SECURITY SERVICES FOR DDS SECURITY PLUGINS

Based on Arm TrustZone

Arm Robotics Software

1 Overview

1.1 Introduction

This document describes a general design of security services for Object Management Group (OMG) Data Distribution Service (DDS) Security Plugin[5]. The security services support and implement the internal security operations required in DDS Security Plugins logic.

This document also specifies a design of the security services implemented on arm platforms, based on arm TrustZone[1]. Arm TrustZone can protect the security assets in Secured World from leakage even if the Normal World is attacked. Please refer to Arm TrustZone page[1] for more features.

DDS security services implementation on arm platforms should be built up as a generic framework of DDS security support, e.g., providing generic DDS Security support to ROS 2.

Besides, the implementation on arm platforms should provide generic interfaces as DDS Security specification requires, other than special implementations to a specific vendor DDS product.

Chapter 2 defines the general architecture and block diagrams. Some examples of general implementations are also showed.

Chapter 3 describe the generic internal APIs for DDS Security Plugins.

Chapter 4 discuss about Arm DDS *Secure Library*.

Chapter 5 specifies the requirements to DDS security service implementations on arm platform.

1.2 References

[1] Arm TrustZone: https://www.arm.com/products/security-on-arm/trustzone

[2] ROS2 wiki: https://github.com/ros2/ros2/wiki

[3] DDS: Data-Distribution Service for Real-Time Systems version 1.4. http://www.omg.org/spec/DDS/1.4/

[4] DDS-RTPS: Data-Distribution Service Interoperability Wire Protocol version 2.2. http://www.omg.org/spec/DDSI-RTPS/2.2/

[5] DDS-SECURITY^{TM}: DDS Security Version 1.0. http://www.omg.org/spec/DDS-SECURITY/1.0/

[6] SROS 2 on Github. https://github.com/ros2/sros2

[7] GlobalPlatform Device Technology TEE System Architecture Version 1.1 Public Release. https://www.globalplatform.org/specificationsdevice.asp

[8] GlobalPlatform Device Technology TEE Internal Core API Specification Version 1.1.1 Public Release. https://www.globalplatform.org/specificationsdevice.asp

[9] GlobalPlatform Device Technology TEE Client API Specification Version 1.0 Public Release. https://www.globalplatform.org/specificationsdevice.asp

[10] eProsima Fast RTPS: http://www.eprosima.com/index.php/products-all/eprosima-fast-rtps

[11] eProsima Fast RTPS Documentation. http://docs.eprosima.com/en/latest/

[12] eProsima Fast RTPS Security Documentation. http://docs.eprosima.com/en/latest/security.html

[13] RTI Connext DDS: https://www.rti.com/products/dds

[14] DDS Security Plugin Internal Security APIs Specification.

1.3 Revision History

Revision	Date	Description	Author
0.01	Sept 27 2017	Draft. Complete a draft of architecture and general work flow.	David Hu
0.1	Oct 16 2017	Refine the design Improve the block diagram and work flow. Clean up the potential security risks and design issues.	David Hu
0.2	Oct 17 2017	Fix issues found in the review. Fix typo and format issues.	David Hu
0.3	Oct 25 2017	Re-design the general architecture. Improve the design as a more generic one. Remove unnecessary implementation details. Adjust the design and the document accordingly. Move the introduction chapter to another document.	David Hu
1.0	Oct 26 2017	Update the document according to the review feedback. Release for Arm internal review.	David Hu
1.1	Nov 22 2017	Modify the document based on the latest design. Focus on the general design of generic internal API and the Arm DDS *SecureLib*. Remove the descriptions of work flow of DDS Security Plugin logic.	David Hu

1.4 Terms and Definitions

This terms and definitions mentioned in this design document are compatible to those in OMG DDS specifications.

Data-Distribution Service for Real-Time Systems (DDS)
The Object Management Group (OMG) open international middleware standard directly addressing publish-subscribe communications for real-time and embedded systems.

Real-Time Publish-Subscribe DDS Interoperability Wire Protocol (DDS-I RTPS)
A protocol standardized by OGM for best effort and reliable pub-sub communications over unreliable transports in both unicast and multicast. ***RTPS*** will be used for short in this document.

Entity
Base class for all RTPS entities. RTPS ***Entity*** represents the class of objects that are visible to other RTPS Entities on the network. As such, RTPS ***Entity*** objects have a globally unique identifier (GUID) and can be referenced inside RTPS messages.

Endpoint
Specialization of RTPS ***Entity*** representing the objects that can be communication endpoints. That is, the objects that can be the sources or destinations of RTPS messages. Please refer to ***Data Writer*** and ***Data Reader***.

Participant
Container of all RTPS entities that share common properties and are located in a single address space.

Publisher
An object responsible for data distribution.

Data Writer
The object the application must use to communicate to a publisher the existence and value of data-objects of a given type.

Subscriber
An object responsible for receiving published data and making it available (according to the Subscriber's QoS) to the receiving application. It may receive and dispatch data of different specified types.

Data Reader
The object attached to subscriber to access the received data.

For more details of DDS terms and definitions, please refer to OMG DDS Specification[3] and OMG RTPS specification[4].

2 General Design

This chapter describes the general design of security services for DDS Security Plugins.

A design of security service implementation based on arm TrustZone[1] is also specified as an example of security services on arm platforms.

Section 2.3 shows a general implementation of security services for eProsima Fast-RTPS DDS built-in Security Plugins.

2.1 General Architecture

The architecture of DDS security service is showed in Figure 2-1 below.

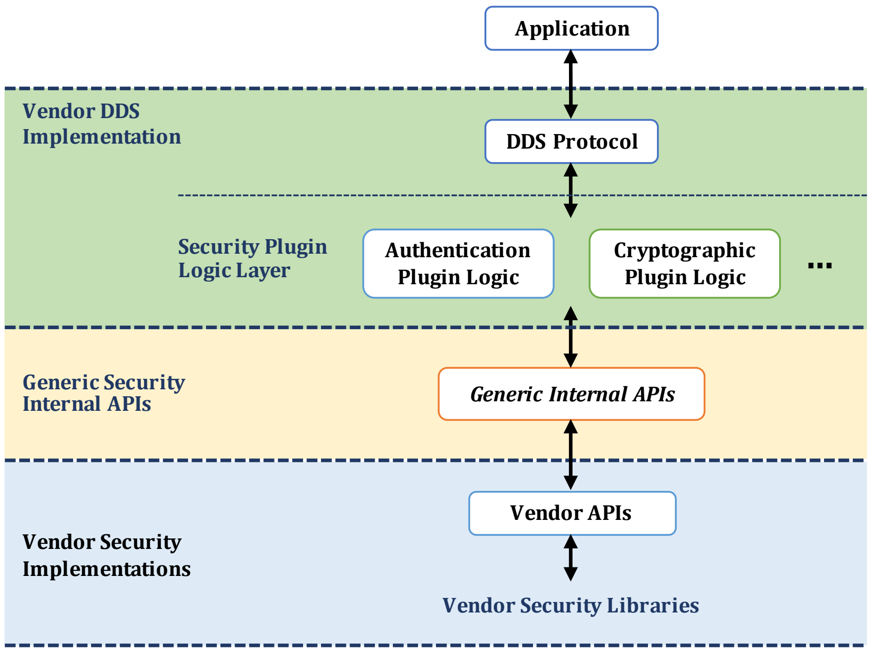
Application
Vendor DDS Implementation
DDS Protocol
Security Plugin Logic Layer
Authentication Plugin Logic
Cryptographic Plugin Logic
...
Generic Security Internal APIs
Generic Internal APIs
Vendor Security Implementations
Vendor APIs
Vendor Security Libraries

FIGURE 2-1 GENERAL HIERARCHY OF SECURITY SERVICES FOR DDS SECURITY PLUGINS

The generic internal APIs provide a generic security interface inside vendor DDS Security Plugin implementations, to wrap the vendor specific security support, such as arm TrustZone.

The underlying vendor security implementations complete the specific security operations and return the results.

2.2 General Design on Arm Platforms

The following sections specify general designs of the security services implementation for DDS Security Plugins on arm platforms. A simplified hierarchy is showed in Figure 2-2.

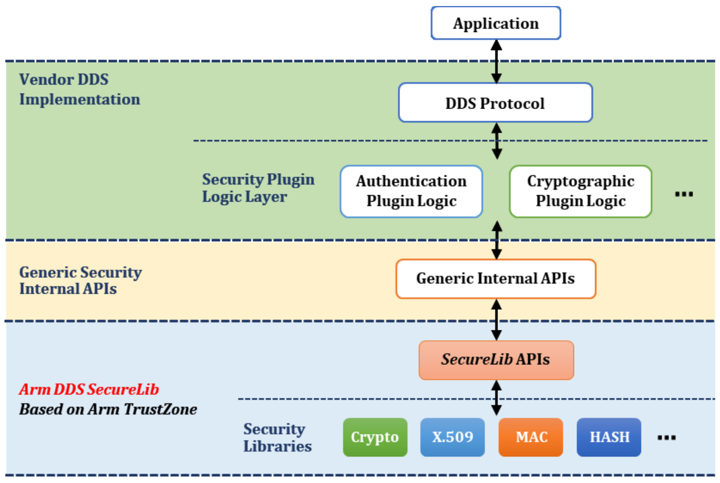

FIGURE 2-2 HIERARCHY OF DDS SECURITY SERVICES IMPLEMENTATION ON ARM PLATFORMS

As shown in Figure 2-2, Arm DDS *SecureLib* consists of a set of APIs and a group of *Security Libraries*.

Arm *SecureLib* APIs implement the generic internal APIs functionalities and communicate with Arm DDS *Security Libraries*.

The *Security Libraries* in Arm DDS *SecureLib* contain the implementations of security algorithms and operations required in Security Plugins logic. It includes the libraries for X.509 certification, key generation and management, encryption and decryption, etc.

The general block diagram of DDS security services implementation based on arm TrustZone[1] is showed in Figure 2-3 below. *SecureLib* APIs in Normal World trigger the *Security Libraries* in Secured World based on arm TrustZone. *Security Libraries* accomplish the specific security operations transferred from Normal World.

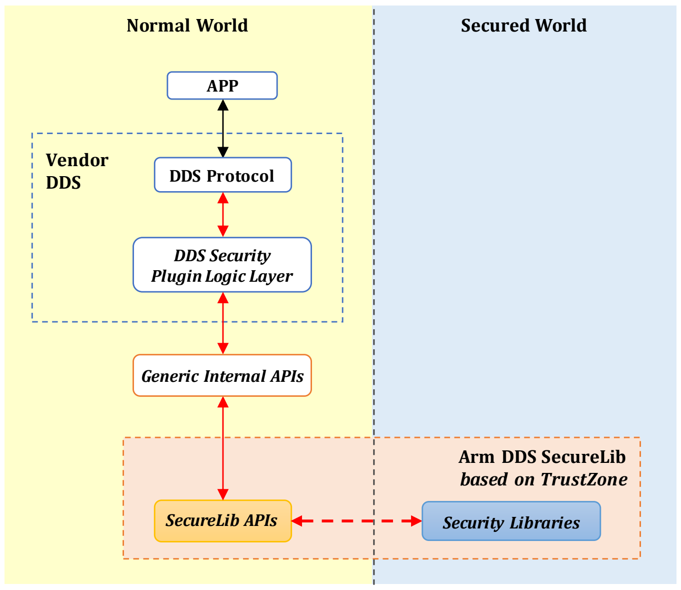

FIGURE 2-3 GENERAL BLOCK DIAGRAM OF IMPLEMENTATION ON ARM PLATFORMS

2.2.1 Block Diagram on Cortex-A Platforms

The block diagram of security services implementation on Cortex-A platform is showed in Figure 2-4 below.

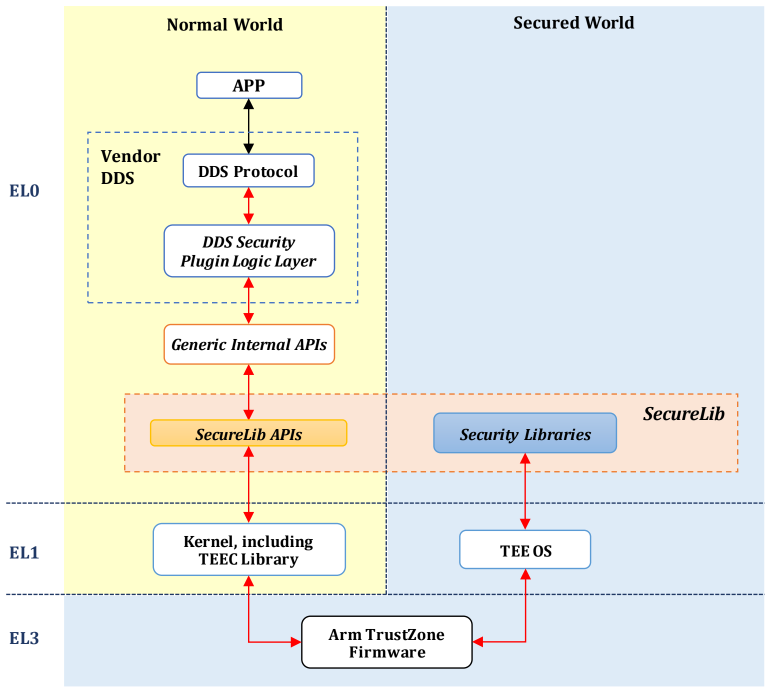

FIGURE 2-4 SECURITY SERVICES IMPLEMENTATION ON ARM CORTEX-A PLATFORMS

DDS Security Plugin Logic Layer invokes the generic internal APIs, which are implemented by *SecureLib* APIs in Normal World.

SecureLib APIs trigger *SecureLib* in Secured World to finally accomplishes the specific security task with TEE support. The communication routine includes TEEC library in Non-Secured World, Arm TrustZone and TEE OS in Secured World.

2.2.2 Block Diagram on Cortex-M Platforms

The simplified block diagram of security services implementation on Cortex-M platform is showed in Figure 2-5 below. Secure firmware might directly handle the communication between Non-Secured World and Secured World on Cortex-M, instead of cooperating with TEE. The actual implementation details might vary based on use cases.

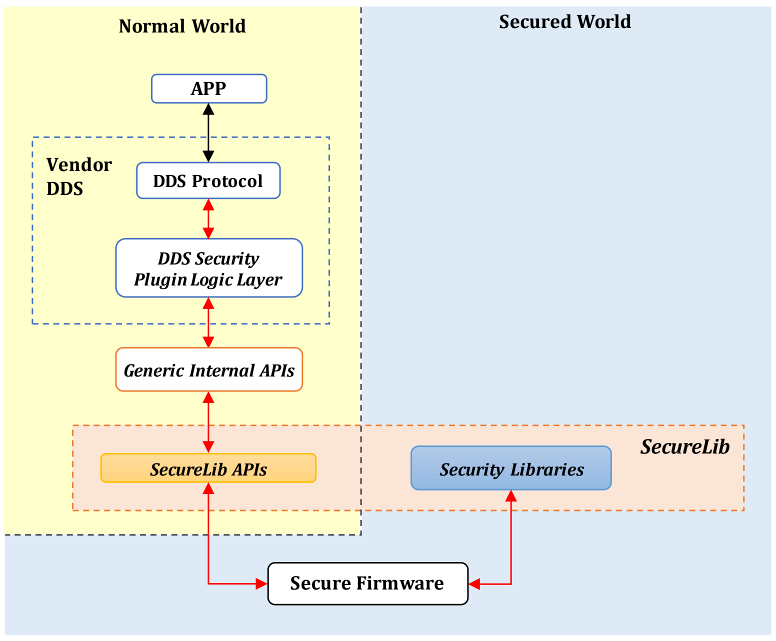

FIGURE 2-5 SECURITY SERVICES IMPLEMENTATION ON ARM CORTEX-M PLATFORMS

2.3 General Implementation in Fast-RTPS (Non-Normative)

This section demonstrates a general implementation of the security services for built-in DDS Security Plugins in eProsima Fast-RTPS[10][11][12]. All the designs showed in this section are informative. The details in implementation relies on actual development and requirements.

The general internal APIs should replace the hard-coded OpenSSL APIs in Fast-RPTS DDS Security Plugins.

A general hierarchy of the implementation in Fast-RTPS is showed in Figure 2-6 below, which takes built-in Authentication Plugin *DDS:Auth:PKI-DH* as an example. As shown in Figure 2-6, Fast-RTPS *DDS:Auth:PKI-DH* plugin calls generic internal APIs under folder *ops*. Vendor security implementation, such as OpenSSL and OP-TEE based on arm TrustZone can provide its own implementation of the generic internal APIs, based on their own specific security libraries. The special underlying security implementation can be selected and configured during compiling. Header file *internal_security_api.h* re-directs to the security implementation specific data types and function definitions.

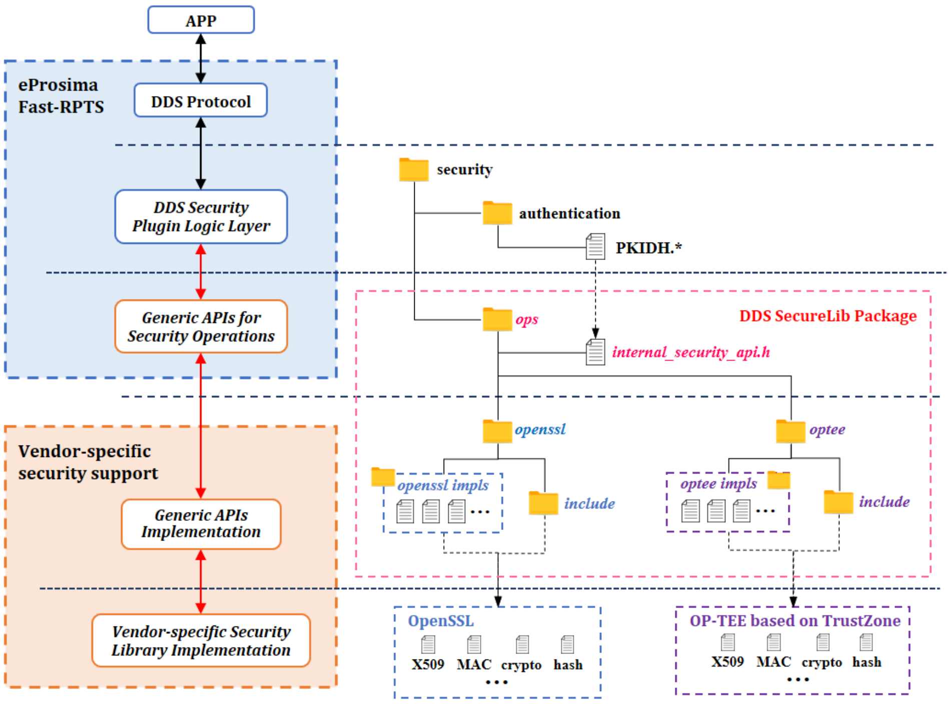

FIGURE 2-6 GENERAL IMPLEMENTATION IN EPROSIMA FAST-RTPS

In practice, the procedure of an operation in OpenSSL might not fit to that in arm TrustZone, and vice versa. Thus, it is difficult to bring up a generic API for that operation. For example, arm TrustZone and OpenSSL have different key generation and management.

To keep the interface generic, some helper functions in which TrustZone and OpenSSL cannot share the same sequence can be extracted out. Those helper functions can be adjusted to be generic as the internal APIs are defined. The helper functions can be implemented by vendor specific security implementation. The implementation is showed in Figure 2-7.

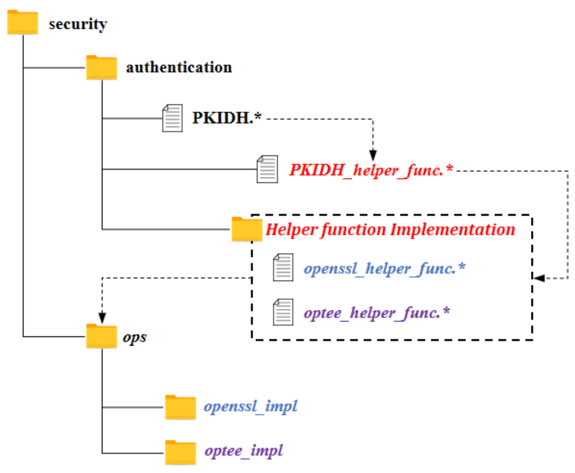

FIGURE 2-7 HELPER FUNCTIONS IMPLEMENTATION

3 DDS Security Plugins Generic Internal APIs

DDS Security Plugins generic internal APIs provide DDS Security Plugins logic with a generic interface in Non-Secure World, to the specific security libraries implementation, such as OpenSSL and Arm DDS *SecureLib*.

This set of generic internal APIs wraps the vendor security APIs which finally invoke the specific vendor security libraries. Through the generic internal APIs layer, the DDS Security Plugin logic, such as eProsima Fast-RTPS[10][11][12] and RTI Connext DDS[13], doesn't need to care about details of the underlying vendor security libraries implementation. The generic internal APIs help vendor DDS Security Plugin logic become more compatible with diverse vendor platforms.

A model of the above procedure is showed in Figure 3-1.

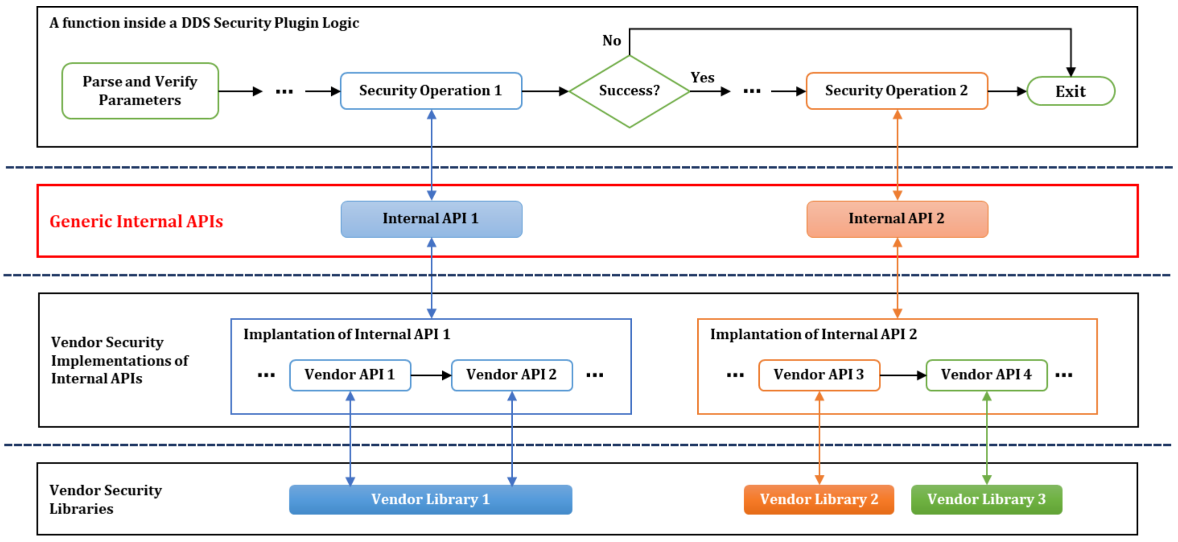
A function inside a DDS Security Plugin Logic
Parse and Verify Parameters
Security Operation 1
Success?
No
Yes
Security Operation 2
Exit
Generic Internal APIs
Internal API 1
Internal API 2
Vendor Security Implementations of Internal APIs
Implantation of Internal API 1
Vendor API 1
Vendor API 2
Implantation of Internal API 2
Vendor API 3
Vendor API 4
Vendor Security Libraries
Vendor Library 1
Vendor Library 2
Vendor Library 3

FIGURE 3-1 EXAMPLE OF A FUNCTIONALITY INSIDE LOGIC MODULE

The generic internal APIs specification[14] defines data types and generic functions in those APIs. The specific implementation of the generic internal APIs relies on the vendor security implementations.

3.1 Support to DDS Security Plugins

The following sections specify the generic internal APIs support to the DDS Security Plugins. The generic internal APIs may not provide entire APIs to all kinds of operations in all vendor DDS Security Plugin implementations. Instead, the generic internal APIs may only support a set of common and significant security functionalities.

3.1.1 Built-in DDS Security Plugins

The generic internal APIs should at least support built-in Authentication Plugin `DDS:Auth:PKI-DH` and Cryptographic Plugin `DDS:Crypto:AES-GCM-GMA` as defined in DDS Security specification[5].

The support to other built-in DDS Security Plugins might be added in the future when the vendor implementation of those plugins is available and appropriate license is provided.

3.1.2 Vendor Specific DDS Security Plugins

It is not required to support vendor specific DDS Security Plugins.

4 Arm DDS SecureLib

Arm DDS *SecureLib* consists of its specific APIs and *Security Libraries*.

Those APIs are used to implement the functionalities of the generic internal APIs in Normal World. They communicate with *Security Libraries* to transfer the security tasks and the results.

The *Security Libraries* in Secure World group the security algorithms and operations, such as authentication, key generation and management, encryption and decryption, etc.

The simplified hierarchy is showed in Figure 4-1.

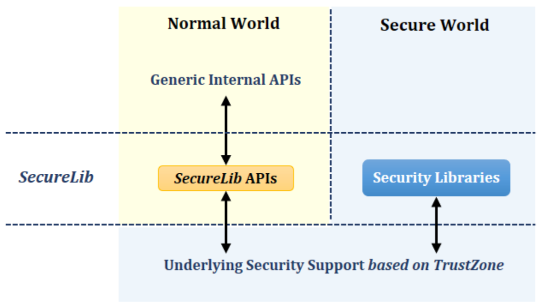

FIGURE 4-1 ARM DDS *SECURELIB*

4.1 Secure Library APIs

Security library should export its APIs in Normal Word. The details of the APIs are defined in Arm DDS *SecureLib* specific API documents.

Those APIs might preprocess the requests from DDS Security Plugin generic internal APIs, such as adjusting and organizing the parameter to fit the format in interfaces.

4.2 Secure Library Implementation

Secure libraries implement the specific security task in Secured World. Those libraries should at least support the following functionalities.

- X509 Certificate, including certificate verification and key validation
- SHA-256 Digest
- Diffie-Hellman key pair generation
- Asymmetric digest signature and verification
- Key derivation with DH keys.
- Hash-based Message Authentication Code (MAC)
- Authentication encryption and decryption with AES-128-GCM/GMAC and AES-256-GCM/GMAC.
- Random number generation
- Support to large multi-precision integers, including the convert between large integers and octet strings.

The specific implementations and algorithms in above functionalities should satisfy the requirement in DDS Security specification[5].

On Cortex-A platforms, libraries communicate with TEE-OS based on arm TrustZone through a generic interface which is compatible to Global Platform TEE specification. The implementation should not rely on any specific vendor TEE-OS.

On Cortex-M platforms, libraries might directly interact with arm secure firmware.

5 Requirements to Implementations on Arm Platforms

The following sections specify some requirements to the implementations of security services for DDS Security Plugins based on arm TrustZone.

5.1 Protection of Root Security Assets

In current DDS framework, some root security assets, including root keys, permissions and certificates are all stored in rich Operation System environment and DDS property fields, which are all in Normal World. As a result, the root security assets are vulnerable.

In DDS security services implementation based on arm TrustZone, the root security assets can be stored and protected in physically isolated Secure World based on arm TrustZone. Normal World should be forbidden to directly access those assets in Secure World. Instead, Security Plugins logic modules in Normal World must invoke DDS *SecureLib* to access them in Secure World. Such a feature supported by arm TrustZone can avoid leakage of root security assets, especially the root private key, even if the rich Operation System environment is hacked.

The provision and installation of the security assets in Secure World are platform specific and are out of the scope of the design document.

The *Security Libraries* for storage and management of the security assets in Secure World should be implemented based on arm TrustZone. Specific parameters might be defined in Arm DDS *SecureLib* APIs to enhance the authentication and access control processes while accessing root security assets. The details depend on the actual implementation and use cases.

5.2 Performance (Non-Normative)

The implementation might follow the requirements listed in DDS Security specification[5] Section 9.2.1.

The implementation should avoid a serious impact to DDS data transfer performance, including throughput and delay. The quantitative requirements might be determined based on implementations and use cases.

The implementation might support hardware security module, such as such as hardware Crypto IP, in implementation to enhance the performance.

5.3 Compatibility

Some interfaces should try to avoid coupling to a specific vendor implementation.

- The generic internal APIs called in DDS Security Plugins should provide a generic interface, instead of a special wrapper of some vendor specific APIs.
- The Arm DDS *SecureLib* APIs should be compatible with GlobalPlatform TEE specification, rather than with a specific vendor TEE implementation.
- On Cortex-A platforms, the interface between DDS *SecureLib* and TEE should be compatible with GlobalPlatform TEE specification, rather than with a specific vendor TEE implementation.