics.

For more information about how this feature affects custom security plugins, please refer to the *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation).

3.3.2.7 <topics> now mandatory in permissions file

The DDS Security specification's XSD schema file omg_shared_ca_permissions.xsd indicates that the <topics> element is mandatory inside a <publish> or <subscribe> element. *Security Plugins* now enforces this rule.

3.3.2.8 Updated matching behavior of allowed partitions condition

The DDS Security specification describes the matching behavior of the <partitions> section within an <allow_rule> of a Permissions file. In order for a *DataWriter* or *DataReader* to be matched with an "allowed partitions" condition, the DDS entity's partitions must be a subset of the partitions in the condition. This release enforces this matching rule.

To change this behavior, you may set the security plugin property **access_control.use_530_partitions** to TRUE. If TRUE, then a *DataWriter* or *DataReader* will be matched with an "allowed partitions" condition as long as at least one of the DDS entity's partitions matches one of the partitions in the condition; this is consistent with Connext 5.3.0 behavior. If FALSE, then the entity's partitions must be a subset of the condition's partitions; this is consistent with the behavior of the DDS Security specification. The default value is FALSE.

Here's an example:

DataWriter Partitions	Allowed Partitions Condition	use_530_partitions	allowed?
[A, B]	[B, C]	TRUE	yes, because B is in [B, C]

DataWriter Partitions	Allowed Partitions Condition	use_530_partitions	allowed?
[A, B]	[B, C]	FALSE	no, because A is not in [B, C]

3.3.3 Cryptography

3.3.3.1 New protection kinds

This feature is related to the Domain Governance Document described in the DDS Security specification.

The following previously unsupported protection kinds are now supported:

- **rtps_protection_kind** = ENCRYPT
- **metadata_protection_kind** = SIGN
- **data_protection_kind** = SIGN

3.4 Other Changes

3.4.1 Authentication and Discovery

3.4.1.1 Identity Certificate chaining

You may now put a chain of certificates in the Identity Certificate by concatenating individual certificates and specifying the concatenated result as a single file or string. The Identity Certificate will be verified against the Identity CA using the following procedure:

- The current certificate is the first certificate in the Identity Certificate chain.
- Perform the following steps up to and including the case when the current certificate is the last certificate in the Identity Certificate chain:
  - If the current certificate is signed by the Identity CA, then the verification succeeds immediately.
  - Otherwise:
    - If a next certificate exists in the chain and the current certificate is signed by that next certificate, then the next certificate becomes the current certificate.
    - Otherwise, verification fails immediately.

3.4.2 Cryptography

3.4.2.1 Support using different writer keys for protecting submessages and serialized data

This release adds support for using different key material for protecting the submessages and serialized data encoded by a *DataWriter*.

By default, *DataWriters* with metadata and data protection kinds other than NONE use the same key material for encoding both submessages and serialized data. To change this behavior, this release adds a new property to the Cryptography plugin:

- **cryptography.share_key_for_metadata_and_data_protection:** Determines whether the metadata and data encoding operations share the same key material or use different keys. Default: TRUE (they share key material).

Note that setting this property to FALSE (that is, using different keys for protecting submessages and serialized data) will break backward compatibility with older versions of *Security Plugins* when both **metadata_protection_kind** and **data_protection_kind** are set to a value other than NONE.

3.4.2.2 WITH_ORIGIN_AUTHENTICATION protection kinds

In the Governance Document, you may now use the protection kinds ENCRYPT_WITH_ORIGIN_AUTHENTICATION and SIGN_WITH_ORIGIN_AUTHENTICATION as explained in the DDS Security specification. WITH_ORIGIN_AUTHENTICATION adds receiver-specific Message Authentication Codes (MACs) to the encoded output. WITH_ORIGIN_AUTHENTICATION may not be used if the property **com.rti.serv.secure.cryptography.max_receiver_specific_macs** is 0. The protection kind values ENCRYPT and SIGN no longer add receiver-specific MACs to the encoded output.

3.4.2.3 Updated configuration property names

This feature is related to the DDS Security specification properties used to configure the builtin Authentication and Access Control plugins.

You may now use the property names that are in the DDS Security specification. The legacy property names are still supported. The following properties are affected:

Legacy Property Name (prefix with 'com.rti.serv.secure')	New Property Name (no prefix)
authentication.ca_file	dds.sec.auth.identity_ca
authentication.private_key_file	dds.sec.auth.private_key
authentication.certificate_file	dds.sec.auth.identity_certificate
access_control.permissions_authority_file	dds.sec.access.permissions_ca
access_control.governance_file	dds.sec.access.governance
access_control.permissions_file	dds.sec.access.permissions

3.4.2.4 Option to specify file contents instead of file name

This feature is related to the DDS Security specification properties used to configure the builtin Authentication and Access Control plugins.

Many of the security properties required a file name as the value. You may now use the contents of the file, prefixed by "data:,", as the value. For example, if the file privateKey.pem contains

```
-----BEGIN PRIVATE KEY-----
abc
def
-----END PRIVATE KEY-----
```

you previously had to specify a property with name **com.rti.serv.secure.authentication.private_key_file** and value "privateKey.pem". This option is still possible, but you may alternatively specify a property with name **dds.sec.auth.private_key** and the value:

```
"data:,-----BEGIN PRIVATE KEY-----\nabcdef\n-----END PRIVATE KEY-----"
```

The two '\n' characters surrounding "abcdef" are required. A '\n' is not required between 'c' and 'd'.

3.4.2.5 Data fragmentation support for Authentication and Key Exchange builtin topics

This release adds support for data fragmentation of the Authentication topic (ParticipantStatelessMessage builtin topic) and asynchronous publishing of the Key Exchange topic (ParticipantSecureVolatileMessageSecure builtin topic), when security is enabled. This feature addresses the scenario in which a security-enabled environment requires DDS-level fragmentation due to a hard limit on the maximum transport message size.

In the case of the Authentication builtin topic, data fragmentation is supported by default. To enable data fragmentation for the Key Exchange topic, you need to enable asynchronous publishing through DiscoveryConfig's **secure_volatile_writer_publish_mode** field.

For more information, refer to Enabling Asynchronous Publishing for the Key Exchange Topic, in the *RTI Security Plugins Getting Started Guide*, and PUBLISH_MODE QosPolicy (DDS Extension), in the *RTI Connext DDS Core Libraries User's Manual*.

3.4.3 New APIs

New APIs are provided to get the *Security Plugins* version:

- RTI_Security_get_build_version_string()
- RTI_Security_get_library_version()

3.4.4 Persistence Service

3.4.4.1 Changed default Persistence Service dds.data_writer. history.key_material_key

The undisclosed non-NULL default **dds.data_writer.history.key_material_key** has changed. As a result, *RTI Persistence Service* databases protected with the old default key will not be accessible by the new *Persistence Service*.

Note that using the default key is discouraged, and you should set **dds.data_writer.history.key_material_key** to a value other than the default.

3.4.4.2 Improved the algorithm to derive a key used to encrypt Persistence Service's encryption key

In the builtin plugins, the key derivation algorithm applied to the **dds.data_writer.history.key_material_key** has improved. The algorithm now involves PBKDF2 (Password-Based Key Derivation Function) with SHA-512 (Secure Hash Algorithm with a 512-bit hash value) and a random salt. *Persistence Service* now stores the random salt along with the PRSTDataWriter’s encrypted key. As a result, databases protected with the old *Persistence Service* will not be accessible by the new *Persistence Service*.

3.4.5 Shapes Demo

3.4.5.1 Added Shapes Demo CA key to Shapes Demo resource folder

This release adds the *Shapes Demo* CA key file (RTI_SHAPES_DEMO_CA_KEY.pem) to the *Shapes Demo* resource\cert folder.

This file is useful for generating new signed Governance and Permissions files that can be used with the shipped *Shapes Demo* certificates.

Chapter 4 What's Fixed in 6.0.0

This section describes bugs that have been fixed in *Security Plugins* 6.0.0.

4.1 Fixes Related to Specification Compliance

4.1.1 Input parameters to Security SPI functions do not have "const"

The header file dds_c/dds_c_trust_plugins.h defines the Security Service Plugin Interface (SPI) functions (for example, DDS_Authentication_ValidateRemoteIdentityFunction). The non-primitive input parameters of many of these functions did not have "const" preceding them. This problem has been resolved. The non-primitive input parameters now have "const" preceding them. For example:

```
typedef
DDS_ValidationResult_t (*DDS_Authentication_ValidateRemoteIdentityFunction)(
    struct DDS_AuthenticationPlugin *auth,
    DDS_IdentityHandle *remote_identity_handle, /* out */
    DDS_AuthRequestMessageToken *local_auth_request_token, /* out */
    const DDS_AuthRequestMessageToken *remote_auth_request_token,
    const DDS_IdentityHandle local_identity_handle,
    const DDS_IdentityToken *remote_identity_token,
    const struct DDS_GUID_t *remote_participant_guid,
    DDS_TrustException *exception);
```

[RTI Issue ID SEC-251]

4.1.2 Mutability of Publisher PartitionQosPolicy

The Publisher PartitionQosPolicy was always mutable, which did not comply with the DDS Security specification. This problem has been resolved. The Publisher PartitionQosPolicy is now immutable if the Publisher contains any *DataWriter* that meets the following two criteria:

1. The TopicSecurityAttributes for that *DataWriter* have **is_read_protected** (which corresponds to <enable_read_access_control> in the Governance Document) set to TRUE.

2. The *DataWriter* has the DurabilityQos policy kind set to something other than VOLATILE.

[RTI Issue ID SEC-453]

4.1.3 Wrong inputs to validate_local_permissions

Inputs to the **validate_local_permissions** function in the Access Control plugin were wrong. This problem has been resolved by replacing the PermissionsCredential with the DomainId_t and the DomainParticipantQos. You will see no impact when using the builtin plugins because PermissionsCredential was never used, and the two new parameters are not used.

[RTI Issue ID SEC-707]

4.1.4 Inconsistent governance configuration incorrectly allowed

In previous releases, *Security Plugins* incorrectly allowed governance's **allow_unauthenticated_participants** to be set to TRUE while **rtps_protection_kind** was set to a value other than NONE. Now, trying to configure a Participant with this inconsistent configuration will result in a failure to create the Participant.

[RTI Issue ID SEC-726]

4.1.5 Wrong return code for operations disallowed by Security Plugins

The DDS Security specification adds an additional return code NOT_ALLOWED_BY_SECURITY, which shall be returned by any operation that fails because the security plugins do not allow it. This return code was incorrectly defined in *Connext DDS* as NOT_ALLOWED_BY_SEC with a value of 13. This return code has now been replaced in *Connext DDS* by NOT_ALLOWED_BY_SECURITY, whose value is 1000. Although *Connext DDS* has never used this return code, it has been replaced to be compliant with the DDS Security specification.

[RTI Issue ID SEC-737]

4.1.6 Authentication handshake failed if plugin MinorVersion was different

The DDS Security 1.1 specification states that in the Builtin Authentication plugin, the **validate_remote_identity()** function shall return VALIDATION_FAILED if the local and remote IdentityTokens have different values for PluginClassName or MajorVersion. *Security Plugins* was incorrectly returning VALIDATION_FAILED if the IdentityTokens had the same PluginClassName and MajorVersion but different values for MinorVersion. *Security Plugins* no longer fails the validation in this scenario.

[RTI Issue ID SEC-739]

4.1.7 RTI_Security_Exception type did not match DDS Security specification

In previous releases, the definition of RTI_Security_Exception type did not match the DDS Security specification. This problem is now resolved by using the following definition, which matches the specification:

```
typedef struct DDS_TrustException {
    char *message;
    DDS_Long code;
    DDS_Long minor_code;
} DDS_TrustException;

typedef DDS_TrustException RTI_Security_Exception;
```

[RTI Issue ID SEC-743]

4.1.8 Entity creation incorrectly succeeded when no governance rule found

The creation of DomainParticipants and Topics incorrectly succeeded when the Domain Governance document did not specify any rules for those entities. According to the DDS Security specification, entity creation should fail with a suitable "permissions error" if there is no governance rule for the entity. This problem has been resolved.

[RTI Issue ID SEC-750]

4.1.9 Builtin Logging Topic not protected

The DDS Security specification states that the Builtin Logging Topic shall use the governance XML tag <metadata_protection_kind>SIGN</metadata_protection_kind>. *Security Plugins* was incorrectly setting **metadata_protection_kind** to NONE. This problem has been resolved by setting **metadata_protection_kind** to SIGN. This change breaks configuration compatibility between this and previous releases when using a *DataReader* to subscribe to the Builtin Logging Topic. For details, see the *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation).

[RTI Issue ID SEC-772]

4.1.10 Access Control API definitions not compliant with specification

The following Access Control plugin interface APIs were not compliant with the DDS Security specification:

- **check_remote_datareader** was missing the relay_only parameter.
- **check_local_datawriter_match** and **check_local_datareader_match** were missing publication/subscription data and incorrectly including the tags parameter.

- **get_datawriter_sec_attributes** and **get_datareader_sec_attributes** APIs were missing.
- **get_endpoint_sec_attributes** was not compliant with the specification.

This release includes the following changes so that the Access Control plugin interface APIs are now compliant with the DDS Security specification:

- Updated **check_remote_datareader** to include the relay_only parameter.
- Updated **check_local_datawriter_match** and **check_local_datareader_match** to include publication/subscription data parameters and to remove the tags parameter.
- Removed the **get_endpoint_sec_attributes** API from the Access Control plugin interface.
- Added **get_datawriter_sec_attributes** and **get_datareader_sec_attributes** to the Access Control plugin interface. These APIs replace **get_endpoint_sec_attributes**.

[RTI Issue ID SEC-789]

4.1.11 Wrong output of enable_logging and log

The output of the **enable_logging** and **log** functions in the Logging Plugin was void, which was consistent with the body of the specification but not with dds_security_plugins_spis.idl. **enable_logging** and **log** now return DDS_Boolean to be consistent with the IDL file.

[RTI Issue ID SEC-869]

4.1.12 Possible discovery delays when communicating with other vendors

The DDS Security specification states that the BuiltinParticipantStatelessMessageReader and BuiltinParticipantVolatileMessageSecureReader have an implied content filter with the logical expression: "destination_participant_guid == GUID_UNKNOWN || destination_participant_guid == reader.participant.guid". *Security Plugins* did not apply this filter on those builtin readers. This problem may have caused severe discovery delays when communicating with other DDS Security vendors. This problem has been resolved; the filter is now applied.

[RTI Issue ID SEC-888]

4.2 Other Fixes

4.2.1 Potential crash in Spy, Ping, or Persistence Service when enabling Security Plugins Logging with distribution over DDS

Enabling *Security Plugins* Logging with distribution over DDS on *Spy*, *Ping*, or *Persistence Service* may have provoked a segfault. This problem is now resolved.

[RTI Issue ID SEC-734]

4.2.2 Insecure random seed implementation

Security Plugins was using a mix of time, process ID, and host ID to seed the OpenSSL random number generator. None of these values is truly random, however, so the security of the seed was weak. This problem has been resolved by using random bytes to generate the seed.

[RTI Issue ID SEC-753]

4.2.3 Wrong product version passed to Authentication plugin's set_remote_participant_info

The DDS_TrustedParticipantInfo's product version that was passed to the Authentication plugin's **set_remote_participant_info** RTI Extension API was wrong; it did not match the actual Participant's product version. This problem has been resolved.

[RTI Issue ID SEC-812]

4.2.4 Could not create topic if deny rule contained partition

If the XML Permissions document contains a <deny_rule>, and the rule has both <publish> and <subscribe> tags for a given topic, then the creation of that topic should fail if and only if neither the <publish> tag nor the <subscribe> tag contains any tags other than <topics> and <topic>. Topic creation, however, failed if either <publish> or <subscribe> contained <partitions> and <partition>. This behavior was incorrect because if the rule denied certain partitions, then any other partition was not denied and topic creation should have been allowed. (But it wasn't.) This problem has been resolved: topic creation is now allowed in this case.

[RTI Issue ID SEC-822]

4.2.5 Specifying "ecdh" as the shared_secret_algorithm resulted in "dh" behavior

Specifying "ecdh" as the value of the property **authentication.shared_secret_algorithm** incorrectly resulted in behavior equivalent to specifying "dh" as the value. This problem was introduced in *Security Plugins* 5.3.0 and has now been resolved.

[RTI Issue ID SEC-824]

4.2.6 OpenSSL global state not properly cleaned up when shutting down an application

A security-enabled application experienced leaks of memory blocks that were still reachable. These memory blocks belonged to OpenSSL global state. One example block came from the function ERR_get_state(). These leaks have been fixed.

[RTI Issue ID SEC-828]

4.2.7 Certificate Authority incorrectly required to be self-signed

The Certificate Authority (CA), which corresponds to the property **authentication.ca_file**, was incorrectly required to be a self-signed root CA. This problem has been resolved by allowing the **ca_file** to be a subordinate CA whose certificate is signed by a superior CA.

[RTI Issue ID SEC-839]

4.2.8 Segmentation fault when trying to decode a malformed RTPS message

According to the DDS Security specification, an encoded RTPS message must have an SRTPS_POSTFIX submessage. When trying to decode an RTPS message with an SRTPS_PREFIX but no SRTPS_POSTFIX, a segmentation fault would occur in the function **MIGRtpsTrustSubmessage_deserializePostfix**. This problem has been resolved by discarding such an RTPS message.

[RTI Issue ID SEC-842]

4.2.9 Unknown shared secret algorithm silently treated as "ecdh"

Setting the property **authentication.shared_secret_algorithm** to a value other than the supported values of **dh** or **ecdh** was equivalent to setting the property to **ecdh**. Similarly, discovering a *DomainParticipant* that was using an unsupported **shared_secret_algorithm** was equivalent to discovering a *DomainParticipant* that was using **ecdh**. These problems have been resolved by treating these scenarios as failure conditions. A *DomainParticipant* that attempts to use an unsupported **shared_secret_algorithm** will fail to be created or authenticated.

[RTI Issue ID SEC-856]

4.2.10 Data protection didn't work if fragmented samples were preallocated

A *DataReader* failed to receive samples in the following scenario:

- **data_protection_kind** was set to a value other than NONE.
- A *DataWriter* wrote fragmented samples whose actual serialized size was very close to or equal to the maximum serialized size.
- **reader_qos.reader_resource_limits.dynamically_allocate_fragmented_samples** was set to DDS_BOOLEAN_FALSE.

This problem has been resolved.

[RTI Issue ID SEC-871]

4.2.11 Persistence Service failure to store encoded samples in database

RTI Persistence Service did not work with security (**data_protection_kind** = ENCRYPT) if the actual serialized sample size was close or equal to the maximum serialized sample size. *Persistence Service* generated these errors when trying to store the samples in the database:

```
sample cannot be stored in database. Increase database_sample_buffer_max_size
```

This problem has been resolved.

[RTI Issue ID SEC-885]

4.2.12 Possible lack of SUBSCRIPTION_MATCHED_STATUS if a DataWriter lost liveliness with the DataReader

There was a race condition in which a *DataReader* may have never reported a SUBSCRIPTION_ MATCHED_STATUS change despite successfully matching and receiving data from a *DataWriter*. This race condition occurred if all of the following conditions were true:

- The *DataReader* set its liveliness **lease_duration** to a very small duration (on the order of milliseconds).
- The *DataReader* was communicating with a *DataWriter* with **metadata_protection_kind** or **data_ protection_kind** set to a value other than NONE.
- The *DataWriter* lost liveliness with the *DataReader* between the time the *DataReader* discovered the *DataWriter* and the time the *DataReader* received key material from the *DataWriter*.
- The *DataWriter* regained liveliness with the *DataReader* after the *DataReader* received key material from the *DataWriter*.

This problem has been resolved by making sure that the SUBSCRIPTION_MATCHED_STATUS change is reported as soon as the *DataWriter* becomes alive once again to the *DataReader*.

[RTI Issue ID SEC-895]

4.2.13 Communication failure when using MultiChannel and changing filter expression parameters

A *DataReader* failed to receive samples from a *DataWriter* in the following scenario:

- The *DataWriter* set **metadata_protection_kind** or **data_protection_kind** to a value other than NONE.
- The *DataWriter* used the MultiChannelQosPolicy to create channels 1 and 2.
- The *DataReader* used the ContentFilteredTopic API to subscribe to channel 1.

- After some time, the *DataReader* changed its filter expression parameters to subscribe to channel 2.
- The *DataWriter* wrote a sample that was sent to channel 2.
- The *DataReader* failed to receive this sample and generated this log message at the ENTITIES category and the STATUS_REMOTE verbosity:

```
key material not yet received. Dropping data...
```

This problem has been resolved. The *DataReader* will now receive the sample.

[RTI Issue ID SEC-896]

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** > 65535. 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]

6 Third-Party Licenses