RTI Security Plugins Release Notes

Version 7.3.0

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 SECURITY PLUGINS *(RTI Security Plugins)* and SECURITY PLUGINS SDK *(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

See 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 1.2 Specification from the Object Management Group (OMG).

The Security Plugins 7.3.0 are interoperable with the *Security Plugins* 5.2.7 and higher.

Persistence Service databases secured with the Security Plugins 7.3.0 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 7.3.0, 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 3.0

The Security Plugins 7.3.0 for OpenSSL are API-compatible with OpenSSL 3.0. The Security Plugins 7.3.0 have only been tested by RTI using OpenSSL 3.0.12. OpenSSL 3.0.12 is not compatible with Security Plugins versions that were tested using OpenSSL 3.0.9 or below.

The Security Plugins SDK has been tested with OpenSSL 3.0.12.

3.2 Compatibility with wolfSSL 5.5

The Security Plugins 7.3.0 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/providers are not supported.
- If you use an unsupported certificate extension, you will get the "error details not available" message instead of OpenSSL's more debuggable "unhandled critical extension" message. See wolfSSL issue #6890 for more information about the problem and future fix.

Chapter 4

Key New Features in 7.3.0 LTS

Security Plugins 7.3.0 LTS is a long-term support release that is built upon and includes all the features in releases 7.0.0, 7.1.0, and 7.2.0 (see *Previous Releases*).

See the Connext Releases page on the RTI website for more information on RTI's software release model.

This section describes key new features and improvements in 7.3.0 LTS, compared to 6.1.2, the previous LTS release.

See also: *What's New in 7.3.0 LTS*.

Table 4.1: Key Features of the Security Plugins 7.3.0 LTS (Releases 7.0.0 - 7.3.0)

🔑	*Dynamic Certificate Revocation and Renewal* enables certificates to be dynamically revoked and renewed in operational systems.
🔐	*Pre-Shared Key (PSK) Protection* extends existing *Builtin Security Plugins* capabilities to protect bootstrapping traffic (such as participant discovery).
🛡	*Lightweight Security* offers a *Security Plugins* alternative for resource-constrained systems, leveraging Pre-Shared Key (PSK) Protection.
TOP SECRET	*New Security Algorithms* protect data up to TOP-SECRET level information.
🔌	*OpenSSL Providers* allow you to plug in a greater variety of external implementations for cryptographic operations without changes to your applications.

Note: For backward compatibility information between 7.3.0 LTS and previous releases, see the *Migration Guide* on the RTI Community Portal.

4.1 Dynamic Certificates Renewal and Revocation

When a certificate expires, certificate owners will be automatically removed, enabling long-running, uninterrupted operation of *Connext* secure systems. Dynamic Access Control can be created based on dynamic Identity Certificates that support renewal, Certificate Revocation Lists (CRL), or a whitelist of Identity Certificate Subject Names. Dynamic Certificates are seamlessly integrated with RTI Infrastructure Services and *Admin Console*.

See the following sections for more information on these features:

- *Changes Related to Dynamic Participant Renewal, Revocation, and Expiration in 7.0.0*
- *Changes Related to Dynamic Participant Renewal, Revocation, and Expiration in 7.1.0*
- *Changes Related to Dynamic Participant Renewal, Revocation, and Expiration in 7.2.0*

4.2 Pre-Shared Key (PSK) Protection

Pre-Shared Key (PSK) Protection expands the *Security Plugins* offering and enables basic-level protection wherever traditional DDS Security mechanisms are unavailable or infeasible due to limited resources, paramount performance requirements, or other reasons. The PSK secures all the traffic from the startup of a DDS *Entity* and restricts the communication only to *Entities* holding the correct pre-shared key seed.

Pre-Shared Key Protection can be leveraged in two different ways:

- As part of the *Builtin Security Plugins*:

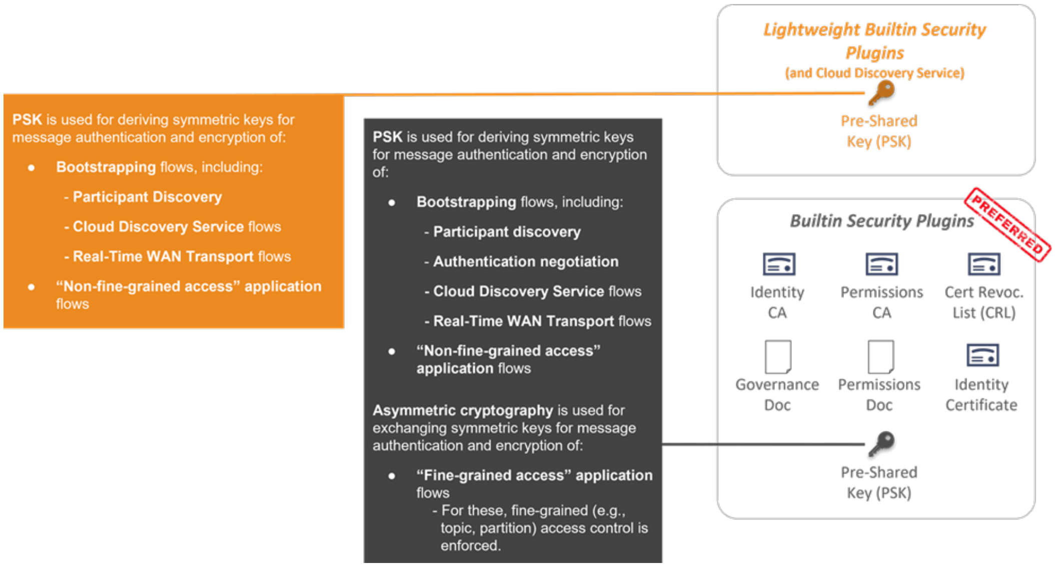
  Pre-Shared Key Protection works alongside existing *Builtin Security Plugins* features and secures the communication happening before and during authentication (known as bootstrapping). Note: while RTPS Bootstrapping messages can only be protected through Pre-Shared Key Protection, non-bootstrapping messages can be protected either with a combination of Pre-Shared Key Protection with other security mechanisms from *Builtin Security Plugins*, or by using non-Pre-Shared Key Protection mechanisms exclusively.

- As part of *Lightweight Builtin Security Plugins* (also known as *Lightweight Security*):

  In this case, all traditional DDS Security mechanisms are disabled and the entire communication is protected with Pre-Shared Key Protection.

  Note: Since Pre-Shared Key Protection by itself does not support granular security or topic permissions, *Lightweight Builtin Security Plugins* can only be used to provide domain-level protection from outsider adversaries.

For more information, see Pre-Shared Key Protection, in the RTI Security Plugins User's Manual.



PSK is used for deriving symmetric keys for message authentication and encryption of:

- **Bootstrapping** flows, including:
  - **Participant Discovery**
  - **Cloud Discovery Service** flows
  - **Real-Time WAN Transport** flows
- **"Non-fine-grained access" application** flows

PSK is used for deriving symmetric keys for message authentication and encryption of:

- **Bootstrapping** flows, including:
  - **Participant discovery**
  - **Authentication negotiation**
  - **Cloud Discovery Service** flows
  - **Real-Time WAN Transport** flows
- **"Non-fine-grained access" application** flows

Asymmetric cryptography is used for exchanging symmetric keys for message authentication and encryption of:

- **"Fine-grained access" application** flows
  - For these, fine-grained (e.g., topic, partition) access control is enforced.

4.3 Lightweight Security

This release of the *Security Plugins* includes *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 3 and wolfSSL crypto libraries. The new library, **nddslightweightsecurity**, is included with the *Security Plugins* bundles.

Using Pre-Shared Key Protection, *Lightweight Security* can protect the confidentiality or integrity of the communication, without the overhead of authentication, key exchange, and enforcing permissions. Therefore, the *Lightweight Builtin Security Plugins* library can be useful in resource-constrained scenarios.

The *Lightweight Builtin Security Plugins* library improves performance by not using the most demanding DDS Security mechanisms such as authentication or access control. It also reduces resource consumption from the CPU and memory. As a result, *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-configured key.

With *Lightweight Security*, 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 you must set consistently across the whole system. The property for configuring the seed is `dds.sec.crypto.rtps_psk_secret_passphrase`.

The entire communication is protected by default using the AES256+GCM cryptographic algorithm in ENCRYPT protection mode. You can choose another algorithm with the `dds.sec.crypto.rtps_psk_symmetric_cipher_algorithm` property. The available options are AES128+GCM and AES256+GCM. Likewise, you can change the protection mode with the `dds.sec.access.rtps_psk_protection_kind` property. The available options are NONE (do not protect), SIGN (protect the integrity), and ENCRYPT (protect the integrity and confidentiality).

The *Lightweight Builtin Security Plugins* library is also part of the *Security Plugins SDK*. This release also includes a tester for the *Lightweight Builtin Security Plugins*.

For more information, see:

- Lightweight Builtin Security Plugins, in the RTI Security Plugins User's Manual
- Lightweight Builtin Security Plugins and Security Plugins Interoperability, in the RTI Security Plugins User's Manual

4.4 New Security Algorithms

The Security Plugins can now operate at the Commercial National Security Algorithm (CNSA) Suite TOP-SECRET level. In particular, *Connext* 7 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 feature are described in:

- *Changes Related to Cryptographic Algorithms*
- *Changes Related to System Extensibility and Configurability*

4.5 OpenSSL Providers

OpenSSL Providers allow you to plug in a greater variety of external implementations for cryptographic operations without changes to your applications. See *Security Plugins now support OpenSSL providers* for more information.

Chapter 5

What's New in 7.3.0 LTS

SECURITY PLUGINS 7.3.0 LTS is a long-term support release that is built upon and includes all the features in releases 7.0.0, 7.1.0, and 7.2.0 (see *Previous Releases*). This section describes new features and improvements in 7.3.0 LTS, compared to 7.2.0.

5.1 Pre-Shared Key Protection and Lightweight Security

5.1.1 Pre-Shared Key Protection is compliant with OMG DDS Security 1.2 standard.

Pre-Shared Key Protection, in both the *Builtin Security Plugins* and the *Lightweight Builtin Security Plugins*, now meets the OMG DDS Security 1.2 standard. Changes span from renaming related properties to wire interoperability. See the *Migration Guide* on the RTI Community Portal.

5.1.2 Update to sender's key identifier used for pre-shared key derivation

The *Security Plugins* derive Pre-Shared keys from a combination of the secret key seed and some publicly available data. Among this publicly available data is the sender's key identifier. In previous releases, this value was equal to the *DomainId*. Starting in 7.3.0 LTS, the *Security Plugins* derive the sender's key identifier not only from the *DomainId* but also from the *DomainTag*. For more information, see How Pre-Shared Key Protection Works, in the RTI Security Plugins User's Manual.

5.1.3 New properties for configuring Pre-Shared Key protection

This release introduces two properties for configuring pre-shared key protection in the *Builtin Security Plugins* and the *Lightweight Builtin Security Plugins*. These properties are in the Cryptography plugin (prefix with `dds.sec.crypto.`):

- `rtps_psk_symmetric_cipher_algorithm`
- `rtps_psk_secret_passphrase`

There is also a new property in the Access Control plugin (prefix with `dds.sec.access.`) that you can use in the *Lightweight Builtin Security Plugins*:

- `rtps_psk_protection_kind`

For more information, see Configuring Pre-Shared Key protection, in the RTI Security Plugins User's Manual and the *Migration Guide*, on the RTI Community Portal.

5.1.4 Pre-Shared Key seed can be provided in a file

The `dds.sec.crypto.rtps_psk_secret_passphrase` (previously named `cryptography.rtps_protection_preshared_key`) property now supports the following two input formats:

- `data:,<ID>:<SEED>` allows you to provide the PSK seed explicitly in the property value.
  - `<ID>` is an integer [0, 254] that identifies the current seed within the system.
  - `<SEED>` is a string used to derive (in combination with publicly available data) the key used for encoding RTPS messages.
- `file:<FILENAME>` allows you to provide the seed in a supplemental file. The file can be optionally tracked by the *Security Plugins* (via the `files_poll_interval` property, enabled by default) for changes that are automatically incorporated into present operations.
  - `<FILENAME>` is the path to a file containing the PSK seed. The content of the file must use the `<ID>:<SEED>` format.

5.2 New Algorithms

5.2.1 Compliance with DDS Security 1.2 Specification regarding cryptographic algorithms

This release of the *Security Plugins* introduces several updates to comply with the DDS Security 1.2 Specification.

It includes changes to conform to naming, such as the addition of the `dds.sec.crypto.symmetric_cipher_algorithm` property and the `rtps_psk_protection_kind` Governance Document XML tag. There are updates to the expected behavior during discovery, such as not serializing the last algorithms if they have the default values, or only checking compatibility of the *DomainParticipant* symmetric cipher algorithms if they have protection enabled at the RTPS message level. This release also changes the values that identify several algorithms (custom ones, like `AES192+GCM`, and experimental ones, like `ECDHE-CEUM+X25519`) on the wire.

For more information, see Cryptographic Algorithms, in the RTI Security Plugins User's Manual.

5.2.2 New mask in Parameter ID indicates the default algorithm that endpoints use to protect data

The Parameter ID of the *DomainParticipant's* symmetric cipher algorithm information (`PID_ENDPOINT_SECURITY_SYMMETRIC_CIPHER_ALGO`) used to have three masks: `supported_mask`, `builtin_endpoints_required_mask`, and `builtin_kx_endpoints_required_mask`. This release introduces an additional one, `user_endpoints_default_required_mask`. The value of this mask indicates the default algorithm that the endpoints of the *DomainParticipant* use to protect user data traffic (unless the endpoint doesn't specify directly the algorithm to use).

The *Security Plugins* will not consider this mask when deciding if two *DomainParticipants* are compatible.

5.2.3 Changes in serialization of cryptographic algorithms

During discovery, *DomainParticipants* may propagate information about the cryptographic algorithms that they require or support. *DomainParticipants* propagate this information when they are configured with non-default cryptographic algorithms. Starting in this release, the *Security Plugins* will skip serialization of the last cryptographic algorithms in a PID (Parameter ID) if they have the default values.

5.3 Extensibility and Usability

5.3.1 Security Plugins now support OpenSSL providers

The OpenSSL default implementations of security algorithms may not meet the needs of every security application. As described in https://www.openssl.org/docs/man3.0/man7/provider.html , OpenSSL providers are a way of loading alternative implementations of algorithms. The *Security Plugins* now support OpenSSL providers, which are the OpenSSL 3.0 replacement for the OpenSSL 1.1.1 Engine API (see https://www.openssl.org/docs/man3.0/man7/migration_guide.html and search for the 'Engines and "METHOD" APIs' section). To use an OpenSSL provider with a *Security Plugins* application, you must use an OpenSSL configuration file to specify the providers you want to load, as described in https://www.openssl.org/docs/man3.0/man5/config.html, and you must set the appropriate environment variables. Then the *Security Plugins* will perform the following actions:

- Log, at a higher verbosity level, which providers are loaded and activated.
- Adjust the *DomainParticipant* discovery data based on which algorithms your providers support.
- Use your providers for security operations.

One example of a provider is the OpenSSL FIPS provider, described in https://www.openssl.org/docs/man3.0/man7/fips_module.html. RTI has tested that the OpenSSL FIPS provider works seamlessly with the *Security Plugins*, with the only caveat being that the FIPS provider does not support the DHE-MODP+2048+256 Key Establishment Algorithm.

See Support for OpenSSL Providers, in the Security Plugins User's Manual for more information.

5.3.2 Ability to send security events to logging aggregator backend with RTI Observability Framework

Now you can use RTI Monitoring Library 2.0 to forward security events logged by the Security Plugins' Logging plugin to RTI Observability Collector Service. The service will store the security events in a logging aggregator backend, such as Grafana® Loki®. Then you can use the message visualization and query functionalities offered by the aggregator backend and Grafana Dashboards.

For more information, see:

- Support for RTI Observability Framework, in the RTI Security Plugins User's Manual
- RTI Connext Observability Framework User's Manual

5.3.3 Improved format of Security Plugins SDK documentation

The *Security Plugins* SDK documentation used to be located inside the `RTI_SecurityPlugins_BuildableSourceCode_Instructions.txt` file of the root directory. This file contained instructions on how to build the SDK and run the tests, as well as API changes between releases. This information is now part of the new HTML documentation shipped with the *Security Plugins* SDK under `doc/html` (start by opening the `index.html` landing page). The new documentation shares the look and feel of RTI's other API documentation pages.

5.3.4 Reduced severity of benign log message related to ParticipantCryptoTokens

When using a Governance Document whose `rtps_protection_kind` is a value other than NONE, you may see a benign log message containing `PRESParticipant_processParticipantCryptoTokens`. This log message raises awareness of an automatically recoverable situation. Therefore, the `NDDS_Config_LogLevel` of the log message has been adjusted from `NDDS_CONFIG_LOG_LEVEL_ERROR` to `NDDS_CONFIG_LOG_LEVEL_STATUS_REMOTE`.

5.3.5 Deprecated property related to payload encoding alignment

The property `dds.data_writer.history.use_530_encoding_alignment` was introduced in *Security Plugins* 6.0.0 in order to mimic 5.3.0 behavior on the sending side of payload protection. However, this behavior does not impact backward interoperability with 5.3.0. The receiving side of payload protection can process 4-byte-aligned and unaligned protected payloads equally well. Therefore, this property is not necessary and has been deprecated. Support for this property may be removed in future versions of the *Security Plugins*. Using this property is highly discouraged.

5.4 Robustness and Security

5.4.1 Remote Secure DomainParticipants with Incompatible Configurations now Treated as Unauthenticated

If a local *DomainParticipant* has a Governance Document with the `allow_unauthenticated_participants` XML tag set to `TRUE`, communication will still happen over unsecure topics (those that have all the topic-level rules set to `FALSE` or `NONE`) even if the remote DomainParticipant is not authenticated.

In previous releases, a remote *DomainParticipant* was considered unauthenticated if it was not using security, or if the authentication process failed. However, if matching failed during discovery, the local and remote *DomainParticipants* wouldn't communicate with each other, even over unsecure topics. This behavior has changed.

Starting in this release, local *DomainParticipants* will also consider a remote *DomainParticipant* to be unauthenticated if it has an incompatible secure configuration or an incompatible set of security plugins. Communication with them may happen for unsecure topics if `allow_unauthenticated_participants` is TRUE.

5.4.2 Security Plugins now robust against sensitive information leaks when using non-secure Dynamic Memory Managers

To make the *Security Plugins* robust against potential sensitive information leaks when running on operating systems using non-secure Dynamic Memory Managers, the *Security Plugins* now sanitize the dynamically allocated buffers used to hold keys and passwords before releasing them.

Chapter 6

What's Fixed in 7.3.0 LTS

This section describes bugs fixed in Security Plugins 7.3.0 LTS. These are fixes since 7.2.0.

Security Plugins 7.3.0 LTS is a long-term support release that is built upon and includes all the fixes in releases 7.0.0, 7.1.0, and 7.2.0 (see *Previous Releases*). See the Connext Releases page on the RTI website for more information on RTI's software release model.

[Critical]: System-stopping issue, such as a crash or data loss. [Major]: Significant issue with no easy workaround. [Minor]: Issue that usually has a workaround. [Trivial]: Small issue, such as a typo in a log.

6.1 Interoperability

6.1.1 [Critical] Wrong out-of-the-box Governance configuration for legacy Builtin Secure Logging topic *

The Governance configuration for the `DDS:Security:LogTopic` legacy builtin topic was wrong in *Security Plugins* 7.2.0 out-of-the-box. This issue prevented `DDS:Security:LogTopic` *DataReaders* from being interoperable with previous versions of *Connext* and with other vendors. The workaround was to manually configure the topic-level security attributes in the Governance Document:

```
<topic_rule>
    <topic_expression>DDS:Security:LogTopic</topic_expression>
    <enable_discovery_protection>false</enable_discovery_protection>
    <enable_liveliness_protection>false</enable_liveliness_protection>
    <enable_read_access_control>true</enable_read_access_control>
    <enable_write_access_control>false</enable_write_access_control>
    <metadata_protection_kind>SIGN</metadata_protection_kind>
    <data_protection_kind>ENCRYPT</data_protection_kind>
</topic_rule>
```

Now there is no need to configure the builtin logging topic security attributes in the Governance Document. Subscriptions to both the legacy and non-legacy builtin logging topics should work out-of-the-box.

[RTI Issue ID SEC-2278]

6.1.2 [Major] Builtin Security Plugins incompatible with Lightweight Builtin Security Plugins when using non-default cryptographic algorithms *

Participant discovery matching between a *DomainParticipant* running the *Lightweight Builtin Security Plugins* and a *DomainParticipant* running the *Builtin Security Plugins* incorrectly evaluated as incompatible if the latter modified the `<allowed_security_algorithms>` tag in the Governance Document (if present) in a way that excluded the default security algorithms. (See allowed_security_algorithms (domain_rule), in the RTI Security Plugins User's Manual.)

Matching also failed if the *DomainParticipant* running the *Builtin Security Plugins* required an algorithm that is not part of the default set. For example, in the Governance Document of the *Builtin Security Plugins* you can restrict the supported key-exchange algorithm to `ECDHE-CEUM+P384` and then configure the `com.rti.serv.secure.authentication.key_establishment_algorithm` property. This configuration would have resulted in the following error message when trying to match with a Lightweight *DomainParticipant*:

```
ERROR [[...]{Entity=DR,MessageKind=DATA}|RECEIVE FROM [...]{Domain=0}|ASSERT␣
↪REMOTE DP|
GET REMOTE DP SECURITY STATE|LC:DISC,SEC]
PRESParticipant_getRemoteParticipantInitialSecurityState:[...]
"security info for authenticated remote participant [...]  does not match the␣
↪one for local participant [...].
Dropping participant announcement..."}}
```

Now the *Builtin Security Plugins* and the *Lightweight Builtin Security Plugins* properly interoperate when using a compatible configuration, as described in
Pre-Shared Key Protection in Lightweight Builtin Security Plugins vs. Pre-Shared Key Protection in Builtin Security Plugins, in the RTI Security Plugins User's Manual.

[RTI Issue ID SEC-2286]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.2 Discovery and Authentication

6.2.1 [Critical] Participant discovery failed to complete after re-authentication when using SPDP2 *

If an asymmetric liveliness occurred between two participants P1 and P2 where P1 lost liveliness with P2, P1 would remove the associated states but P2 would keep the authenticated state. P2 could re-authenticate P1 if P1 sent it an Authentication Request message. Participants using SPDP2 would not complete re-discovery after re-authentication completed and communication would not be possible.

[RTI Issue ID SEC-2336]

6.2.2 [Critical] Participants using SPDP2 failed to complete discovery if using Pre-Shared Key protection with domain tags *

DomainParticipants using SPDP2 failed to complete discovery if using Pre-Shared Key (PSK) Protection and having matching domain tags. Now, *DomainParticipants* can use SPDP2 with Pre-Shared Key Protection and domain tags.

[RTI Issue ID SEC-2339]

6.2.3 [Major] allow_unauthenticated_participants did not allow authenticated participant to replace unauthenticated one

The DDS Security 1.2 specification has this sentence about the Governance Document tag `allow_unauthenticated_participants`:

```
Additionally, a DomainParticipant that later authenticates would kick out the␣
↪unauthenticated DomainParticipant if it has the same GUID.
```

The *Security Plugins* did not implement this behavior. The authentication attempt from the new *DomainParticipant* was incorrectly ignored. Now, this behavior works as long as the `DomainParticipantQos` property `dds.participant.discovery_config.use_stateless_participant_discovery_reader` is set to `TRUE`.

For details about the new behavior, see allow_unauthenticated_participants (domain_rule) in the RTI Security Plugins User's Manual.

[RTI Issue ID SEC-1660]

6.2.4 [Major] Participants with SPDP2 failed to complete discovery if using Lightweight Builtin Security Plugins or HMAC-only mode *

Participants using SPDP2 failed to complete participant discovery if they were also using the *Lightweight Builtin Security Plugins,* or the *Builtin Security Plugins* with HMAC-only mode. This has been resolved and participants with SPDP2 can now use the *Lightweight Builtin Security Plugins* or HMAC-only mode.

[RTI Issue ID SEC-2338]

6.2.5 [Major] Participants with SPDP2 and allow_unauthenticated_participants failed to communicate if authentication failed *

Participants using SPDP2 with `allow_unauthenticated_participants` set to TRUE failed to communicate if both participants were using security and authentication failed. This has been resolved for the case where both participants have `allow_unauthenticated_participants` set to TRUE and both participants fail authentication. In this case, communication between the participants will proceed for all non-secure topics.

[RTI Issue ID SEC-2340]

6.2.6 [Major] Errors setting the RSA padding kind did not trigger a failure in signing or verifying DomainParticipant data *

If the `authentication.rsa_pss_pad` plugin-specific property is set to `AUTO` or `TRUE`, then the *DomainParticipant* may:

- Use *RSA_PKCS1_PSS_PADDING* padding when signing messages (depending on the Identity Certificate).
- Accept *RSA_PKCS1_PSS_PADDING* padding when verifying the remote *DomainParticipant's* signed messages.

Signing and verifying messages should fail if there are errors when setting the right RSA padding. This was not the case for previous releases of the *Security Plugins* (7.0, 7.1, and 7.2). The *Builtin Security Plugins* logged `RTI_Security_CertHelper_setRsaPadding:OpenSSL function EVP_PKEY_CTX_set_rsa_mgf1_md failed with error` when they couldn't set the RSA padding kind, but the sign/verify process continued. As a consequence, the signature type may not have been the one the user expected. This release of the *Security Plugins* fixes the issue. Errors in setting the RSA padding kind during signing or verifying a message result in a failure.

[RTI Issue ID SEC-2392]

6.2.7 [Minor] Failure to announce lack of PSS Padding support when authentication.rsa_pss_pad property was FALSE *

When the `authentication.rsa_pss_pad` property is set to `FALSE` in the *Builtin Security Plugins*, the associated *DomainParticipant* is expected to announce through discovery the lack of support for the `RSASSA-PSS-MGF1SHA256+2048+SHA256` algorithm to other *DomainParticipants*. *Security Plugins* 7.0, 7.1, and 7.2 had a bug in this mechanism. As a result, two incompatible *DomainParticipants* may have incorrectly matched during discovery. Then these *DomainParticipants* would have tried to authenticate, resulting in a failed authentication.

In previous releases, you could have worked around this issue by configuring the `<allowed_security_algorithms>` tag in your Governance Document, and omitting `RSASSA-PSS-MGF1SHA256+2048+SHA256` in your `<digital_signature_identity_trust_chain>` algorithms. The workaround is no longer necessary.

[RTI Issue ID SEC-2394]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.3 Usability

6.3.1 [Major] Incorrect information about cryptographic algorithms displayed in Admin Console *

In *Connext* releases 7.0.0 through 7.2.0, information about default cryptographic algorithms was displayed in *Admin Console* even though the *Security Plugins* were not enabled. The algorithms were displayed when viewing a DomainParticipant's QoS. Now this information is not displayed until the *Security Plugins* are enabled.

[RTI Issue ID SEC-2069]

** This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.4 Cryptography

6.4.1 [Major] Non-random initial session ID for entities other than Secure Key Exchange Channel

For entities other than the Secure Key Exchange Channel, the initial Session ID was always set to 0x00000001. This behavior violated section 9.5.3.3.2 of the DDS Security specification, which states that the Session ID is initially a random value.

The initial Session ID is now a random value for all entities.

[RTI Issue ID SEC-2220]

6.4.2 [Major] Memory leak when calling set_qos() without changing pre-shared key seed ID *

A memory leak happened if Pre-Shared Key Protection was enabled and the *DomainParticipant's* `set_qos()` API was called without changing the value of the `cryptography.rtps_protection_preshared_key` property. The number of bytes leaked was the same as the length of the pre-shared key seed. This issue affects the *Security Plugins* 7.2.0 release.

[RTI Issue ID SEC-2292]

** This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.5 Cryptographic Algorithms

6.5.1 [Major] Security Plugins for wolfSSL incorrectly propagated mask indicating support for Diffie-Hellman key establishment algorithm *

In previous releases, the *Security Plugins* for wolfSSL propagated a mask that incorrectly showed support for the Diffie-Hellman key establishment algorithm. As a result, two *DomainParticipants* could be considered compatible during discovery (with respect to their algorithms) but fail to establish a shared secret key. Starting in this release, the *Security Plugins* for wolfSSL will not set the bit for Diffie-Hellman in the mask of supported key establishment algorithms.

[RTI Issue ID SEC-2362]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.6 Dynamic Participant Renewal, Revocation, and Expiration

6.6.1 [Critical] Invalid read when simultaneously changing a file and changing a property value for an identity certificate or CRL *

Suppose you had set `authentication.identity_certificate_file_poll_period.millisec` to a value other than 0. If you changed the contents of your identity certificate file and then called `set_qos()` to change the `dds.sec.auth.identity_certificate` property value, a race condition would have occurred because those two operations were not thread-safe with respect to each other. This race condition led to the reading of invalid memory. 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. A similar problem existed for CRLs (the affected properties were `authentication.crl_file_poll_period.millisec` and `authentication.crl`).

Note that in *Security Plugins* 7.3.0, the two `poll_period.millisec` properties mentioned above have been replaced with a new property called `files_poll_interval`.

[RTI Issue ID SEC-2384]

6.6.2 [Critical] Invalid read when simultaneously changing an identity certificate file and authenticating another participant *

Suppose you had set `authentication.identity_certificate_file_poll_period.millisec` to a value other than 0. If you changed the contents of your identity certificate file while authenticating another *DomainParticipant*, a race condition would have occurred because those two operations were not thread-safe with respect to each other. This race condition led to the reading of invalid memory. 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.

Note that in *Security Plugins* 7.3.0, the `poll_period.millisec property` mentioned above has been replaced with a new property called `files_poll_interval`.

[RTI Issue ID SEC-2405]

6.6.3 [Minor] Changing identity certificate property from string to equivalent file not detected

The *Builtin Security Plugins* did not detect changes to a *DomainParticipant's* identity certificate in the following scenario:

1. The `dds.sec.auth.identity_certificate` property was set to a data string (using the `data:,` prefix).
2. The `authentication.identity_certificate_file_poll_period.millisec` property was set to a non-zero value.
3. The *DomainParticipant's* identity certificate was changed from the value with the `data:,` prefix to an equivalent value with the `file:` prefix. The file contents were the same as the string in the `data:,` value.

In this case, the poll period was ineffective and the *Builtin Security Plugins* failed to detect any changes in the file.

The same problem occurred if you started with one file and then changed to a different file with the same contents: the *Builtin Security Plugins* would not detect changes in the new file.

[RTI Issue ID SEC-2319]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.7 Logging

6.7.1 [Minor] DomainParticipants in same application may have have had different verbosities for logging security messages *

The last *DomainParticipant* explicitly that set the plugin-specific `logging.verbosity` property (set by you, in the QoS settings) will apply that setting to all the *DomainParticipants* within the application and all security-related messages logged from the *Connext* libraries. This is the behavior in all *Connext* releases, except for 7.2.0.

Connext 7.2.0 introduced support for running a combination of *DomainParticipants* from the *Builtin Security Plugins* library (**nddssecurity**) and the *Lightweight Builtin Security Plugins* library (**nddslightweightsecurity**) within the same application. In 7.2.0, the logging verbosity (for all security-related messages) of all the *DomainParticipants* belonging to the same security library was determined by the last *DomainParticipant* of that library (as opposed to the last *DomainParticipant* within the application) created by your application.

Connext 7.3.0 reverts this behavior. The logging verbosity for all security-related messages is again determined by the last *DomainParticipant* that your application creates.

[RTI Issue ID SEC-2368]

6.7.2 [Trivial] Typo in “non-compliant DDS Security implementation” log message

There was a typographical error in the “non-compliant DDS Security implementation” warning message that you see when interoperating with certain DDS Security vendors:

```
This is likely caused by a non-compliantDDS Security implementation.
```

Notice there was a space missing after `compliant`. This problem only affected 7.2.0 and 6.1.2.10.

[RTI Issue ID SEC-2248]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.8 Crashes

6.8.1 [Critical] Security Plugins crashed when enabling pre-shared key protection after DomainParticipant creation *

The *Builtin Security Plugins* crashed if you tried to enable pre-shared key protection (setting the `cryptography.rtps_protection_preshared_key` property) in an already created *DomainParticipant* that had a Governance Document with `rtps_preshared_secret_protection_kind` equal to NONE. You would have seen a backtrace that included the `RTI_Security_CryptographyPresharedKeyState_setIdAndString()` function.

Enabling or disabling pre-shared key protection at runtime is not allowed. If you try to do so, the *Builtin Security Plugins* will now fail gracefully.

The issue only affected *Security Plugins* 7.2.0.

The *Security Plugins* 7.3.0 LTS removed the `cryptography.rtps_protection_preshared_key` property and introduced the `dds.sec.crypto.rtps_psk_secret_passphrase` property, defined in the OMG DDS Security specification, version 1.2. This release also renames the `rtps_preshared_secret_protection_kind` XML tag of the Governance Document to `rtps_psk_protection_kind`.

[RTI Issue ID SEC-2246]

6.8.2 [Critical] Segmentation fault when running out of memory while calling custom OpenSSL allocator function *

When setting `authentication.enable_custom_allocators` to TRUE, running out of memory while calling the custom OpenSSL `realloc` allocator function triggered a segmentation fault. This issue only affected *Security Plugins* 7.0.0 and above.

[RTI Issue ID SEC-2326]

6.8.3 [Critical] Crash when polling a file and changing QoS from file to string *

If you created a *DomainParticipant* with the properties `dds.sec.auth.identity_certificate` set to a filename and `authentication.identity_certificate_file_poll_period.millisec` set to a non-zero value, and then you changed the identity certificate from a value with the `file:` prefix to a value with the `data:,` prefix, you would have seen this error when using the debug libraries:

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

When using the release libraries, this scenario caused a segmentation fault. There was a similar problem for the `authentication.crl` property.

Note: `authentication.identity_certificate_file_poll_period.millisec` has been replaced with `files_poll_interval` (see :ref:`section-SEC-2083`).

These problems only affected *Security Plugins* 7.2.0.

[RTI Issue ID SEC-2317]

6.8.4 [Critical] Segmentation fault when running out of memory during creation of local crypto tokens

Running out of memory during the creation of the local crypto tokens would later lead to a segmentation fault when destroying the endpoint.

The segmentation fault, which happened in the function `PRESPsService_removeMatchSecurity()`, called by `PRESPsService_destroyLocalEndpointWithCursor()`, will no longer occur.

[RTI Issue ID SEC-2303]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.9 Hangs

6.9.1 [Critical] Deadlock when simultaneously changing a file and changing a property value for an identity certificate *

Suppose you had set `authentication.identity_certificate_file_poll_period.millisec` to a value other than 0. If you changed the contents of your identity certificate file and then called `set_qos()` to change the `dds.sec.auth.identity_certificate` property value, a race condition would have occurred because those two operations were not thread-safe with respect to each other. This race condition led to a deadlock, which led to a hang.

Note that in *Security Plugins* 7.3.0, the `authentication.identity_certificate_file_poll_period.millisec` property has been replaced with a new property called `files_poll_interval`.

[RTI Issue ID SEC-2393]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.10 Memory Leaks/Growth

6.10.1 [Minor] Memory leak when running out of memory while validating permissions of local or remote DomainParticipants

If either of the internal functions `RTI_Security_AccessControl_validate_local_permissions()` or `RTI_Security_AccessControl_validate_remote_permissions()` ran out of memory, then validation of the permissions would fail with a memory leak. Here is one example set of error messages, along with a valgrind result:

```
ERROR [CREATE DP|LC:DISC,SEC]RTI_Security_AccessControl_validate_local_
↪permissions:{"DDS:Security:LogTopicV2":{"f":"10","s":"1","t":{"s":
↪"1701168980","n":"715641999"},"h":"RTISP-10036","i":"0.0.0.0","a":"RTI␣
↪Secure DDS Application","p":"22166","k":"security","x":[{"DDS":[{"domain_id
↪":"<unknown>"},{"guid":"<unknown>"},{"plugin_class":"Access Control"},{
↪"plugin_method":"RTI_Security_AccessControl_validate_local_permissions"}]}],
↪"m":"failed to validate local permissions. XML file:"}}
[...]
ERROR [CREATE DP|LC:DISC,SEC]DDS_DomainParticipantTrustPlugins_
↪getLocalParticipantSecurityState:VALIDATION FAILURE | Local permissions.
ERROR [CREATE DP|LC:DISC,SEC]DDS_DomainParticipantTrustPlugins_
↪getLocalParticipantSecurityState:RETURN FAILURE | Reverting local DP␣
↪security state due to the errors.
ERROR [CREATE DP|LC:DISC,SEC]DDS_DomainParticipant_createI:INVALID␣
↪CONFIGURATION | New DP's security state is incorrect.
ERROR LC:DISC| DDS_DomainParticipantFactory_create_participant_disabledI:!
↪create participant
[...]

==22166== 144 bytes in 1 blocks are definitely lost in loss record 1 of 1
==22166==    at 0x484DA83: calloc (in /usr/libexec/valgrind/vgpreload_
↪memcheck-amd64-linux.so)
==22166==    by 0x498454F: RTIOsapiHeap_reallocateMemoryInternal (heap.c:830)
==22166==    by 0x1631CBC: RTI_Security_Heap_allocate (InfrastructurePSM.
↪c:366)
==22166==    by 0x15740E9: RTI_Security_AccessControl_validatePermissions␣
↪(AccessControl.c:525)
==22166==    by 0x1576226: RTI_Security_AccessControl_validate_local_
↪permissions (AccessControl.c:1102)
==22166==    by 0x932607: DDS_DomainParticipantTrustPlugins_
↪getLocalParticipantSecurityState (DomainParticipantTrustPlugins.c:2466)
==22166==    by 0x97062E: DDS_DomainParticipant_createI (DomainParticipant.
↪c:11719)
==22166==    by 0x90B6F2: DDS_DomainParticipantFactory_create_participant_
↪disabledI (DomainParticipantFactory.c:2816)
==22166==    by 0x907E8E: DDS_DomainParticipantFactory_create_participant␣
↪(DomainParticipantFactory.c:1542)
```

The leak only happened if memory was already exhausted, so this problem did not lead to unbounded memory growth.

Now, those two functions will fail, without a memory leak.

[RTI Issue ID SEC-2331]

6.10.2 [Trivial] Memory leak when running Security Plugins SDK tester *

Running a *Security Plugins* tester (`AccessControlTester`, `CryptographyTester`, or `LightweightTester`) caused a memory leak, because the testers didn't finalize the DomainParticipantFactory. Here is an extract from the valgrind result:

```
==23517== HEAP SUMMARY:
==23517==     in use at exit: 1,802,975 bytes in 11,641 blocks
==23517==   total heap usage: 67,021 allocs, 55,380 frees, 44,232,249 bytes␣
 ↪allocated
==23517==
==23517== 5 bytes in 1 blocks are possibly lost in loss record 11 of 1,645
==23517==    at 0x484DA83: calloc (in /usr/libexec/valgrind/vgpreload_
 ↪memcheck-amd64-linux.so)
==23517==    by 0x653E49E: RTIOsapiHeap_reallocateMemoryInternal (heap.c:821)
==23517==    by 0x6688605: RTIXMLDtdAttribute_new (DtdParser.c:181)
==23517==    by 0x66896F0: RTIXMLDtdParser_onAttlistDecl (DtdParser.c:481)
==23517==    by 0x66758CB: RTI_doProlog (xmlparse.c:5013)
==23517==    by 0x6674539: RTI_externalParEntProcessor (xmlparse.c:4589)
==23517==    by 0x667416C: RTI_externalParEntInitProcessor (xmlparse.c:4460)
==23517==    by 0x666E2BC: RTI_XML_Parse (xmlparse.c:1938)
==23517==    by 0x668A301: RTIXMLDtdParser_parse (DtdParser.c:703)
==23517==    by 0x6682D84: RTIXMLParser_onExternalEntityRef (Parser.c:819)
==23517==    by 0x66752B2: RTI_doProlog (xmlparse.c:4913)
==23517==    by 0x66746E7: RTI_prologProcessor (xmlparse.c:4637)
==23517==
==23517== 6 bytes in 1 blocks are possibly lost in loss record 21 of 1,645
==23517==    at 0x484DA83: calloc (in /usr/libexec/valgrind/vgpreload_
 ↪memcheck-amd64-linux.so)
==23517==    by 0x653E49E: RTIOsapiHeap_reallocateMemoryInternal (heap.c:821)
==23517==    by 0x6635829: REDAString_duplicate (String.c:1597)
==23517==    by 0x668AA9E: RTIXMLExtensionClass_initialize (ExtensionClass.
 ↪c:143)
==23517==    by 0x668AE08: RTIXMLExtensionClass_new (ExtensionClass.c:208)
==23517==    by 0x5DCA557: DDS_XMLParser_register_builtin_extensions (Parser.
 ↪c:542)
==23517==    by 0x5DCC616: DDS_XMLParser_initialize_w_params (Parser.c:1046)
==23517==    by 0x5DCC7A7: DDS_XMLParser_new_w_params (Parser.c:1077)
==23517==    by 0x59E0B3E: DDS_QosProvider_initialize (QosProvider.c:2620)
==23517==    by 0x59E0E3E: DDS_QosProvider_new (QosProvider.c:2672)
==23517==    by 0x59EAD57: DDS_DomainParticipantFactory_initializeI␣
 ↪(DomainParticipantFactory.c:3750)
==23517==    by 0x59E2F22: DDS_DomainParticipantFactory_newI␣
 ↪(DomainParticipantFactory.c:891)
[...]
```

To correct this, the testers now call `DDS_DomainParticipantFactory_finalize_instance()` upon finalization.

[RTI Issue ID SEC-2279]

** This bug does not affect you if you are upgrading from 6.1.x or earlier.*

6.11 Vulnerabilities

6.11.1 [Major] Pre-Shared Key Protection plus Cloud Discovery Service did not protect against Participant Announcement Replay attacks *

The new Pre-Shared Key Protection feature was introduced as a way to protect *Cloud Discovery Service* in version 7.2.0. However, the CDS + PSK combination did not protect against Participant Announcement Replay attacks as described in the Security Plugins User's Manual. Now this combination will protect against these attacks.

[RTI Issue ID SEC-2287]

** This bug does not affect you if you are upgrading from 6.1.x or earlier.*

** This bug does not affect you if you are upgrading from 6.1.x or earlier.*

Chapter 7

Previous Releases

7.1 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.

7.1.1 Changes Related to Interoperability

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.

As part of this change, the seconds field of the `DDS_Time` structure has been updated from a 32-bit value to a 64-bit value. This change to the seconds field was made in anticipation of a future update to the OMG DDS specification, but for now the change is an RTI extension. Therefore, this change could affect portability of code between DDS vendors who follow the specification as it is today.

Year 2038 support applies only to this and future releases of *Connext*; there are currently no plans to backport it to previous releases.

Builtin Security Plugins interoperates 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.
```

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

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

7.1.2 Changes Related to Access Control

Improved endpoint discovery time by removing redundant calling of check_remote_topic

The OMG 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.

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 *Builtin Security Plugins* to only accept certificates including X.509 v3 key usage extensions when using wolfSSL as the underlying cryptographic library. For more information, see the `authentication.x509v3_extension_enforcement.key_usage` property in Table 4.2 Properties for Configuring Authentication of the *RTI Security Plugins User's Manual*.

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, 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*.

7.1.3 Changes Related to Cryptographic Algorithms

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).

7.1.4 Changes Related to Cryptography

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.

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*.

7.1.5 Changes Related to Discovery and Authentication

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.

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`.

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.

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.

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.

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

Dynamically control access to your Security Plugins 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.

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*.

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*.

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.

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 *Builtin Security Plugins* enforce these changes as long as `com.rti.serv.secure.authentication.identity_certificate_file_poll_period.millisec` (in release 7.3, this property is replaced by a new property, `files_poll_interval`) is set to a value other than `0`. 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 *RTI Security Plugins User's Manual*.

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 *Builtin Security Plugins* enforce these changes as long as `authentication.crl_file_poll_period.millisec` (replaced by a new property, `files_poll_interval`, in 7.3) is set to a value other than `0`. 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 *RTI Security Plugins User's Manual*.

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*.

7.1.7 Changes Related to Platforms and Builds

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 RTI Security Plugins Installation Guide to get started with the Security Plugins for wolfSSL.

7.1.8 Changes Related to Security Plugins SDK

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.

Added support for building and testing Lightweight Security library

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

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.

7.1.9 Changes Related to System Extensibility and Configurability

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.

7.1.10 Changes Related to Third-Party Software

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.

7.2 What's Fixed in 7.2.0

[Critical]: System-stopping issue, such as a crash or data loss. [Major]: Significant issue with no easy workaround. [Minor]: Issue that usually has a workaround. [Trivial]: Small issue, such as a typo in a log.

7.2.1 Fixes Related to Security Plugins SDK

[Trivial] 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]

7.2.2 Fixes Related to Crashes

[Critical] 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]

7.2.3 Fixes Related to Cryptography

[Critical] 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]

[Major] 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]

7.2.4 Fixes Related to Access Control

[Major] 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]

7.2.5 Fixes Related to Interoperability

[Critical] 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). 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 7.1: 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 7.2: Endpoint Discovery PIDs

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

[RTI Issue ID SEC-2071]

[Critical] Placement of GUID within RTPS message incorrectly affected vendor interoperability

In previous releases, the *Builtin 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]

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

[Critical] 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]

[Critical] 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]

[Major] Using a preshared key and calling banish_ignored_participants led to decoding failures *

When using Pre-Shared Key Protection with the *Builtin 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]

[Major] Builtin Security Plugins 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 of Security Plugins for wolfSSL, the Builtin Security Plugins 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]

7.2.7 Fixes Related to Usability

[Major] Lightweight Builtin Security Plugins Library and Builtin Security Plugins Library could not be simultaneously loaded into the same application *

Previously, trying to simultaneously load both the *Lightweight Builtin Security Plugins* library (`nddslightweightsecurity`) and the *Builtin 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 Builtin Security Plugins* in your application, see Configuring the Lightweight Builtin Security Plugins in the *RTI Security Plugins User's Manual* (link).

[RTI Issue ID SEC-2077]

[Minor] 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]

7.2.8 Fixes Related to XML Configuration

[Trivial] 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 (since renamed `rtps_psk_protection_kind`). Instead of:

```
<xs:element name="rtps_psk_protection_kind" type="BasicProtectionKind" >
```

It should be:

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

This issue has been fixed.

[RTI Issue ID SEC-2090]

7.2.9 Fixes Related to Discovery and Authentication

[Critical] Discovery time scaled poorly

Endpoint discovery time scaled poorly as the number of endpoints increased. Moreover, when using the *Lightweight Builtin Security Plugins* or the *Builtin Security Plugins* running under *HMAC-Only mode*, 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 have been fixed. The discovery time is now comparable with that of Security Plugins 5.3.1.

[RTI Issue ID SEC-2170]

[Major] 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]

[Trivial] Builtin Security Plugins 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.

In previous releases of Security Plugins for wolfSSL, the Builtin Security Plugins did try to verify the certificate against all the Certificate Authorities. As a result, the Builtin 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]

7.2.10 Fixes Related to Shipped Examples

[Trivial] 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]

7.2.11 Fixes Related to Vulnerabilities

[Critical] 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]

7.2.12 Other Changes

[Trivial] Confidential property not listed in Release Notes

In release 7.1.0, the new property `rtps_protection_preshared_key` was documented in the RTI 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]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

7.3 What's New in 7.1.0

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

7.3.1 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 *Builtin Security Plugins* check that the desired certificate is currently valid, as described in Verifying the certificate validity on the current date and time in the *RTI Security Plugins User's Manual*. When the certificate is about to expire, you may want to be notified so that you can replace the certificate with one that expires later. In this release, the *Builtin 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 *RTI 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 *RTI Security Plugins User's Manual*, when mutually authenticating with a remote *DomainParticipant*, the local *DomainParticipant* checks that the remote *DomainParticipant's* certificate is currently valid.

If the certificate is currently valid but later becomes expired, the local *DomainParticipant* may want to stop communicating with the remote *DomainParticipant*. In this release, the *Builtin 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 *RTI Security Plugins User's Manual*.

7.3.2 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 *Builtin Security Plugins*: RTPS PSK Protection works alongside existing *Builtin Security Plugins* features and secures the communication that occurs before two participants authenticate each other.
- As part of *Lightweight Builtin Security Plugins*: 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.

7.3.3 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 Builtin Security Plugins* library are not interoperable with those from the *Builtin Security Plugins* library (**nddssecurity**).

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

7.3.4 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 *RTI 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 OMG DDS Security 1.2 specification (pending publication). 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 *RTI Security Plugins User's Manual*
- Relevant Types for the Security Algorithms in the *RTI Security Plugins User's Manual*

7.3.5 Changes Related to Usability

7.3.6 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 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.

7.3.7 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 *RTI Security Plugins User's Manual*), please note that the Security Plugins 7.1.0 do not support providers (see https://www.openssl.org/docs/man3.0/man7/provider.html), but the Security Plugins 7.3.0 do support providers.

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.

7.3.8 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.

7.3.9 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

7.3.10 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 *RTI 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 *RTI Security Plugins User's Manual* for more information.

7.3.11 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”

7.4 What’s Fixed in 7.1.0

[Critical]: System-stopping issue, such as a crash or data loss. [Major]: Significant issue with no easy workaround. [Minor]: Issue that usually has a workaround. [Trivial]: Small issue, such as a typo in a log.

7.4.1 Fixes Related to Discovery and Authentication

[Critical] 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]

[Critical] 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]

[Minor] 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]

7.4.2 Fixes Related to Cryptography

[Critical] 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 cases, in a crash. This issue, which did not affect the confidentiality or integrity of *Connext* applications, has been fixed.

[RTI Issue ID SEC-1892]

[Critical] 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 *RTI 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]

[Critical] 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]

[Critical] 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 *RTI Security Plugins User's Manual*.

These problems have been resolved.

[RTI Issue ID SEC-1825]

[Critical] 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 the 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]

[Major] 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 *[Major] 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]

[Major] 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]

[Major] 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]

[Minor] 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]

[Trivial] 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]

7.4.3 Fixes Related to Reliability Protocol and Wire Representation

[Minor] 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]

7.4.4 Fixes Related to Entities

[Major] 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]

7.4.5 Fixes Related to Shipped Examples

[Trivial] 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]

7.4.6 Fixes Related to Vulnerabilities

[Critical] 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]

[Critical] 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]

[Critical] 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]

7.4.7 Fixes Related to SECURITY PLUGINS SDK

[Minor] 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]

7.4.8 Fixes Related to Crashes

[Critical] 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.

Specifically, 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]

* *This bug does not affect you if you are upgrading from 6.1.x or earlier.*

7.5 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 SECURITY PLUGINS 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.

7.5.1 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.

7.5.2 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 *Builtin 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 *RTI 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 *Builtin Security Plugins* will enable additional digital signature and key establishment algorithms that are not part of the OMG 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.

7.5.3 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 or 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 *RTI Security Plugins User's Manual*, and
- How the XML is Validated, in the RTI Connext Core Libraries User's Manual.

7.5.4 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.

7.5.5 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 *Builtin 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 *RTI Security Plugins User's Manual*), and the Permissions Document (see Permissions Document in the *RTI Security Plugins User's Manual*).

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

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

7.5.6 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 *RTI 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.

7.5.7 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.

7.5.8 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

7.6 What's Fixed in 7.0.0

[Critical]: System-stopping issue, such as a crash or data loss. [Major]: Significant issue with no easy workaround. [Minor]: Issue that usually has a workaround. [Trivial]: Small issue, such as a typo in a log.

7.6.1 Fixes Related to Discovery and Authentication

[Major] 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]

[Major] 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]

7.6.2 Fixes Related to Cryptography

[Major] 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]

7.6.3 Fixes Related to Access Control

[Major] When parsing domain rules from a Permissions document, *Builtin Security Plugins* applied an incorrect order-of-precedence

In 6.1.1, the *Builtin 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]

7.6.4 Fixes Related to Interoperability with Other Vendors

[Major] Could not detect participant discovery changes from DomainParticipants using non-Security Plugins

If a *DomainParticipant* using 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 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]

[Trivial] 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 Security Plugins or *Connext Micro*, but it might have caused interoperability issues with other vendors. The issue has been fixed.

[RTI Issue ID SEC-1674]

7.6.5 Fixes Related to Debuggability

[Minor] 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]

[Minor] 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]

[Minor] 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]

[Trivial] 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]

7.6.6 Fixes Related to the Security Plugins SDK

[Trivial] 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]

7.6.7 Fixes Related to Shipped Examples

[Trivial] 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 8

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.

8.1 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
```

8.2 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]

8.3 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]

8.4 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]

8.5 '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]

8.6 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]