OCI | WE ARE SOFTWARE ENGINEERS.

Revolutionizing Data Distribution with an Open and Secure DDS

OpenDDS®

August 22, 2018

WHAT IS OpenDDS?

OpenDDS is an open source and widely adopted standards-based real-time publish/subscribe solution for distributed systems.

- opendds.org
- GitHub: https://github.com/objectcomputing/OpenDDS

DEVELOPING APPLICATIONS WITH OpenDDS

Developers use OpenDDS as a framework for enabling C++ and Java applications to distribute data over the network using a publish/subscribe architecture.

Unlike low-level transport protocols, the OpenDDS middleware is aware of the schema and semantics of the data. OpenDDS helps shield application developers from the inherent complexities of distributed computing.

HOW DOES DDS FIT IN THE APPLICATION ARCHITECTURE?

The Industrial Internet Consortium (www.iiconsortium.org) Connectivity Framework defines a stack model consisting of multiple layers.

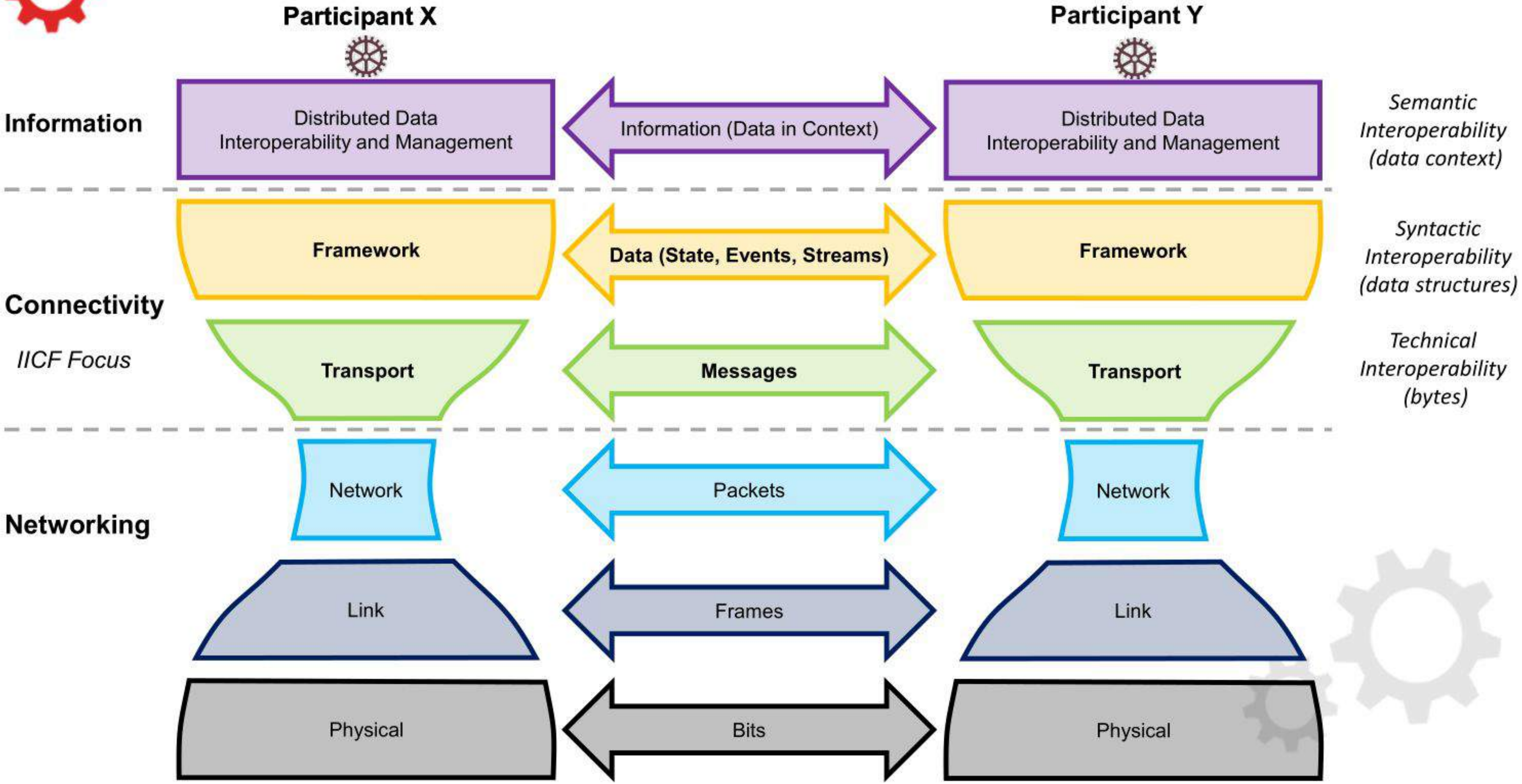
The DDS API sits at the Framework Layer, providing Syntactic Interoperability among heterogeneous systems. DDS products also include an interoperable standards-based Transport Layer providing Technical Interoperability.

Many alternatives to DDS provide only the Transport Layer, requiring each application to provide its own solutions to the concerns of the Framework Layer.

*diagrams on the following slides featuring this red gear icon are from the IIC

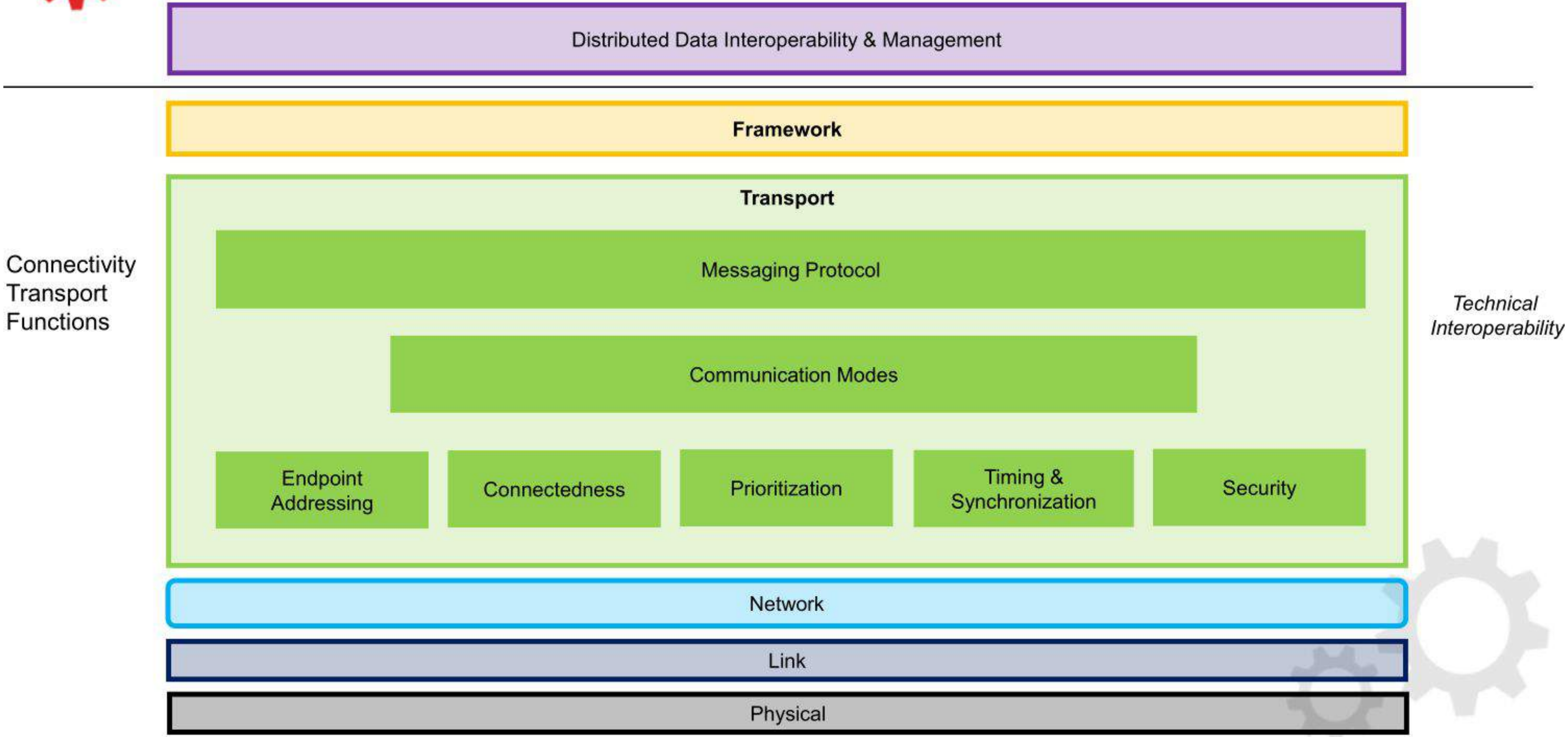
industrial internet® CONSORTIUM

IIoT Connectivity Stack Model

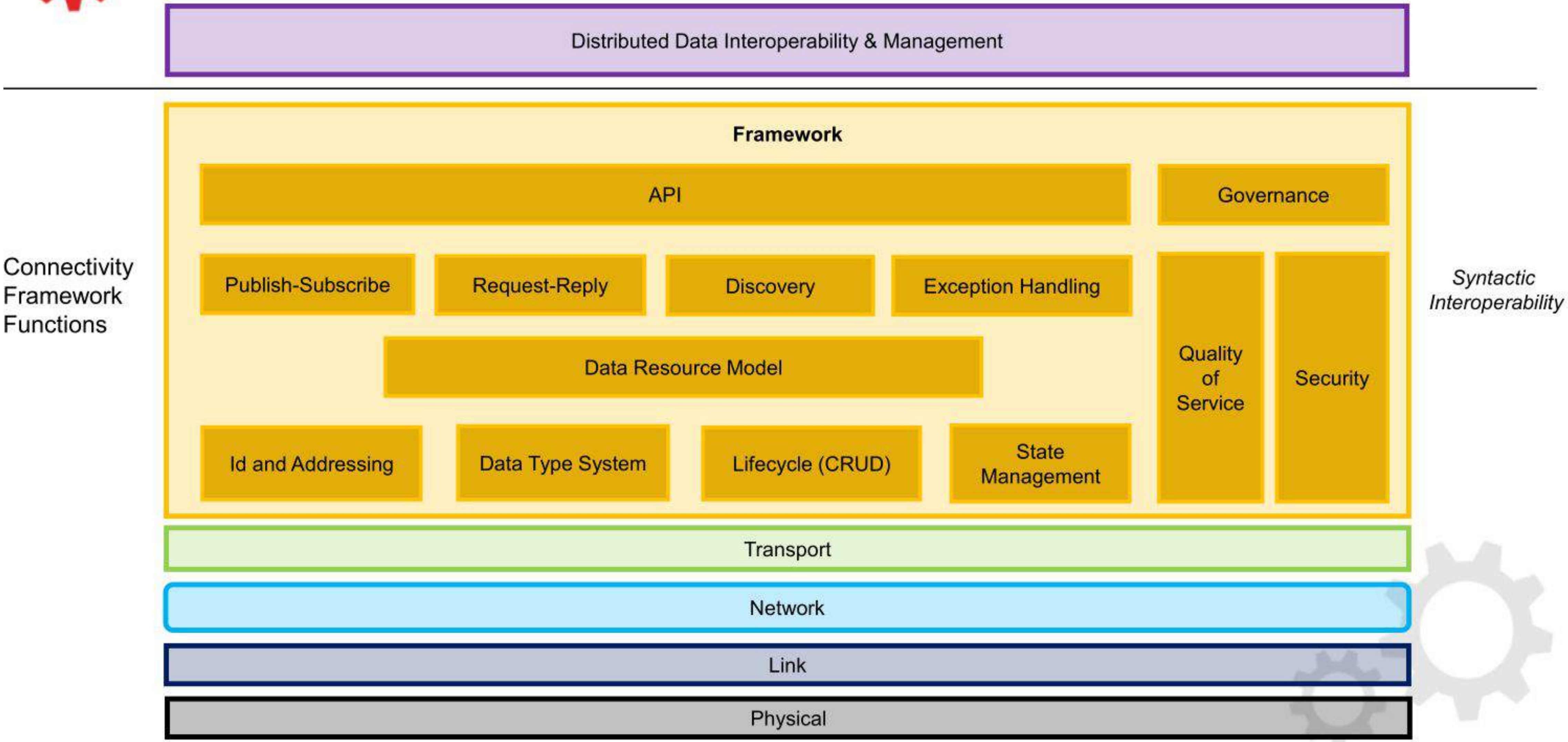


Participant X
Participant Y
Information
Distributed Data Interoperability and Management
Information (Data in Context)
Distributed Data Interoperability and Management
Semantic Interoperability (data context)
Connectivity
IICF Focus
Framework
Data (State, Events, Streams)
Framework
Syntactic Interoperability (data structures)
Transport
Messages
Transport
Technical Interoperability (bytes)
Networking
Network
Packets
Network
Link
Frames
Link
Physical
Bits
Physical

Connectivity Transport Layer



Connectivity Framework Layer



Distributed Data Interoperability & Management

Connectivity Framework Functions

Framework

API

Governance

Publish-Subscribe

Request-Reply

Discovery

Exception Handling

Data Resource Model

Id and Addressing

Data Type System

Lifecycle (CRUD)

State Management

Quality of Service

Security

Syntactic Interoperability

Transport

Network

Link

Physical

OpenDDS AND OMG DDS

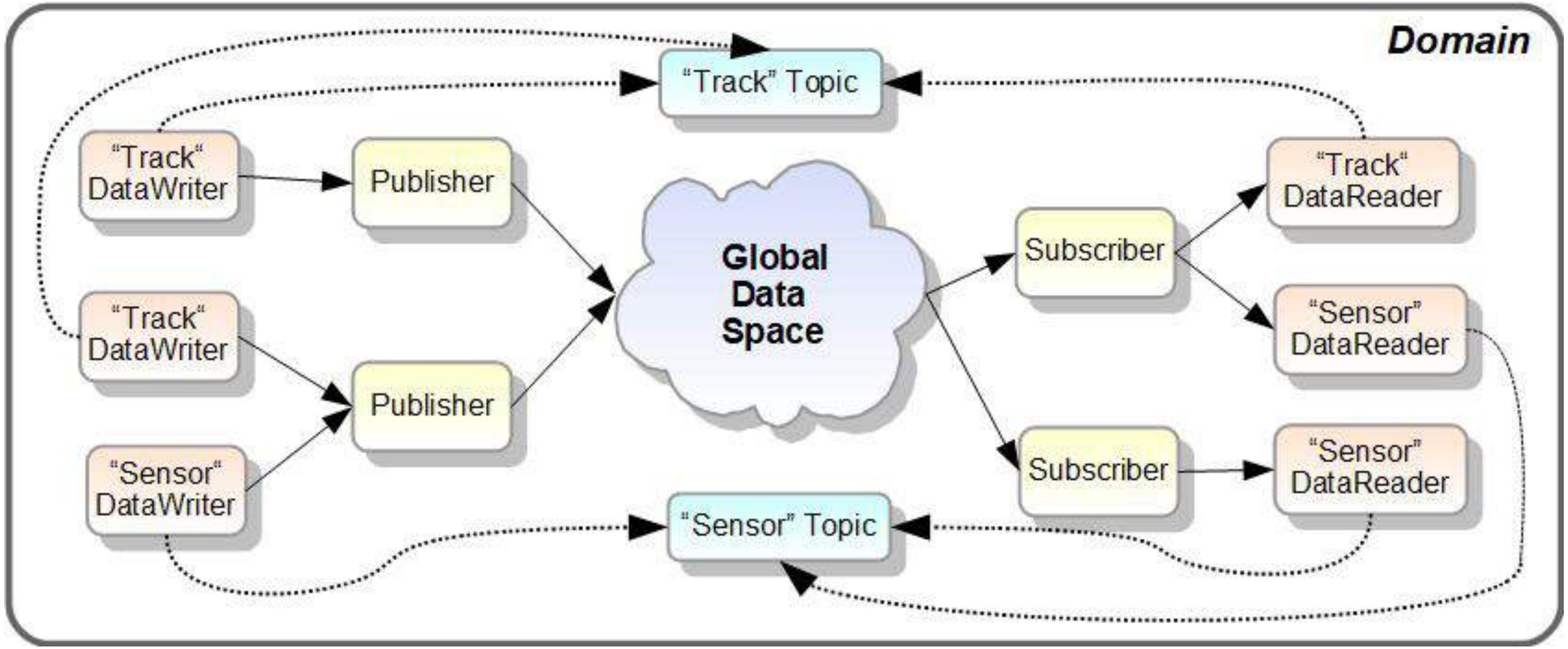
OpenDDS is OCI's implementation of the Object Management Group's standard Data Distribution Service (DDS). The DDS spec describes the API and its semantics, but not how to exchange data over the network.

Interoperability with other DDS products is made possible through the OMG's DDS-RTPS specification which defines a standardized discovery mechanism and wire protocol for DDS.

OpenDDS AS A SOFTWARE PROJECT

- Development started in 2005 by OCI
- Version 1.0 (debut of the name OpenDDS): July 2007
- > 15K commits in git: OCI and community contributors
- GitHub repository hosts active development, not snapshots
- CI builds automate testing on many platforms/compilers
- Community support: GitHub-hosted site (opendds.org), Issues and Pull Requests, plus SourceForge mailing lists
- Commercial support, custom development, and training provided by OCI

REVIEW OF DDS TERMS AND CONCEPTS



DDS SECURITY OVERVIEW

DDS applications share data across the network. Without DDS Security, this data is sent “in the clear.” This makes traditional DDS applicable to closed networks or to networks that provide security outside the application space (for example, using VPNs).

The DDS Security specification uses industry-standard cryptographic algorithms and techniques to protect DDS applications from specific threats: unauthorized publication/subscription, tampering/replay, and unauthorized access to data.

Architecturally this is similar to HTTPS, however DDS-RTPS+Security runs over UDP/IP and supports multicast (including encryption).

DDS-Security : DDS-RTPS :: HTTPS : HTTP

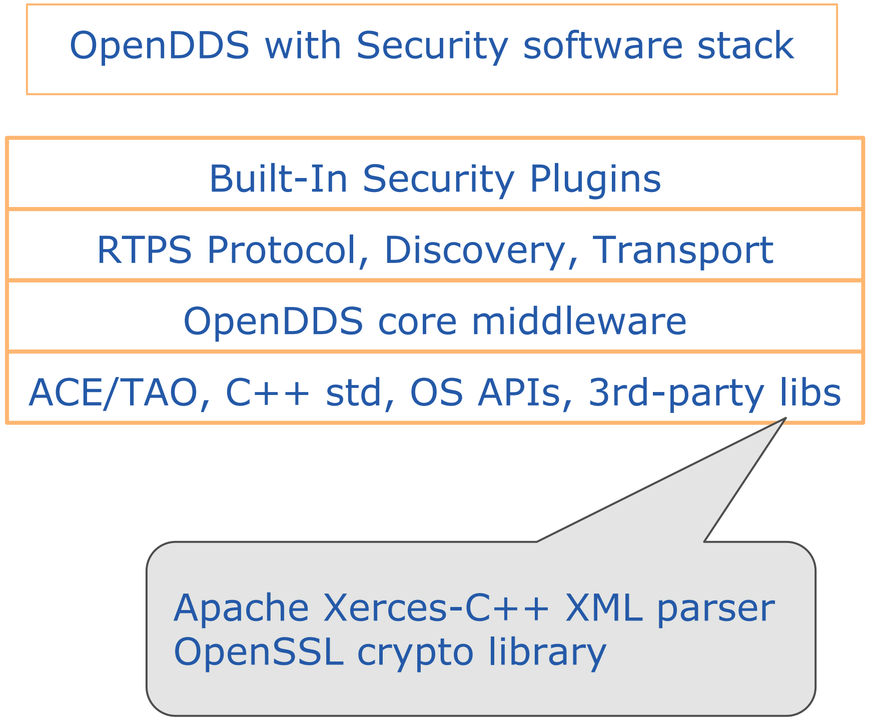
DDS SECURITY ARCHITECTURE

DDS Security spec defines plugin APIs for Authentication, Access Control, and Cryptographic operations.

The spec also defines Built-In implementations of these plugins.

OpenDDS includes these implementations, but developers could also write their own.

This presentation only uses Built-In Plugins.



OpenDDS with Security software stack

Built-In Security Plugins

RTPS Protocol, Discovery, Transport

OpenDDS core middleware

ACE/TAO, C++ std, OS APIs, 3rd-party libs

Apache Xerces-C++ XML parser
OpenSSL crypto library

DDS SECURITY SPECIFICATION

Object Management Group’s specification for DDS Security includes:

- Authentication of Participating Applications
  - Application identities determined by certificates signed by a common CA
- Access Control by Topic
  - Configuration files (signed by CA) determine which applications have access (read/write/both) to which topics
- Data Protection via Encryption and/or Message Authentication
  - Topic-by-topic configuration determines whether to encrypt or only sign network messages
  - Scope of data protection is also configurable: payload only or including headers

BUILT-IN PLUGIN DETAILS

- Authentication: PKI-DH
  - Public-Key Infrastructure: uses a trusted Identity Certificate Authority
  - Identity principal is the DDS Domain Participant (each has a Certificate)
  - Authentication and key agreement using Digital Signatures and Diffie-Hellman
- Access Control: Governance and Permissions
  - XML documents, schema defined by DDS Security
  - Signed by the Permissions CA (may be the same as Identity CA)
  - Governance: common across the domain   Permissions: for each participant
- Cryptographic: AES-GCM-GMAC, 256-bit keys (128-bit option)
  - Advanced Encryption Standard Galois Counter Mode / Message Authentication Codes

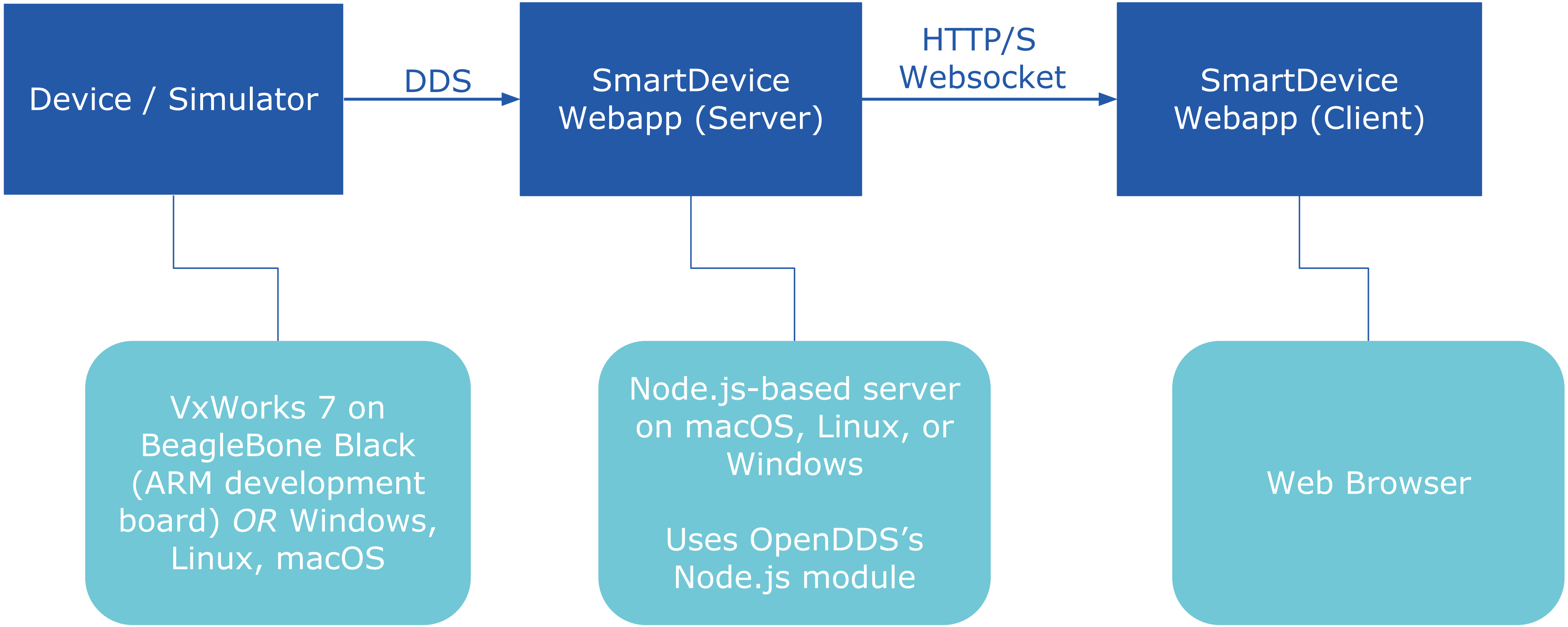
OpenDDS SECURITY BETA IMPLEMENTATION

- OpenDDS 3.13 implements the 3 required Built-in Plugins
- Some options are not yet implemented:
  - Origin authentication
  - 128-bit AES
  - Relay permissions
  - Signing/encrypting at the full-message level
    - Sub-messages may be signed or encrypted; payloads may be encrypted
- Details in the "Using DDS Security in OpenDDS" document and later in this presentation

SECURITY USE CASE: INDUSTRIAL INTERNET OF THINGS

- Network-connected sensors measure valve pressure
- Web application collects and displays sensor readings
- Security goals:
  - Prevent imposter sensors/values from appearing in the UI
  - Keep operational data confidential

DEMO SOFTWARE COMPONENTS


Device / Simulator

DDS

SmartDevice Webapp (Server)

HTTP/S Websocket

SmartDevice Webapp (Client)

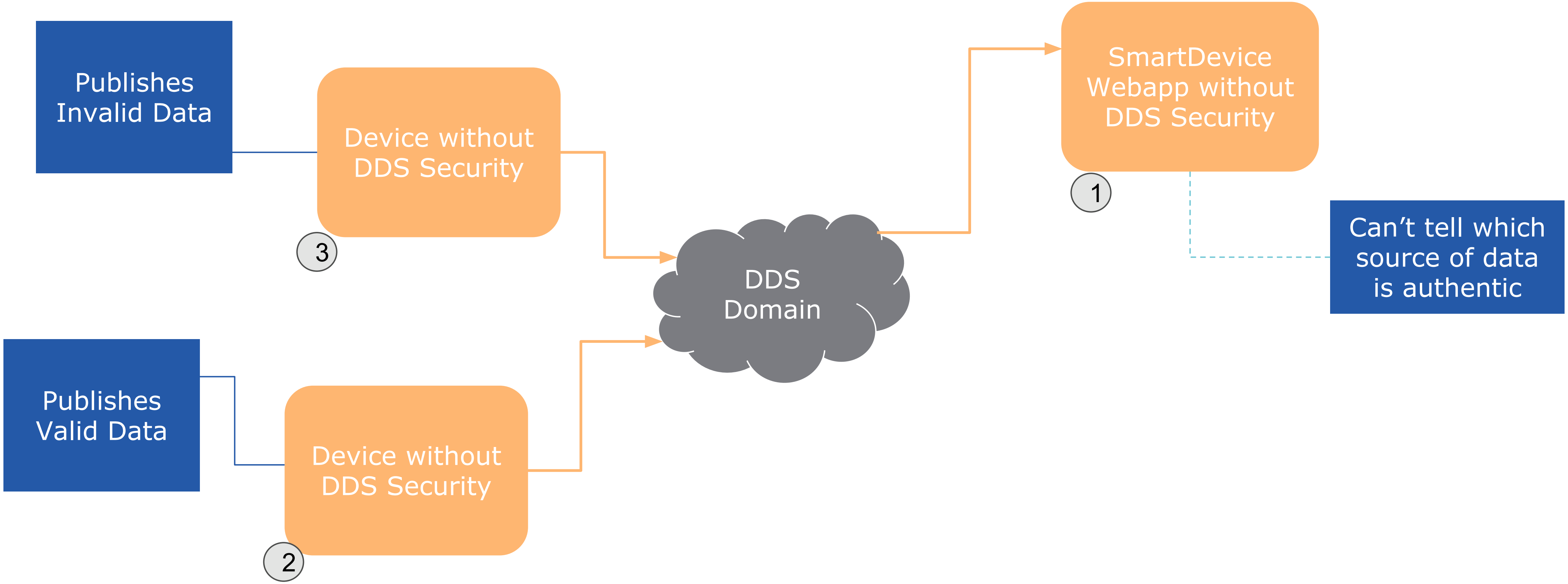
VxWorks 7 on BeagleBone Black (ARM development board) *OR* Windows, Linux, macOS

Node.js-based server on macOS, Linux, or Windows

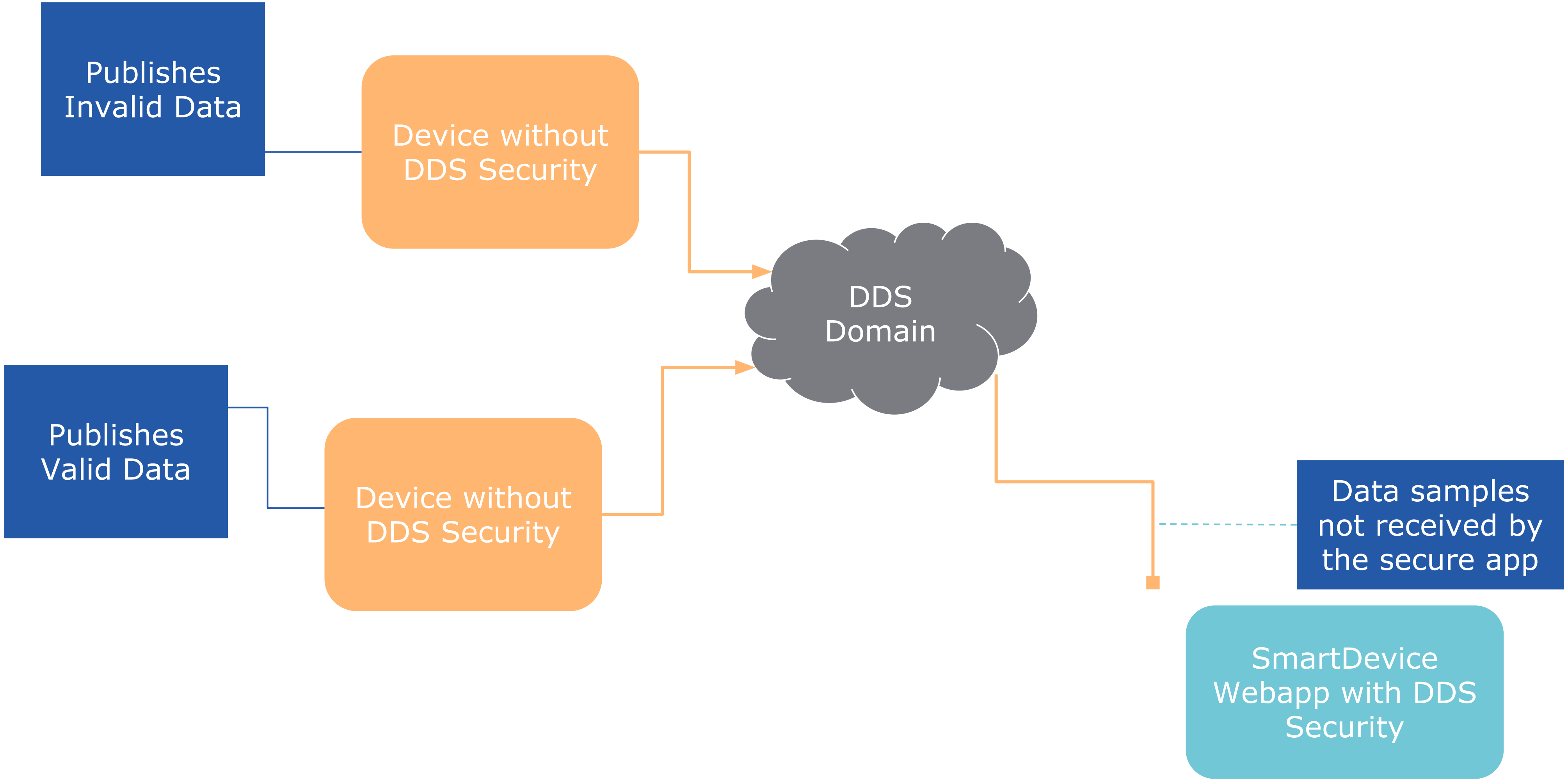
Uses OpenDDS's Node.js module

Web Browser

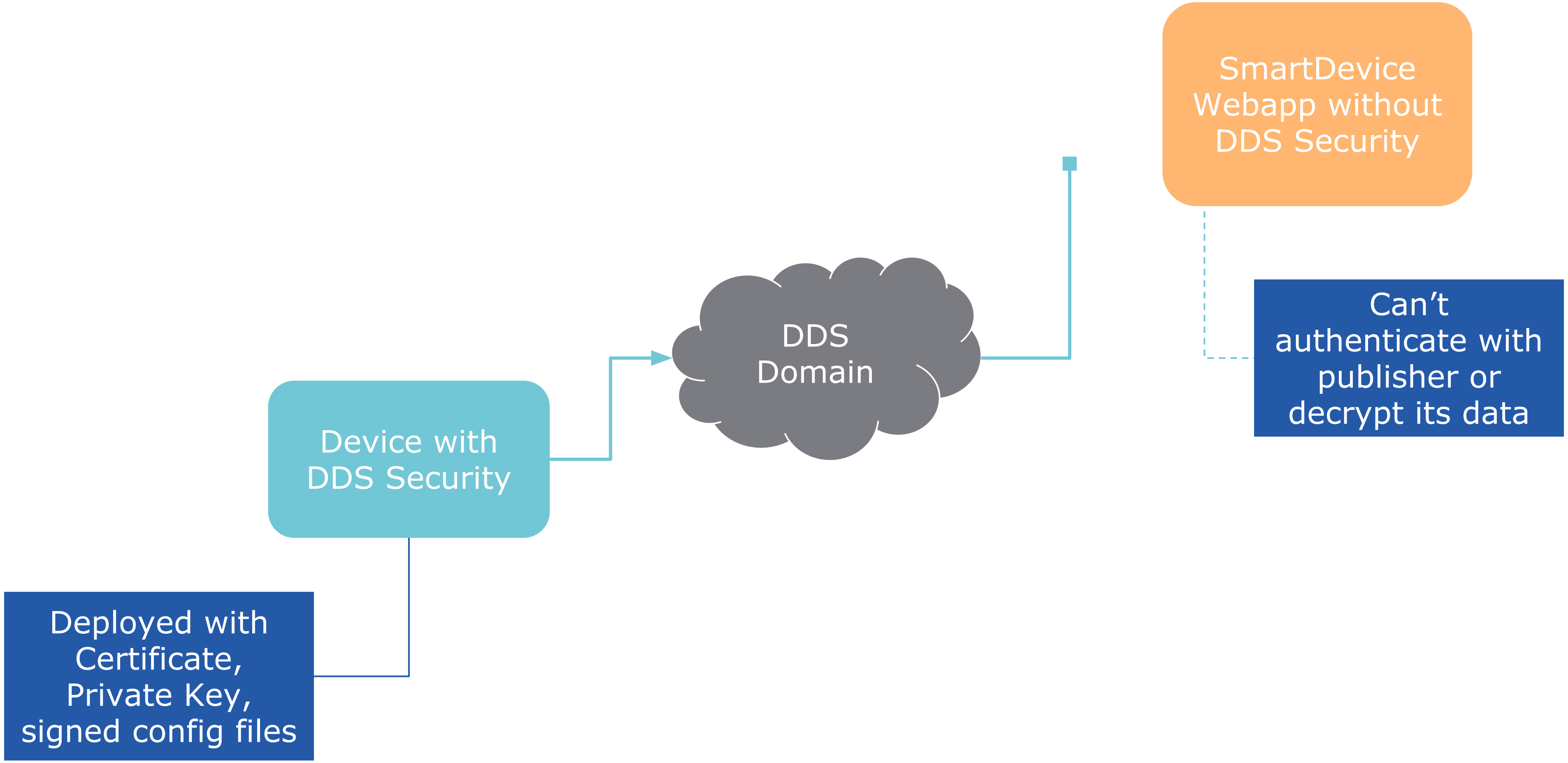
STEP 1: BEFORE DDS SECURITY


Publishes Invalid Data
Device without DDS Security
3
SmartDevice Webapp without DDS Security
1
Can’t tell which source of data is authentic
DDS Domain
Publishes Valid Data
Device without DDS Security
2

STEP 2: INSECURE PUBLISHERS / SECURE SUBSCRIBER


Publishes Invalid Data
Device without DDS Security
DDS Domain
Publishes Valid Data
Device without DDS Security
Data samples not received by the secure app
SmartDevice Webapp with DDS Security

STEP 3: SECURE PUBLISHER / INSECURE SUBSCRIBER



SmartDevice Webapp without DDS Security

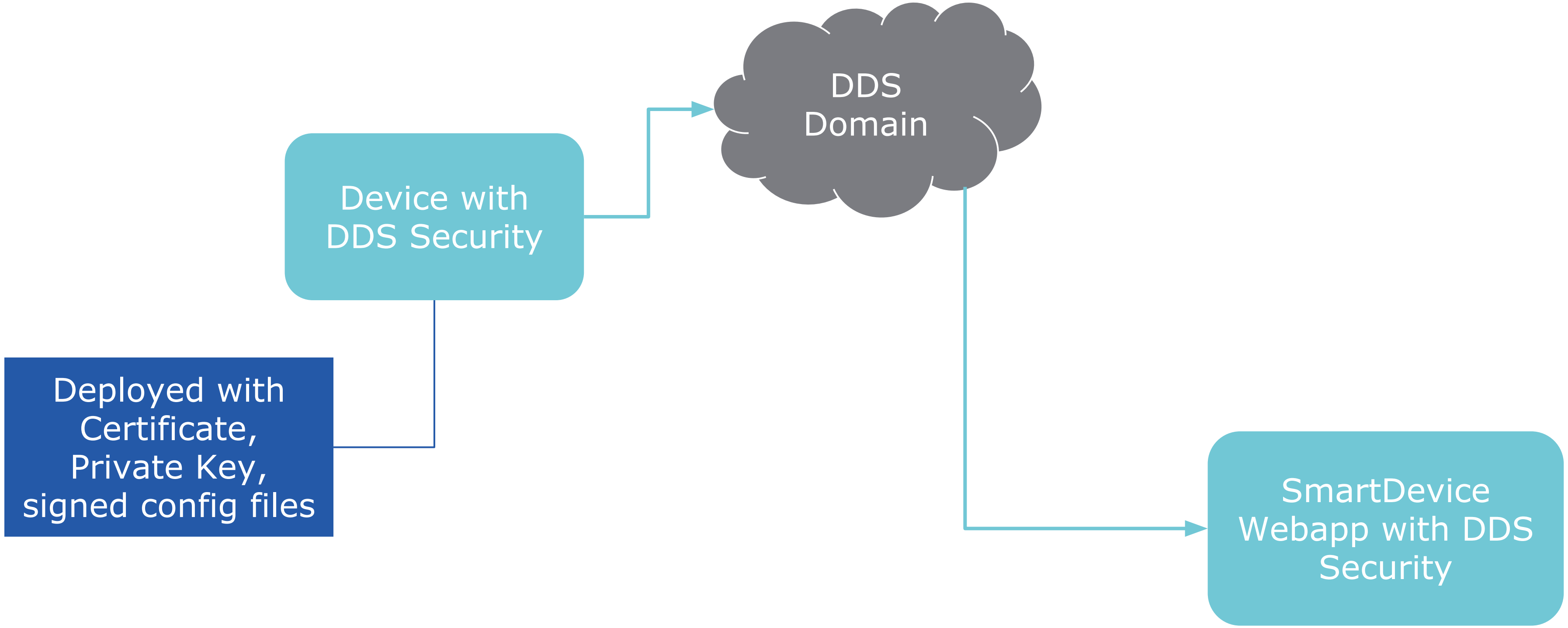
Can't authenticate with publisher or decrypt its data

DDS Domain

Device with DDS Security

Deployed with Certificate, Private Key, signed config files

STEP 4: SECURE PUBLISHER AND SUBSCRIBER



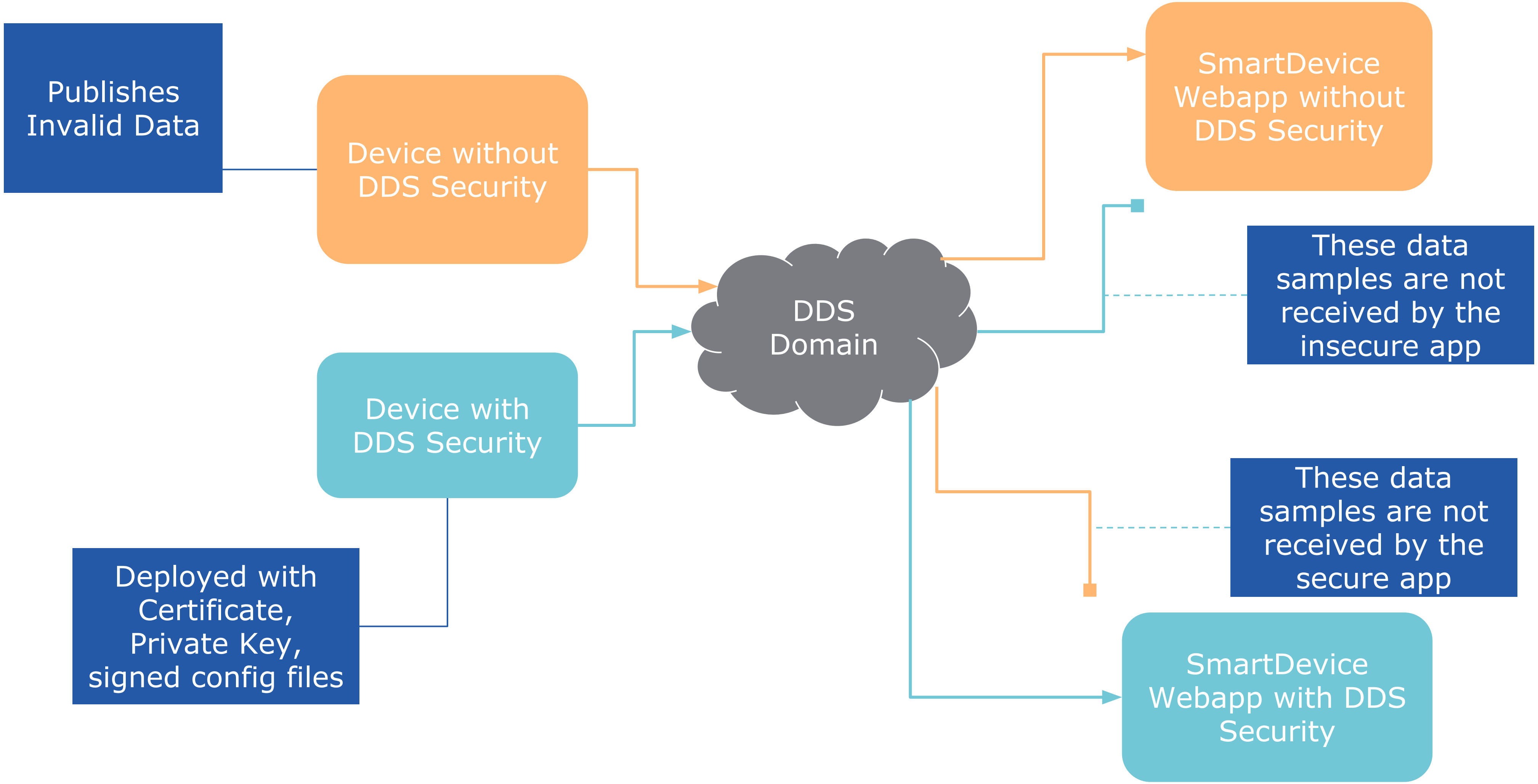
DDS Domain

Device with DDS Security

Deployed with Certificate, Private Key, signed config files

SmartDevice Webapp with DDS Security

STEP 5: SECURE AND INSECURE PARTICIPANTS



Publishes Invalid Data

Device without DDS Security

Device with DDS Security

Deployed with Certificate, Private Key, signed config files

DDS Domain

SmartDevice Webapp without DDS Security

These data samples are not received by the insecure app

These data samples are not received by the secure app

SmartDevice Webapp with DDS Security

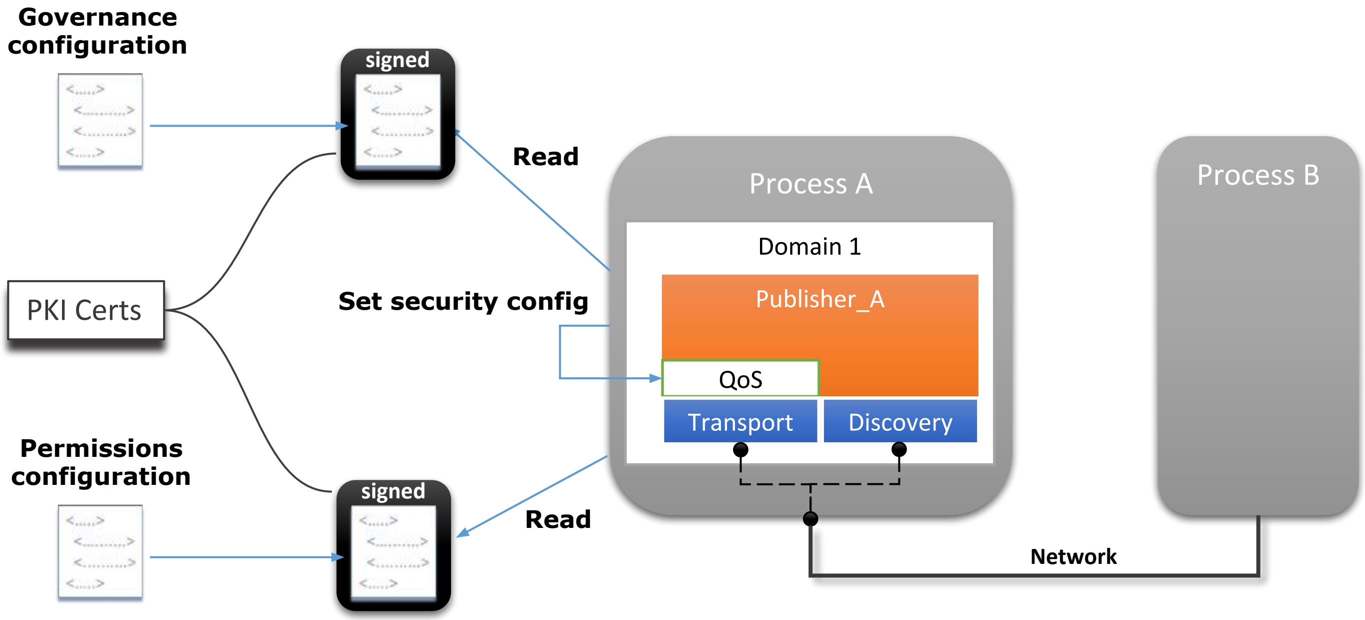
SECURING AN OpenDDS C++ APPLICATION

```
TheServiceParticipant->set_security(true); // or use .ini file
DDS::DomainParticipantQos qos;
factory->get_default_participant_qos(qos);
DDS::PropertySeq& props = qos.property.value;
append(props, "dds.sec.auth.identity_ca", "file:identity_ca_cert.pem"); // append() is a helper function
append(props, "dds.sec.access.permissions_ca", "file:permissions_ca_cert.pem");
append(props, "dds.sec.auth.identity_certificate", "file:test_participant_01_cert.pem");
append(props, "dds.sec.auth.private_key", "file:test_participant_01_private_key.pem");
append(props, "dds.sec.access.governance", "file:governance_signed.p7s");
append(props, "dds.sec.access.permissions", "file:permissions_1_signed.p7s");
DDS::DomainParticipant_var participant = factory->create_participant(DOMAIN_ID, qos, 0);
```

WHERE DO THESE FILES COME FROM?

- `identity_ca_cert.pem`
  - Certificate Authority for Identity: can be an existing CA or one created for the application
- `permissions_ca_cert.pem`
  - Certificate Authority for Permissions: can be the same as identity_ca_cert.pem or distinct
  - A distinct CA allows this private key to sign XML but not issue new Identity Certificates
- `test_participant_01_cert.pem`
  - Certificate issued by Identity CA that identifies this Domain Participant
- `test_participant_01_private_key.pem`
  - Private Key for the Certificate above
- `governance_signed.p7s`
  - Governance rules common to all participants in the domain
  - XML document signed by the Permissions CA
- `permissions_1_signed.p7s`
  - Permissions for this particular participant, contains its Subject Name (matches its Certificate)
  - XML document signed by the Permissions CA

XML FILES FOR SECURITY CONFIGURATION



GOVERNANCE XML

```
<domains><id>23</id></domains>
<allow_unauthenticated_participants>false</allow_unauthenticated_participants>
<enable_join_access_control>false</enable_join_access_control>
<discovery_protection_kind>NONE</discovery_protection_kind>
<topic_access_rules>
  <topic_rule>
    <topic_expression>*</topic_expression>
    <enable_discovery_protection>false</enable_discovery_protection>
    <metadata_protection_kind>ENCRYPT</metadata_protection_kind>
    <data_protection_kind>NONE</data_protection_kind>
  </topic_rule>
</topic_access_rules>
```

Optional secure discovery

If used, secure discovery can be set per-topic

Per-topic configuration of Cryptographic plugin

Not a complete Governance file

PERMISSIONS XML

```
<permissions>
  <grant name="Permission">
    <subject_name>emailAddress=cto@acme.com, CN=DDS Shapes Demo, OU=CTO Office, O=ACME Inc., L=Sunnyvale,
ST=CA, C=US</subject_name>
    <validity>
      <not_before>2015-10-26T00:00:00</not_before><not_after>2020-10-26T22:45:30</not_after>
    </validity>
    <allow_rule>
      <domains><id>23</id></domains>
      <publish>
        <topics><topic>*</topic></topics>
      </publish>
    </allow_rule>
    <default>DENY</default>
  </grant>
</permissions>
```

Matches subject of identity certificate

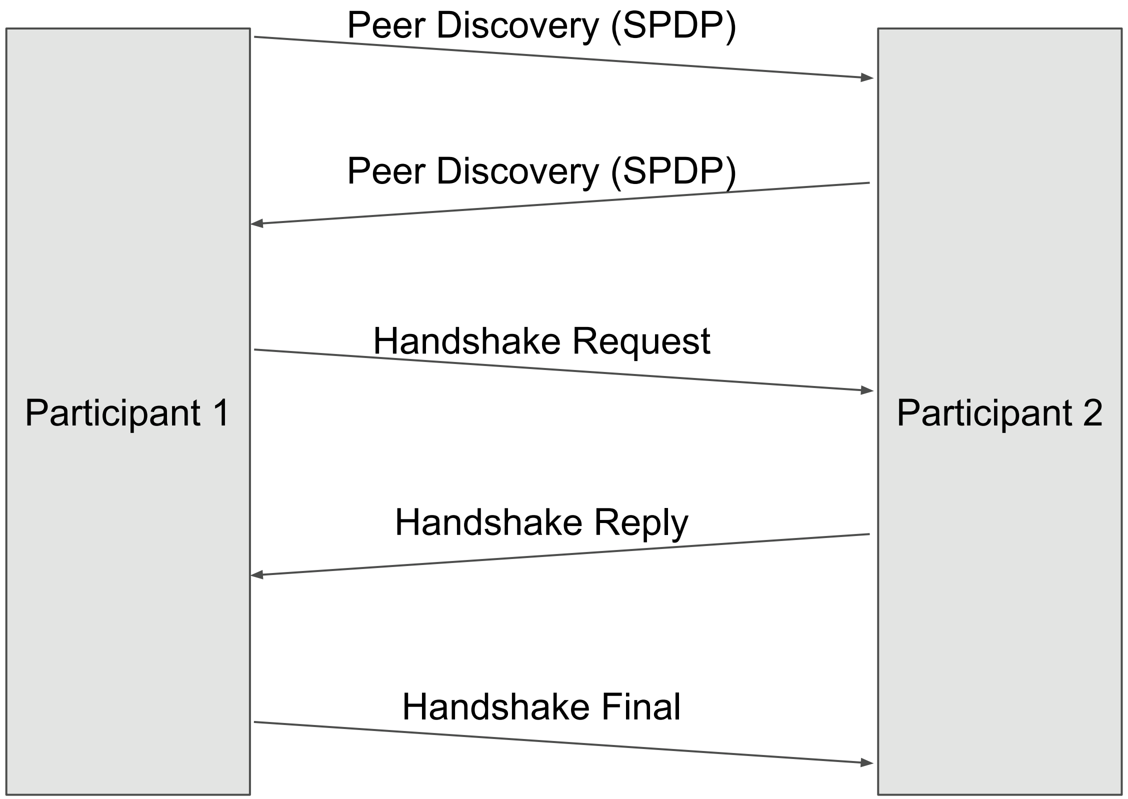
This participant can publish (create a DataWriter for) any topic in domain 23

Rules are checked in order of appearance in the file.
Default used if no rule matches the access control check.

Not a complete Permissions file

AUTHENTICATION PLUGIN IN ACTION

- Each pair of security-enabled participants completes a 3-way handshake
- Certificates and Permissions exchanged
- Digital Signatures verified
- DH key agreement



ACCESS CONTROL PLUGIN IN ACTION

- Sets up domain-wide configuration (Governance)
- Checks that actions of the local participant are allowed
- Enforces permissions of remote participants
  - Unauthorized entities ignored

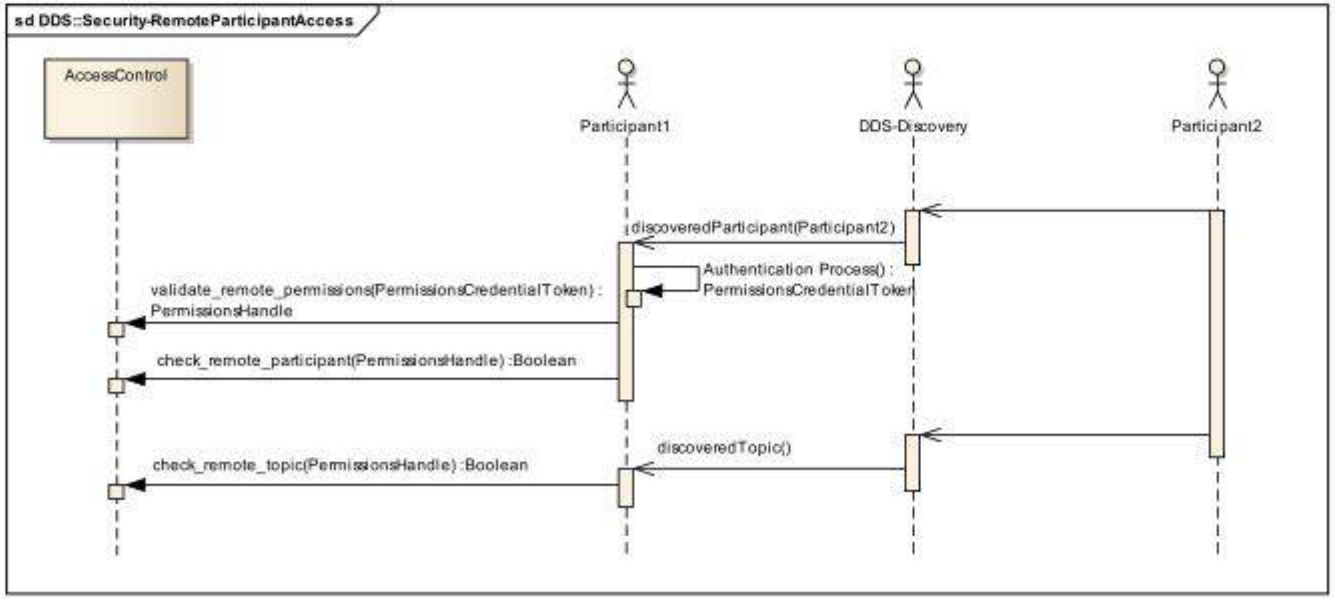

sd DDS::Security-RemoteParticipantAccess
AccessControl
Participant1
DDS-Discovery
Participant2
discoveredParticipant(Participant2)
Authentication Process() : PermissionsCredentialToken
validate_remote_permissions(PermissionsCredentialToken) : PermissionsHandle
check_remote_participant(PermissionsHandle) :Boolean
discoveredTopic()
check_remote_topic(PermissionsHandle) :Boolean

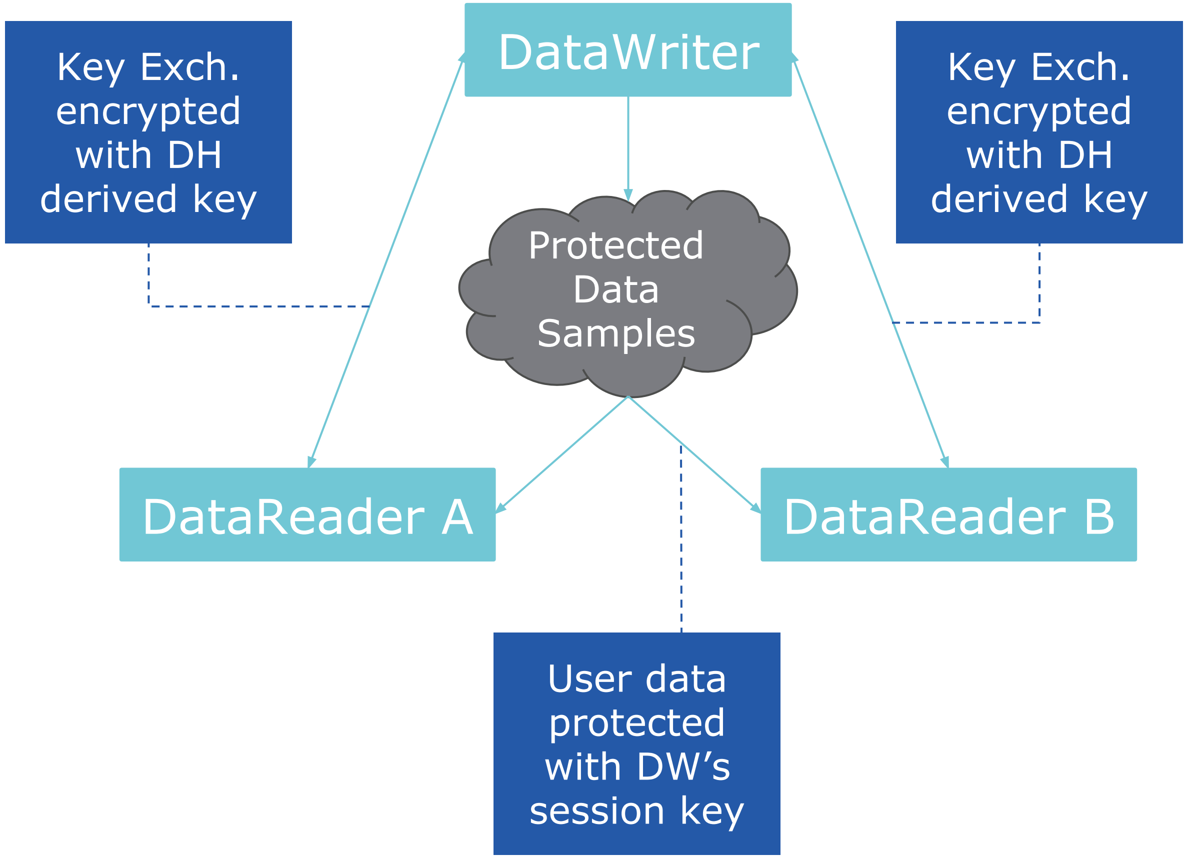
Figure 24 – AccessControl sequence diagram with discovered DomainParticipant

OMG DDS Security 1.1 (formal/2018-04-01)

CRYPTO PLUGIN IN ACTION

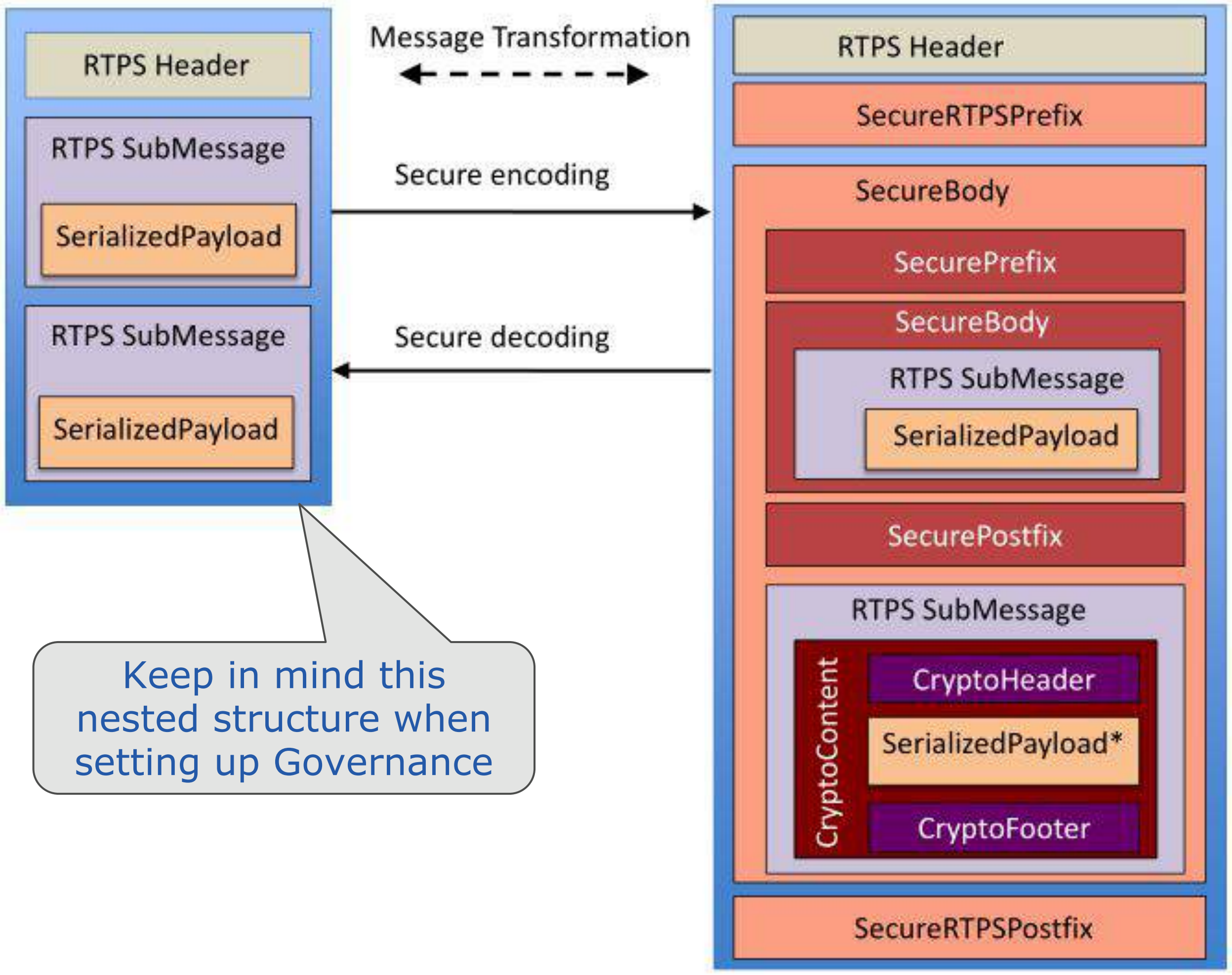
- Key Generation
- Key Exchange
- Data Transformation

Protection Kinds:	Payload	Submessage	Message
None	Trivial	Trivial	Trivial
Sign	Spec issue	OK	Not impl.
Encrypt	OK	OK	Not impl.
Sign+Origin	Not impl.	Not impl.	Not impl.
Encrypt+Origin	Not impl.	Not impl.	Not impl.


Key Exch. encrypted with DH derived key
DataWriter
Key Exch. encrypted with DH derived key
Protected Data Samples
DataReader A
DataReader B
User data protected with DW’s session key

IMPACT ON RTPS PROTOCOL

- Each RTPS Message is a single UDP Datagram
- The Message has a header followed by any number of Submessages
- In Data Submesages, Payloads contain data samples



Keep in mind this nested structure when setting up Governance

Figure 6 – RTPS message transformations

OMG DDS Security 1.1 (formal/2018-04-01)

DEVELOPMENT POST-BETA

- We expect Security capabilities to mature as users get experience with the beta
- You can participate through either model
  - Community support
    - Post on mailing lists, add to FAQ, submit GitHub Issues, Pull Requests
  - Commercial support
    - Design and architecture support
    - Custom development in the middleware or application layers
    - Testing, analysis, and integration support
    - Training and consulting

REPRESENTATIVE CLIENT ENGAGEMENT

- OCI is assessing the viability of using DDS in a next-generation architecture for high-rate data dissemination
- Baseline study: needed integration patterns and goodness-of-fit to DDS features
- Also being assessed:
    - DDS Security features and its benefit to the proposed DDS deployment
    - Understanding of DDS footprint, especially in resource constrained environments
    - Scaling and reliability of DDS for proposed architecture
- Good results are showing already:
    - DDS QoS showing very good alignment to scaling and reliability needed
    - Flexibility of DDS Security is allowing much tighter security decisions to be made
    - Strong access control flexibility combined with strong encryption fits well with need
    - Source-level integration of DDS aligns well with distributed application development

QUESTIONS?

FOR MORE INFORMATION

- OpenDDS project: opendds.org
- Source repository: github.com/objectcomputing/OpenDDS
- Demo (code, binaries, video): opendds.org/quickstart/GettingStartedShapesDemo.html
- Community support: opendds.org/support.html
- OCI commercial support, training, consulting, development: objectcomputing.com/products/opendds
- Webinar: Designing a Distributed Application using DDS QoS: https://www.brighttalk.com/webcast/12231/281491

LEARN MORE ABOUT OCI EVENTS & TRAINING

OCI | WE ARE SOFTWARE ENGINEERS.

Events:

- objectcomputing.com/events

Training:

- objectcomputing.com/training
- grailstraining.com
- micronauttraining.com

Or email info@ocitraining.com to schedule a custom training program for your team online, on site, or in our state-of-the-art, Midwest training lab.

OCI | WE ARE SOFTWARE ENGINEERS.

CONNECT WITH US

☎ +1 (314) 579-0066

@objectcomputing

objectcomputing.com