RTI Security Plugins Release Notes

Version 7.2.0

Your systems.
Working as one.

Contents

Chapter 1

Copyrights and Notices

Trademarks

RTI, Real-Time Innovations, Connext, Connext Drive, NDDS, the RTI logo, 1RTI and the phrase, "Your Systems. Working as one." are registered trademarks, trademarks or service marks of Real-Time Innovations, Inc. All other trademarks belong to their respective owners.

Copy and Use Restrictions

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form (including electronic, mechanical, photocopy, and facsimile) without the prior written permission of Real-Time Innovations, Inc. The software described in this document is furnished solely under and subject to RTI's standard terms and conditions available at https://www.rti.com/terms and in accordance with your License Acknowledgement Certificate (LAC) and Maintenance and Support Certificate (MSC), except to the extent otherwise accepted in writing by a corporate officer of RTI.

Securing a distributed, embedded system is an exercise in user risk management. RTI expressly disclaims all security guarantees and/or warranties based on the names of its products, including Connext Secure, RTI Security Plugins, and RTI Security Plugins SDK. Visit https://www.rti.com/terms/ for complete product terms and an exclusive list of product warranties.

Third-Party Software

RTI software may contain independent, third-party software or code that are subject to third-party license terms and conditions, including open source license terms and conditions. Copies of applicable third-party licenses and notices are located at community.rti.com/documentation. IT IS YOUR RESPONSIBILITY TO ENSURE THAT YOUR USE OF THIRD-PARTY SOFTWARE COMPLIES WITH THE CORRESPONDING THIRD-PARTY LICENSE TERMS AND CONDITIONS.

Notices

Deprecations and Removals

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

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

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/

Chapter 2

Supported Platforms

RTI® Security Plugins 7.2.0 is a feature release based on release 7.1.0.

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

Chapter 3

Compatibility

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

The *Security Plugins* 7.2.0 are interoperable with the *Security Plugins* 5.2.7 and higher.

Persistence Service databases secured with the *Security Plugins* 7.2.0 are incompatible with databases generated by versions of *Persistence Service* older than 7.0.0.

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

In release 7.2.0, the *Security Plugins* are available for use with OpenSSL® 1.1.1, OpenSSL 3.0, and wolfSSL® 5.5. There are separate installation packages for each of these options.

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

3.1 Compatibility with OpenSSL 1.1.1 and 3.0

The *Security Plugins* 7.2.0 for OpenSSL are API-compatible with OpenSSL 1.1.0 - 1.1.1t, and OpenSSL 3.0. The *Security Plugins 7.1.0* have only been tested by RTI using OpenSSL 1.1.1t and 3.0.9.

The *Security Plugins* SDK has been tested with OpenSSL 1.1.1t and 3.0.9.

Limitations when using OpenSSL

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

3.2 Compatibility with wolfSSL 5.5

The *Security Plugins* 7.2.0 for wolfSSL have been tested with wolfSSL 5.5.1 on following target platform:

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

Limitations when using wolfSSL

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

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

Chapter 4

What's New in 7.2.0

This section describes what's new, compared to the *RTI Security Plugins* 7.1.0.

This section includes descriptions of products, features, and platforms that are *deprecated* or *removed* starting in release 7.2.0.

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

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

4.1 Changes Related to Interoperability

4.1.1 Support for Connext systems running beyond 2038

RTI Connext applications have added support for systems running beyond the year 2038. This update, which is compliant with the latest OMG Real-Time Publish-Subscribe (RTPS) specification, version 2.5 (and was introduced in version 2.3), now makes it possible to run *Connext* applications up to year 2106. This update applies only to this and future releases of *Connext*; there are currently no plans to backport it to previous releases.

4.1.2 Connext Secure integrates with non-compliant DDS Security vendors

This release enables interoperability with DDS Security vendors that use a non-compliant Identity Certificate serialization format as part of the authentication handshake. When this situation is detected, the *Security Plugins* will fall back to a compatibility mode that allows the authentication process to continue as usual. In addition, the following warning message is logged:

```
identity certificate binary property contains a malformed certificate.
In particular, the certificate is not properly null terminated.
This is likely caused by a non-compliant DDS Security implementation.
```

4.1.3 Constrained devices running Lightweight Security can now integrate with devices running Security Plugins

The *Lightweight Security Plugins* are now interoperable with the *Security Plugins* in some configurations. For details, see the Lightweight Security Plugins and Security Plugins Interoperability in the Security Plugins User's Manual.

4.2 Changes Related to Access Control

4.2.1 Improved endpoint discovery time by removing redundant calling of check_remote_topic

The DDS Security specification describes the Access Control plugin operation `check_remote_topic()`. This function was implemented and invoked during endpoint discovery starting from *Security Plugins* 6.0.0. However, due to the absence of `TopicBuiltinTopicData` propagation, this invocation did not serve an effective security purpose and was redundant due to the operations `check_remote_datawriter()` and `check_remote_datareader()`. Moreover, the presence of `check_remote_topic()` made custom plugins code more error prone and potentially led to vulnerabilities when incorrectly used as a partial or complete replacement of `check_remote_datawriter()` or `check_remote_datareader()`.

For these reasons, the *Security Plugins* no longer call `check_remote_topic()`. See the *Migration Guide* on the RTI Community Portal for migration issues related to this removal.

4.2.2 Security Plugins for wolfSSL now support key usage extensions

The *Security Plugins* for wolfSSL now support enforcing the presence of the keyUsage X.509 v3 extension. In previous releases, this feature was supported only in the *Security Plugins* for OpenSSL.

You can now configure the *Security Plugins* for wolfSSL to only accept certificates including X.509 v3 key usage extensions. For more information, see the `authentication.x509v3_extension_enforcement.key_usage` property in Table 4.2 RTI Security Plugins Properties for Configuring Authentication of the *Security Plugins User's Manual*.

4.2.3 Secured communications between RTI Monitoring Library 2.0 and RTI Observability Collector Service

The telemetry data published by *RTI Monitoring Library 2.0* (previously known as *RTI Observability Library*) might contain sensitive information about your *RTI Connext* applications (for example, logging messages). Sensitive information must be protected from unauthorized access.

Starting with *Connext* 7.2.0, *RTI Security Plugins* can be enabled for both *RTI Monitoring Library 2.0* and *RTI Observability Collector Service*. DDS traffic containing metrics and logs can be encrypted or signed, and subscription permissions to telemetry data can be configured.

Security Plugins can be enabled for *Monitoring Library 2.0* through XML using the monitoring.distribution_settings.dedicated_participant.participant_qos_profile_name tag with a QoS profile that asserts the security artifacts. The *Observability Collector Service* Docker image also provides built-in configurations for enabling the *Security Plugins* either if it is deployed using RTI's prepackaged Docker Compose or as a separate Docker deployment.

New tags in the Governance and Permissions documents provide an easy and flexible way of configuring security for the *Observability Framework* topics and entities:

- The monitoring_metrics_protection_kind and monitoring_logging_protection_kind tags of the Governance document determine the level of protection applied to metrics and logs, respectively:

```
<dds>
    <domain_access_rules>
        <domain_rule>
            ...
            <monitoring_metrics_protection_kind>SIGN</monitoring_metrics_
↪protection_kind>
            <monitoring_logging_protection_kind>ENCRYPT</monitoring_logging_
↪protection_kind>
            ...
        </domain_rule>
    </domain_access_rules>
</dds>
```

- The subscribe_monitoring tag of the Permissions document determines the telemetry data *Observability Collector Service* is allowed to subscribe (metrics and logs, just metrics, or nothing). Publishing telemetry data is always allowed:

```
<dds>
    <permissions>
        <grant name="Participant_Monitoring">
            <subject_name>...</subject_name>
            <validity>...</validity>
            <allow_rule>
                ...
                <subscribe_monitoring>ALL</subscribe_monitoring>
            </allow_rule>
            <default>DENY</default>
        </grant>
    </permissions>
</dds>
```

For additional information, see Security in the *Observability Framework User's Manual*.

4.3 Changes Related to Cryptographic Algorithms

4.3.1 Added Pre-Shared Key Protection to Cloud Discovery Service and Real-Time WAN Transport

In *Connext* 7.1.0 we introduced Pre-Shared Key (PSK) Protection as a new protection mechanism, complementary to more-advanced *Security Plugins* features or standalone. In this release, we added PSK support for *Cloud Discovery Service* (protecting discovery information relayed by CDS) and *Real-Time WAN transport* (protecting UDP Binding Ping).

4.4 Changes Related to Cryptography

4.4.1 Added unique identifier to pre-shared key property

The *Security Plugins* now expect a different format for the `cryptography.rtps_protection_preshared_key property`. In previous releases, the property value was directly `<SEED>`, where `<SEED>` was the secret seed that the *Security Plugins* use to derive (in combination with other publicly available data) the per-participant pre-shared key. Starting in this release, the property value has to be `str:<ID>:<SEED>`, where `<ID>` is a number between 0 and 254 that uniquely identifies the `<SEED>`. RTPS messages that are protected using a pre-shared key have this `<ID>` associated with them. This allows *DomainParticipants* to compare the pre-shared key used to protect the incoming message with their local pre-shared key. If the identifiers do not match, the local *DomainParticipant* will drop the incoming RTPS message.

4.4.2 Added mutability to pre-shared key seed

Starting in this release, you can modify the value of the `cryptography.rtps_protection_preshared_key` property at runtime. Mutability of the pre-shared key seed allows you to update the security of your system without having to re-create the affected *DomainParticipants*. *DomainParticipants* that have the updated property value will generate a new local pre-shared key and protect their RTPS messages with it. When these *DomainParticipants* receive a RTPS message protected with a pre-shared key, they will update the key associated to the remote *DomainParticipant*.

4.5 Changes Related to Discovery and Authentication

4.5.1 Enabled configuration of protection kind for the builtin service request channel

Previously, the protection kind of the builtin service request channel was inferred from the discovery protection kind configured in the governance file. A new domain-level rule has been added to the governance file that allows the protection kind of the service request channel to be explicitly configured: `service_request_protection_kind`. If `service_request_protection_kind` is not set in the governance file, the protection kind is inherited from `discovery_protection_kind`. When the protection kind is inherited from discovery, the `service_request_protection_kind` is not propagated during discovery.

4.5.2 Support retrieving subject name of a remote DomainParticipant that is not authenticated yet

This release introduces support for retrieving the subject name of a remote *DomainParticipant* that is not authenticated yet. This feature allows you to make dynamic permissions decisions based on the subject name. You can do so using the `discovered_participant_subject_name`, `ignore_participant`, and `banish_ignored_participants` APIs in the `on_data_available` callback of the `DDS_ParticipantBuiltinTopicData`'s *DataReader*.

This feature is possible because *DomainParticipants* can now propagate their Identity Certificate's Subject Name during discovery. They will only propagate their Identity Certificate's Subject Name if the value of the `authentication.enable_discovery_subject_name_propagation` property is TRUE. The advantage of setting the property to TRUE is that it allows a remote *DomainParticipant* to get the Subject Name of the Identity Certificate before completing authentication.

If the property value is FALSE (default value), the local *DomainParticipant* won't propagate the Subject Name of its Identity Certificate during discovery. This configuration reduces discovery overhead. The disadvantage is that if a remote *DomainParticipant* calls `discovered_participant_subject_name` before authenticating the local *DomainParticipant*, this function will return `DDS_RETCODE_NO_DATA`.

4.5.3 SPDP2 participant announcements now subject to sample signature verification

TrustedState is a *Security Plugins* mechanism that ensures that the contents of a participant discovery message are legitimate. TrustedState has previously only been supported with Simple Particpant Discovery (SPDP) participants. Starting with this release, TrustedState is supported in Simple Participant Discovery 2.0 (SPDP2) participants, too.

See Simple Participant Discovery 2.0, in the RTI Connext Core Libraries User's Manual for more information about SPDP2.

4.5.4 Re-validation of remote participant data after authorization

A *DomainParticipant* will now re-validate a change in the remote participant data even after initial authorization completes. If a field in the remote participant data changes to a value that violates the permissions document, the remote participant will be removed. This enhancement applies to both SPDP and SPDP2 participants.

See Simple Participant Discovery 2.0, in the RTI Connext Core Libraries User's Manual for more information about SPDP2.

4.5.5 Unauthorized remote participants now removed instead of ignored, allowing for re-authorization

Previously, if a remote *DomainParticipant* failed authorization it would be ignored, meaning that authorization would not be attempted again, even if the remote participant changed its properties. Because mutable fields can now be re-validated after authorization, authorization failures no longer ignore a remote participant, but simply remove it. If a participant fails authorization because it has an unallowed partition, but it then changes to an allowed partition, the *Security Plugins* now re-perform authentication and authorization, and ultimately authorize the participant. This change applies to both SPDP and SPDP2 participants.

See Simple Participant Discovery 2.0, in the RTI Connext Core Libraries User's Manual for more information about SPDP2.

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

4.6.1 Dynamically control access to your Connext Secure system using a whitelist of trusted subject names

A new *DomainParticipant* PropertyQos property, `dds.participant.trust_plugins.subject_name_whitelist`, enables you to dynamically control system access using a whitelist. `dds.participant.trust_plugins.subject_name_whitelist` configures a whitelist of subject names for authenticated *DomainParticipants*. If set (even if set to an empty string), an authenticated *DomainParticipant* is allowed into the system only if its subject name matches one in the whitelist. Any authenticated *DomainParticipant* whose subject name does not match the whitelist will be ignored automatically.

This property does not affect allowed, non-authenticated participants; the whitelist is enforced only on authenticated *DomainParticipants*. If the list is modified after a *DomainParticipant* is enabled, any *DomainParticipant* that was previously ignored will be unignored. This creates an opportunity to successfully authenticate if the *DomainParticipant* subject name is in the updated whitelist; if not, the *DomainParticipant* will be ignored as usual.

4.6.2 Added configuration option for certificate expiration notice frequency

In 7.1.0, according to Dynamic Certificate Expiration of the Local DomainParticipant, if you implemented the on_invalid_local_identity_status_advance_notice callback function and the certificate was going to expire within the advance notice duration, then the *Security Plugins* would notify you every second that your certificate was about to expire. The frequency of this notification is now configurable using the property `dds.participant.trust_plugins.certificate_expiration_advance_notice_reminder_period.sec`. For more information, see 4. Authentication in the RTI Security Plugins User's Manual 7.2.0 documentation.

4.6.3 Increased robustness against DomainParticipants leaving the system before their certificates become expired or revoked

As described in Dynamic Certificate Expiration of Remote DomainParticipants, in the RTI Security Plugins User's Manual 7.1.0, the *Security Plugins* will automatically create a new Key Revision in order to render a *DomainParticipant*'s Key Material outdated when the *DomainParticipant*'s certificate expires. In previous versions, this mechanism worked when the expiration happened **before** the *DomainParticipant* leaves the system, but it did not work when the expiration happened **after** the *DomainParticipant* left the system. So if Participant A lost liveliness of Participant B, and then Participant B's certificate expired, then Participant B would still be able to decrypt messages that Participant A was sending to Participant C if Participant B was able to intercept those messages.

In this release, the *Security Plugins* close this loophole by automatically creating a new Key Revision whenever a previously-alive *DomainParticipant*'s certificate becomes expired or revoked. The number of previously-alive *DomainParticipants* to keep track of is configurable using the new property `dds.participant.trust_plugins.max_removed_participants_per_key_revision`. For more information, see Properties for Configuring Cryptography Affecting Any Cryptography Plugin, in the RTI Security Plugins User's Manual.

4.6.4 Improved debuggability, usability, and forward compatibility of Key Revisions

As described in Crypto Header, in the 7.0.0 RTI Security Plugins User's Manual, the interpretation of four of the bytes in the Crypto Header depended on whether or not Key Revisions were enabled (using the `dds.participant.trust_plugins.key_revision_max_history_depth` property). This ambiguous behavior hindered debuggability; for example, it prevented Wireshark from accurately dissecting the Crypto Header.

This release changes that behavior. Now, the interpretation of those four bytes is always the same, regardless of the value of the `dds.participant.trust_plugins.key_revision_max_history_depth` property. This change impacts backward compatibility with 7.1.0, as described in the Migration Guide, but it improves forward compatibility with future releases. In addition, this behavior improves the usability of RTPS PSK Protection because the *Security Plugins* can now easily distinguish between a mismatch of preshared key algorithms and a mismatch of preshared key values.

See Crypto Header, in the 7.2.0 RTI Security Plugins User's Manual for more information.

4.6.5 Allowed Identity Certificate to be mutable

You may now dynamically change your Identity Certificate without having to restart your *DomainParticipant*. The Identity Certificate is now mutable in two ways:

- Changing the value of the `dds.sec.auth.identity_certificate` property using the *DomainParticipant* `set_qos` API. This method works regardless of whether the old or new value of the property has the `data:,` or `file:` prefix.
- Leaving the value of the `dds.sec.auth.identity_certificate` property unchanged and instead changing the contents of the actual certificate file. The *Security Plug-*

ins enforce these changes as long as the com.rti.serv.secure.authentication.identity_certificate_file_poll_period.millisec is set to a value other than 0 (0 is the default). This method works only when the value of the dds.sec.auth.identity_certificate property has the file:, prefix.

As soon as an Identity Certificate change is detected, the Security Plugins will propagate the new certificate to all trusted remote *DomainParticipants* so that communication with them will not be interrupted when the old certificate expires.

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

4.6.6 Allowed Certificate Revocation List to be mutable

You may now dynamically revoke new *DomainParticipants* without having to restart your *DomainParticipant*. The certificate revocation list is now mutable in two ways:

- Changing the value of the authentication.crl property using the *DomainParticipant* set_qos API. This method works regardless of whether the old or new value of the property has the data:, or file: prefix.
- Leaving the value of the authentication.crl property unchanged and instead changing the contents of the actual CRL file. The *Security Plugins* enforce these changes as long as the authentication.crl_file_poll_period.millisec is set to a value other than 0 (0 is the default). This method works only when the value of the authentication.crl property has the file: prefix.

As soon as a CRL change is detected, the *Security Plugins* will remove any newly revoked remote *DomainParticipants*.

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

4.6.7 New example for dynamic certificate revocation and renewal

There is now a C example that demonstrates dynamic certificate revocation and renewal. You can find it in <path to examples>/connext_dds/c/hello_dynamic_certificates. For more information, see Advanced Authentication Concepts, in the *RTI Security Plugins User's Manual*.

4.7 Changes Related to Platforms and Builds

4.7.1 Support for Security Plugins for wolfSSL 5.5.1 on certain Linux platforms

Previously, the *Security Plugins* for wolfSSL were only supported in the armv8QNX7.1qcc_gpp8.3.0 target architecture.

This release of the *Security Plugins* introduces support for wolfSSL 5.5.1 on these platforms when using x64 CPUs:

- Red Hat Enterprise Linux 8.0 and 9.0 systems on x64 CPUs (RTI architecture: x64Linux4gcc7.3.0)
- Ubuntu 18.04 LTS, 20.04 LTS, and 22.04 LTS systems on x64 CPUs (RTI architecture: x64Linux4gcc7.3.0)

The RTI architecture for these platforms is `x64Linux4gcc7.3.0`.

See the instructions in the Security Plugins Installation Guide to get started with the *Security Plugins* for wolfSSL.

4.8 Changes Related to Security Plugins SDK

4.8.1 Removed dependency on CMocka third-party library

Starting with this release, the *Security Plugins* SDK no longer depends on the CMocka third-party library for building and running the test suite. The *Security Plugins* SDK now uses a test library ("rtitest") shipped with *Connext*. Users of the *Security Plugins* SDK don't have to install the separate CMocka bundle anymore. The CMake recipe in the SDK imports the test library dependency automatically.

4.8.2 Added support for building and testing Lightweight Security library

In previous releases, the result of building the *Security Plugins* SDK was the full *Security Plugins* library. Starting in this release, users of the *Security Plugins* SDK get an additional library: the *Lightweight Security* library. This release also introduces a tester for the *Lightweight Security* library.

4.8.3 Improved logging during clean up of Security Plugins

The *Security Plugins* did not log errors if they encountered an issue during clean up (for example, errors when freeing resources). This issue has been resolved. The *Security Plugins* now propagate errors found during clean up and they log the proper error messages.

4.9 Changes Related to System Extensibility and Configurability

4.9.1 Properties that could increase system vulnerability have been removed

The following properties have been removed since their usage could make the system more vulnerable to attackers:

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

Connext 7.1.0 already deprecated these properties. See *Deprecated properties related to the propagation of security info* for more information about it.

4.10 Changes Related to Third-Party Software

4.10.1 Upgraded OpenSSL to version 3.0.9 and removed OpenSSL 1.1.1 support

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

Third-party Tool	Old Versions	New Version
OpenSSL	1.1.1t, 3.0.8	3.0.9

In this release, the *Security Plugins* are only available as a set of **nddssecurity** libraries built against OpenSSL 3.0.9 (which is supported until September, 2026). The support of OpenSSL 1.1.1 has been removed, because it is end-of-life in September, 2023.

Chapter 5

What's Fixed in 7.2.0

5.1 Fixes Related to Security Plugins SDK

5.1.1 Warning when statically building the Security Plugins SDK on macOS systems

The *Security Plugins SDK* used to warn about the crypto library adapters having no symbols when building statically on MacOS systems:

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

These warnings occurred because the SDK was linking against all the crypto library adapter files, including files that may be empty, depending on preprocessor macros. The warnings were not harmful, since the SDK does not use the empty files mentioned in the warnings.

This issue has been resolved. Now, the *Security Plugins SDK* only links against the crypto library adapter files that match your chosen crypto library.

[RTI Issue ID SEC-1984]

5.2 Fixes Related to Crashes

5.2.1 Segmentation fault when receiving corrupted handshake message with zero-length certificate

If the identity certificate in a corrupted authentication handshake message had zero length, the receiving *DomainParticipant* would experience a segmentation fault. This problem has been fixed. Now, the *DomainParticipant* will not experience a segmentation fault, and will print this error:

```
failed to get reference to the last character of the identity certificate␣
↪because the identity certificate supposedly has zero length
```

[RTI Issue ID SEC-2227]

5.3 Fixes Related to Cryptography

5.3.1 Incorrect processing of endpoint CryptoTokens or precondition failure when destination participant was incorrect

A message on the ParticipantVolatileMessageSecure topic (see the Cryptography section in the *RTI Security Plugins User's Manual*) includes the GUID of the *DomainParticipant* that is the intended recipient of the message. After the actual recipient successfully decrypts such a message, the recipient must verify that the intended recipient is the actual recipient.

If the message included the Key Material of a *DataWriter* or *DataReader*, then this verification was only done in the debug libraries; then, if the verification failed, an error displayed regarding the internal function `PRESPsService_processEndpointCryptoTokens` and mentioning `!precondition`. Since the verification was only done in the debug libraries, it was possible for release libraries to accept *DataWriter* or *DataReader* Key Material from a DDS Security implementation that did not populate the `ParticipantVolatileMessageSecure` topic correctly.

[RTI Issue ID SEC-1954]

5.3.2 Lack of origin authentication led to unnecessary allocation and possible discovery failure

When the property `cryptography.max_receiver_specific_macs` was unset or set to 0, there was an unnecessary memory allocation related to receiver-specific MACs whenever creating or discovering an entity. In some cases, the cryptographic library may have failed to make this allocation, in which case entity creation or discovery would have failed with this error message:

```
RTI_Security_CryptoLibAdapterEvpNewMacKey (MasterReceiverSpecificKey) failed␣
↪with error
```

This problem only affected versions 6.0.1.29 to 6.0.1.33, versions 6.1.1 to 6.1.2.11, and versions 7.0.0 to 7.1.0. This problem has been fixed. The *Security Plugins* no longer attempt to make this allocation if origin authentication is not used.

[RTI Issue ID SEC-2210]

5.4 Fixes Related to Access Control

5.4.1 Unexpected error when Permissions Document is configuring certain not_before/not_after dates

When the Permissions Document contained a not_before/not_after date in the interval `2038-01-19T02:00:00` to `2038-01-19T03:00:00` in combination with a timezone in minutes, an unexpected error ("`dateTime is before the unix epoch (1970-01-01T00:00:00Z)`") may have triggered, causing the Permissions Document parsing to fail.

This issue has been fixed; configuring a not_before/not_after date in the specified interval no longer triggers an error.

[RTI Issue ID SEC-2035]

5.5 Fixes Related to Interoperability

5.5.1 Security PIDs did not comply with OMG DDS Security standard

Connext 7.0.0 added four security-related PIDs to aid in *DomainParticipant* discovery, matching, and early detection of security configuration issues. These PIDs were erroneously implemented and caused a conflict with the OMG DDS Security standard, specifically IDENTITY_STATUS_TOKEN (0x1006). *RTI Security Plugins* interoperability with other vendors was also negatively affected. This issue was fixed by moving all of the affected PIDs to positions defined in the OMG DDS Security standard. See the tables below for affected PIDs and their values (notated as “old value –> new value”).

Table 5.1: Participant Discovery PIDs

digital_signature	ParticipantSecurityDigitalSignatureAlgorithms (see 7.2.9)	PID_PARTICIPANT_SECURITY_DIGITAL_SIGNATURE_ALGO	0x1006 –> 0x1010
key_establishment	ParticipantSecurityKeyEstablishmentAlgorithms (see 7.2.9)	PID_PARTICIPANT_SECURITY_KEY_ESTABLISHMENT_ALGO	0x1007 –> 0x1011
symmetric_cipher	ParticipantSecuritySymmetricCipherAlgorithms (see 7.2.9)	PID_PARTICIPANT_SECURITY_SYMMETRIC_CIPHER_ALGO	0x1008 –> 0x1012

Table 5.2: Endpoint Discovery PIDs

symmetric_cipher	EndpointSecuritySymmetricCipherAlgorithms (see 7.2.10)	PID_ENDPOINT_SECURITY_SYMMETRIC_CIPHER_ALGO	0x1009 –> 0x1013

[RTI Issue ID SEC-2071]

5.5.2 Placement of GUID within RTPS message incorrectly affected vendor interoperability

In previous releases, the *Security Plugins* expected the PID_PARTICIPANT_GUID to be serialized in the RTPS message before any other field and failed whenever the PID_PARTICIPANT_GUID was preceded with a different field. This negatively affected interoperability with other vendors. This issue has been fixed. Now the PID_PARTICIPANT_GUID can be serialized in any place within the message.

[RTI Issue ID SEC-1717]

5.6 Fixes Related to Dynamic Participant Renewal, Revocation, and Expiration

5.6.1 Segmentation fault after banish_ignored_participants if the participant had a disabled writer

Calling `banish_ignored_participants` led to a segmentation fault if the *DomainParticipant* had a disabled *DataWriter*, either due to creating a *DataWriter* from a Publisher with `PublisherQos.entity_factory.autoenable_created_entities` set to false, or due to creating a *DataWriter* that was in the middle of being enabled. With debug libraries, you would have gotten this error:

```
!precondition: "me == ((void *)0)"
```

This problem only affected *Security Plugins* 7.0.0 and above and has been fixed.

[RTI Issue ID SEC-2190]

5.6.2 Using a preshared key and calling banish_ignored_participants led to decoding failures

When using Pre-Shared Key Protection with the Security Plugins, a *DomainParticipant* that called `banish_ignored_participants` sent messages protected by the pre-shared key that the receiver failed to decode. The receiver would log this error:

```
EVP_DecryptFinal_ex failed with error: (error details not available)
```

This problem has been fixed. Pre-Shared Key Protection is now compatible with `banish_ignored_participants`.

[RTI Issue ID SEC-2176]

5.6.3 Security Plugins for wolfSSL may not have invoked the on_invalid_local_identity_status_advance_notice callback at the right time

The `on_invalid_local_identity_status_advance_notice` callback is invoked when the local *DomainParticipant's* Identity Certificate has already expired or will expire within the duration specified by the `dds.participant.trust_plugins.certificate_expiration_advance_notice_duration.sec` property.

In version 7.1.0, the *Security Plugins* for wolfSSL may not have invoked this callback at the right time due to a bug in wolfSSL's `ASN1_TIME_to_tm` API. You can find more information in wolfSSL's GitHub repository, issue #6387. As a result, if the local time had an offset with respect to GMT or Daylight Saving Time was in effect, neither were considered when calculating the time to trigger the callback. If Daylight Saving Time was in effect, the callback would be triggered 1 hour later than expected. An offset with respect to GMT would also imply that the Security Plugins for wolfSSL would invoke `on_invalid_local_identity_status_advance_notice` early (if the offset was positive), or late (if the offset was negative).

The *Security Plugins* currently requires a version of wolfSSL that presents this bug (5.5.1). The issue has been addressed using a workaround in the *Security Plugins* for wolfSSL, which now avoids using the `ASN1_TIME_to_tm` API.

[RTI Issue ID SEC-2072]

5.6.4 Intraparticipant communication crashed when using banish_ignored_participants

If the Governance Document tag `<rtps_protection_kind>` was set to a value other than `NONE`, a race condition may have led to a hang or crash when using a *DataWriter* to communicate with a *DataReader* on the same *DomainParticipant* and when calling the API `banish_ignored_participants`. This problem only affected *Security Plugins* 7.1.0 and has been fixed.

[RTI Issue ID SEC-2082]

5.7 Fixes Related to Usability

5.7.1 Lightweight Security Library and Security Plugins Library could not be simultaneously loaded into the same application

Previously, trying to simultaneously load both the *Lightweight Security Plugins* library (`nddslightweightsecurity`) and the *Security Plugins* library (`nddssecurity`) within the same application may have triggered linking errors. This configuration is now fully supported. For details on how to load the *Lightweight Security Plugins* in your application, see Configuring the Lightweight Security Plugins in the Security Plugins User's Manual.

[RTI Issue ID SEC-2077]

5.7.2 Disabling TypeObject caused a precondition failure in debug libraries

`serialized_type_object_dynamic_allocation_threshold` was not properly adjusted when disabling TypeObject, causing a precondition to fail when using debug libraries. This issue did not cause any errors with release libraries and simply allocated more memory than needed. This has now been fixed; disabling TypeObject no longer causes a precondition failure with debug libraries.

[RTI Issue ID SEC-1815]

5.8 Fixes Related to XML Configuration

5.8.1 Governance Document XML schema definition had a syntax error

The Governance Document XSD (`dds_security_governance.xsd`) had a syntax error in release 7.1.0. A forward slash was missing at the end of the `rtps_preshared_secret_protection_kind` element definition. Instead of:

```
<xs:element name="rtps_preshared_secret_protection_kind" type=
↪"BasicProtectionKind" >
```

It should be:

```
<xs:element name="rtps_preshared_secret_protection_kind" type=
↪"BasicProtectionKind" />
```

This issue has been fixed.

[RTI Issue ID SEC-2090]

5.9 Fixes Related to Discovery and Authentication

5.9.1 Could not create multiple participants in the same application when using OpenSSL engine for private key

This issue was fixed in release 7.1.0, but not documented at that time.

When using the `openssl_engine` property and setting the `authentication.keyform` property to `engine`, you could not create multiple *DomainParticipants* using the same engine on the same application. You would get an error mentioning `RTI_Security_CertHelper_loadPrivateKey` and `cannot load ENGINE keyform: OpenSSL engine not defined`. This problem has been fixed. Creating multiple *DomainParticipants* now succeeds in this scenario.

[RTI Issue ID SEC-2103]

5.9.2 Discovery time scaled poorly

Endpoint discovery time scaled poorly as the number of endpoints increased. Moreover, when using HMAC-Only mode or the *Lightweight Security Plugins*, participant discovery time incorrectly did not scale as the number of participants increased. These problems only affected the *Security Plugins* 6.0.0 and above and has been fixed. The discovery time is now comparable with that of *Security Plugins* 5.3.1.

[RTI Issue ID SEC-2170]

5.9.3 Security Plugins for wolfSSL incorrectly tried to verify a revoked Identity Certificate against all Certificate Authorities

DomainParticipants using an Identity Certificate included in a signed (by the Identity Certificate's issuer) Certificate Revocation List should not be created; the issuer revoked the Identity Certificate, and it is no longer valid. Therefore, the certificate does not need to be verified against the alternative Identity Certificate Authorities.

Previously, the *Security Plugins* for wolfSSL did try to verify the certificate against all the Certificate Authorities. As a result, the *Security Plugins* logged the revocation error message `error -361: CRL Cert revoked` once for each of the Certificate Authorities.

This issue has been fixed. The *Security Plugins* for wolfSSL now detect if an Identity Certificate is revoked when verifying it against the main CA, and will fail without continuing further validation.

[RTI Issue ID SEC-2076]

5.10 Fixes Related to Shipped Examples

5.10.1 hello_banish example XML file had XSD validation errors

The hello_banish example USER_QOS_PROFILES.xml had a `DDS_` prefix for reliability and durability values, which triggered XSD validation errors. This problem has been fixed by removing the `DDS_` prefix.

[RTI Issue ID SEC-2241]

5.11 Fixes Related to Vulnerabilities

5.11.1 Potential Denial of Service when using OpenSSL 3.0 due to a vulnerability in OpenSSL 3.0

The *Security Plugins* had a third-party dependency on OpenSSL 3.0, which is known to be affected by a number of publicly disclosed vulnerabilities.

These vulnerabilities have been fixed by upgrading OpenSSL to the latest stable version, 3.0.9. See *Changes Related to Third-Party Software* for more details.

User Impact without Security

No impact.

User Impact with Security

The impact on *Security Plugins* applications of using the previous version was as follows:

- Exploitable by triggering the parsing of malicious Permissions Documents, even when they were not properly signed by a CA.
- The application could have experienced notable to very long delays.
- CVSS Base Score: 7.5 HIGH
- CVSS v3.1 Vector: AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:H

[RTI Issue ID SEC-2100]

5.12 Other Changes

5.12.1 Confidential property not listed in Release Notes

In release 7.1.0, the new property `rtps_protection_preshared_key` was documented in the Security Plugins User's Manual but not included in a list of sensitive properties in *Redaction of sensitive properties when logging DDS 'Entities' PropertyQos configuration*. This release includes it in the list of sensitive properties found in *Redaction of sensitive properties when logging DDS 'Entities' PropertyQos configuration*.

[RTI Issue ID SEC-2049]

Chapter 6

Previous Releases

6.1 What's New in 7.1.0

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

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

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

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

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

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

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

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

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

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

6.1.2 Changes Related to Cryptography

Pre-Shared Key-based RTPS protection mechanism

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

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

RTPS PSK Protection can be leveraged in two different ways:

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

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

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

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

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

6.1.3 Changes Related to Performance and Scalability

RTI Lightweight Security

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

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

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

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

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

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

6.1.4 Changes Related to APIs

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

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

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

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

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

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

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

For more information, see:

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

6.1.5 Changes Related to Usability

6.1.6 Changes Related to Debuggability

Adjusted verbosity of several security event logged messages

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

- DDS_LOGGING_EMERGENCY_LEVEL: Used to log fatal error conditions that prevent RTI Security Plugins from continuing to run properly.
- DDS_LOGGING_ALERT_LEVEL: Used to log security alerts. Usually derived from a remote peer not being properly configured or being malicious.
- DDS_LOGGING_CRITICAL_LEVEL: Used to log critical, unexpected errors. In most cases, these errors will be triggered by the local host running out of resources. While the *Security Plugins* can continue operating, it is likely that new errors will continue to be triggered.
- DDS_LOGGING_ERROR_LEVEL: Used to log error conditions.
- Higher verbosity levels: Used to log non-error conditions, from warnings to informative messages.

6.1.7 Changes Related to Third-Party Software

Upgraded OpenSSL to versions 1.1.1t and 3.0.8

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

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

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

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

If you are using OpenSSL Engines (see Support for OpenSSL Engines, in the Security Plugins User's Manual), please note that the *Security Plugins* do not support providers (see https://www.openssl.org/docs/man3.0/man7/provider.html).

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

Upgraded to wolfSSL 5.5.1

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

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

6.1.8 Changes Related to System Extensibility and Configurability

Deprecated properties related to the propagation of security info

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

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

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

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

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

6.1.9 Changes Related to Supported Platforms

New Platforms

This release adds support for this platform:

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

Removed Platforms

The following platforms are no longer supported:

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

6.1.10 Changes Related to Shipped Examples

Support building shipped examples using different crypto libraries

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

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

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

Security examples now support secp384r1 curve

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

New example for banishing participants

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

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

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

6.1.11 Other Changes

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

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

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

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

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

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

6.2 What's Fixed in 7.1.0

6.2.1 Fixes Related to Discovery and Authentication

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

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

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

[RTI Issue ID SEC-1779]

Missing security information in the Participant Builtin Topic data

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

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

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

[RTI Issue ID SEC-1871]

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

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

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

[RTI Issue ID SEC-1883]

6.2.2 Fixes Related to Cryptography

Session keys renewed half as frequently as they should have been

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

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

The issue has been fixed.

[RTI Issue ID SEC-1231]

data_protection_kind = SIGN was sometimes treated as ENCRYPT

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

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

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

[RTI Issue ID SEC-1773]

Possible crash when disable_endpoint_security_info_propagation was true

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

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

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

[RTI Issue ID SEC-1747]

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

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

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

[RTI Issue ID SEC-1786]

Value AES192+GCM for cryptography.encryption_algorithm did not work

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

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

[RTI Issue ID SEC-1806]

Setting wrong value for symmetric cipher algorithm failed silently

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

[RTI Issue ID SEC-1807]

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

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

These problems have been resolved.

[RTI Issue ID SEC-1825]

Communication failure when using origin authentication and banish_ignored_participants

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

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

This problem has been resolved.

[RTI Issue ID SEC-1862]

Communication failure when using origin authentication and max_blocks_per_session

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

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

This problem has been resolved.

[RTI Issue ID SEC-1863]

Potential invalid read while decoding encrypted messages

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

[RTI Issue ID SEC-1892]

6.2.3 Fixes Related to Reliability Protocol and Wire Representation

Unexpected error 'Fragment data not supported by this writer'

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

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

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

[RTI Issue ID SEC-1813]

6.2.4 Fixes Related to Entities

Error creating participant with specific local GUID prefixes using security

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

[RTI Issue ID SEC-1835]

6.2.5 Fixes Related to Shipped Examples

DomainParticipantQoS in C hello_security was not finalized

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

[RTI Issue ID SEC-1699]

6.2.6 Fixes Related to Vulnerabilities

Submessage Protection was ineffective at protecting against submessage tampering

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

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

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

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

[RTI Issue ID SEC-1887]

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

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

User Impact without Security: No impact.

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

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

This issue has been fixed.

[RTI Issue ID SEC-1955 THIRDPARTY-70]

Authentication handshake did not effectively protect against GUID impersonation

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

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

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

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

[RTI Issue ID SEC-1988]

6.2.7 Fixes Related to Security Plugins SDK

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

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

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

[RTI Issue ID SEC-1884]

6.2.8 Fixes Related to Crashes

Potential crash while decoding protected submessages

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

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

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

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

[RTI Issue ID SEC-1960]

6.3 What's New in 7.0.0

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

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

- **Seamlessly Regenerate and Redistribute Key Material**

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

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

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

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

  The specific new features related to this are described in *Changes Related to Cryptographic Algorithms* and *Changes Related to System Extensibility and Configurability*.

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

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

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

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

Support for kicking Participants off a system

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

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

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

This feature introduces new properties:

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

To enable this feature, you must set the property **dds.participant.trust_plugins.key_revision_max_history_depth** to a non-zero value. A *DomainParticipant* that sets this property to a non-zero value will not communicate with a *DomainParticipant* that sets this property to 0, or with a *DomainParticipant* of a release older than *Security Plugins* 7.0.0.

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

New API for kicking Participants off a system

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

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

6.3.2 Changes Related to Cryptographic Algorithms

Support for secp384r1 key-establishment and digital-signature

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

Changes to property that configures key-establishment algorithm

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

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

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

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

Removed support for Digital Signature Algorithm (DSA)

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

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

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

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

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

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

Changed default symmetric cipher algorithm to AES256+GCM

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

6.3.3 Changes Related to System Extensibility and Configurability

Information about supported and used cryptographic algorithms propagated in discovery

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

DomainParticipants propagate the following information:

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

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

Compatibility Rules

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

- Identity trust chain digital signature algorithms
  - PA's supported algorithms intersect with any bit from PB's used algorithm, *and*
  - PB's supported algorithms intersect with any bit from PA's used algorithm.
- Authentication digital signature algorithms
  - PA's supported algorithms intersect with PB's used algorithm, *and*
  - PB's supported algorithms intersect with PA's used algorithm.
- Key establishment algorithms
  - PA's supported algorithms intersect with PB's preferred algorithm, *and*
  - PB's supported algorithms intersect with PA's preferred algorithm.
- Symmetric cipher algorithms
  - PA's supported algorithm intersects with PB's used algorithm, *and*
  - PB's supported algorithm intersects with PA's used algorithm, *and*
  - PA's builtin endpoint key exchange algorithm is equal to PB's builtin endpoint key exchange algorithm.
- Two endpoints, EPA and EPB, are compatible if:
  - PA's supported symmetric cipher algorithms intersect with EPB's used algorithm, and
  - PB's supported symmetric cipher algorithms intersect with EPA's used algorithm.

Ability to configure system-wide allowed security algorithms

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

- **<digital_signature>**

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

- - RSASSA-PSS-MGF1SHA256+2048+SHA256
  - RSASSA-PKCS1-V1_5+2048+SHA256
  - ECDSA+P256+SHA256
  - ECDSA+P384+SHA384
- **<digital_signature_identity_trust_chain>**

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

  Possible values:

  - RSASSA-PSS-MGF1SHA256+2048+SHA256
  - RSASSA-PKCS1-V1_5+2048+SHA256
  - ECDSA+P256+SHA256
  - ECDSA+P384+SHA384
- **<key_establishment>**

  Algorithms that *DomainParticipants* can use for key establishment.

  Possible values:

  - DHE+MODP-2048-256
  - ECDHE-CEUM+P256
  - ECDHE-CEUM+P384
- **<symmetric_cipher>**

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

Possible values:

- AES128+GCM
- AES256+GCM

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

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

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

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

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

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

Note: Using **must_interpret** in your Governance of Permissions Document *breaks compatibility* with versions of the *Security Plugins* before 7.0.0. For more information, see:

- The *Migration Guide* on the RTI Community Portal (https://community.rti.com/documentation),
- How the Governance Document is Interpreted, in the Security Plugins User's Manual, and
- How the XML is Validated, in the RTI Connext Core Libraries User's Manual.

6.3.4 Changes Related to Performance and Scalability

Improved throughput when batching protected data

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

- The combination of compression, batching, and data protection is now supported. First the batch will be compressed, then the compressed batch will be protected.

- The throughput of batching and data protection has been improved because the overhead of data protection only appears once per batch.

Added optional custom allocator for Security Plugins for OpenSSL

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

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

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

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

6.3.5 Changes Related to Usability

“file:” prefix is now optional when specifying filename properties

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

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

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

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

New APIs to identify DomainParticipants by subject name

When using the *Security Plugins*, it is natural to identify *DomainParticipants* by their Distinguished Names (subject names). Subject names appear in the Identity Certificate (see Identity Certificates, in the Security Plugins User's Manual), and the Permissions Document (see Permissions Document, in the Security Plugins User's Manual).

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

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

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

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

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

6.3.6 Changes Related to Debuggability

Improved message content in case of permissions validation failure

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

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

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

- The Logging Plugin will log a message if its log level is less than or equal to the verbosity of either the *Security Plugins* or the SECURITY category.
- *Connext* will log a security-related message if its log level is less than or equal to the SECURITY category verbosity.

- Setting the verbosity of the *Security Plugins* also configures the verbosity for the SECURITY category, which will affect any security-related message (including those logged from *Connext*) logged from any *DomainParticipant* within the same application.

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

Increased logging in case of identity validation failure

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

6.3.7 Changes Related to the Security Plugins SDK

New functions in SDK test infrastructure

There are several new functions you can use for testing:

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

New '-verbosity' argument for SDK testers

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

More meaningful return types for SDK tests

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

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

More return types may be added in the future.

6.3.8 Changes Related to Supported Platforms

New Platforms

This release adds support for these platforms:

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

Removed Platforms

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

- Android
- These Linux platforms:
  - CentOS 6.x
  - NI Linux 3
  - Red Hat Enterprise Linux 6.x

  - Ubuntu 18.04 LTS on Arm v7
- QNX Neutrino 6.x, 7.0.4
- VxWorks 7.x

6.4 What's Fixed in 7.0.0

6.4.1 Fixes Related to Discovery and Authentication

Reader incorrectly lost liveliness with writer when using enable_liveliness_protection

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

[RTI Issue ID SEC-1630]

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

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

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

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

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

[RTI Issue ID SEC-1676]

6.4.2 Fixes Related to Cryptography

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

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

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

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

[RTI Issue ID SEC-627]

6.4.3 Fixes Related to Access Control

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

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

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

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

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

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

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

[RTI Issue ID SEC-1687]

6.4.4 Fixes Related to Interoperability with Other Vendors

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

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

- The following warning would have incorrectly been logged when receiving a ParticipantBuiltinTopicData sample indicating a QoS change from any *DomainParticipant* beyond the fourth one:
- This warning was benign if it was logged upon receiving a ParticipantBuiltinTopicData sample from a *DomainParticipant* that was created using the *RTI Security Plugins*. But if the *DomainParticipant* was not created using a different implementation of DDS Security, then its QoS change would have gone undetected.

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

[RTI Issue ID SEC-1639]

Incorrect key agreement algorithm sent by replier DomainParticipant

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

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

[RTI Issue ID SEC-1674]

6.4.5 Fixes Related to Debuggability

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

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

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

[RTI Issue ID SEC-1640]

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

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

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

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

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

[RTI Issue ID SEC-1648]

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

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

[RTI Issue ID SEC-1653]

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

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

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

The right message was also logged:

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

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

[RTI Issue ID SEC-1710]

6.4.6 Fixes Related to the Security Plugins SDK

Certificate Revocation Lists expired after 30 days

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

[RTI Issue ID SEC-1677]

6.4.7 Fixes Related to Shipped Examples

Secure Hello World example always linked OpenSSL dynamically

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

[RTI Issue ID SEC-880]

Chapter 7

Known Issues

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

7.1 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
```

7.2 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]

7.3 subscription_data and publication_data in check_local_datawriter_match / check_local_datareader_match are not Populated

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

[RTI Issue ID SEC-758]

7.4 relay_only parameter in check_remote_datareader is not Populated

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

[RTI Issue ID SEC-852]

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

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

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

[RTI Issue ID SEC-1242]

7.6 Source and destination overlap in memcpy (called from wc_AesGcmInit) when using the Security Plugins for wolfSSL

Valgrind 3.15.0 (and lower versions) may detect an overlap in the source and destination memory when calling `memcpy` from `wc_AesGcmInit`. This is an issue in wolfSSL 5.5.1, not in the *Security Plugins*. The overlap happens if wolfSSL is compiled with `--enable-aesgcm-stream`. For more information, read wolfSSL's #6413 GitHub issue. This issue doesn't affect the behavior of the *Security Plugins* for wolfSSL.

[RTI Issue ID SEC-2087]

7.7 Segmentation fault when trying to enable pre-shared key protection at runtime

The *Security Plugins* will crash if you try to enable pre-shared key protection (setting the **cryptography.rtps_protection_preshared_key** property) in an already created *DomainParticipant* that has a Governance Document with **rtps_preshared_secret_protection_kind** equal to NONE. You will see a backtrace including the `RTI_Security_CryptographyPresharedKeyState_setIdAndString` function.

Enabling or disabling pre-shared key protection at runtime is not allowed.

[RTI Issue ID SEC-2247]