



Secure Networking Configuration

Release 6.x

Contents

1	Preface	1
1.1	About The Secure Networking Configuration Guide	1
1.2	Intended Audience	1
1.3	Organisation	1
1.4	Conventions	1
2	Overview	3
2.1	Vortex OpenSplice Secure Networking Service	3
2.2	Getting Started	4
3	Secure Networking Configuration	5
3.1	Activating Secure Networking	5
3.2	Secure Networking Configuration Elements	6
3.3	Secure Networking Example Configuration	11
4	Secure Networking Tutorial	13
4.1	Prerequisites	13
4.2	Preparation	13
4.3	Activating Secure Networking	15
5	Access Control	20
5.1	Overview	20
5.2	Mandatory Access Control	20
5.3	Access Control in OpenSplice	21
5.4	MAC in OpenSplice	22
6	Access Control Configuration	24
6.1	Access Control Policy Elements	24
6.2	Access Control Example Configuration	29
7	Troubleshooting	31
7.1	Known Issues	31
7.2	How to Diagnose Problems - Logging	31
8	Contacts & Notices	34
8.1	Contacts	34
8.2	Notices	34

1

Preface

1.1 About The Secure Networking Configuration Guide

The *Vortex OpenSplice Secure Networking Configuration Guide* is intended to be a complete reference for configuring and managing the Vortex OpenSplice Secure Networking Module.

1.2 Intended Audience

The *Secure Networking Configuration Guide* is the starting point for anyone who wishes to use and configure the Vortex OpenSplice Secure Networking Module for securing information transmission between DDS applications across multiple system nodes.

1.3 Organisation

The *Guide* is organised as follows:

The *Overview* provides an introduction to the secure networking module.

Secure Networking Configuration describes how to configure the OpenSplice Secure Networking Module.

The *Tutorial* gives a step-by-step example showing how to configure the OpenSplice Secure Networking Module.

The next section explains how *Access Control* is realized in OpenSplice.

Access Control Configuration describes how to configure the OpenSplice Mandatory Access Control Module.

Finally, the *Troubleshooting* section provides information to help with the diagnosis of common security-related error messages and configuration problems.

1.4 Conventions

The icons shown below are used in PrismTech product documentation to help readers to quickly identify information relevant to their specific use of OpenSplice DDS.

<i>Icon</i>	<i>Meaning</i>
	Item of special significance or where caution needs to be taken.
	Item contains helpful hint or special information.
	Information applies to Windows (<i>e.g.</i> XP, 2003, Windows 7) only.
	Information applies to Unix-based systems (<i>e.g.</i> Solaris) only.
	Information applies to Linux-based systems (<i>e.g.</i> Ubuntu) only.
	C language specific.
	C++ language specific.
	C# language specific.
	Java language specific.

2

Overview

This section gives an overview of the features provided by the Vortex OpenSplice Secure Networking Service.

2.1 Vortex OpenSplice Secure Networking Service

Vortex OpenSplice is a suite of software products for the creation of a secure, performant, and predictable information backbone for distributed systems and systems-of-systems, comprised of a high-performance, low-overhead run-time environment, development tools for modelling information and applications, and run-time tools for monitoring system performance.

The objective of the OpenSplice suite is to reduce complexity, shorten time-to-market, raise quality, and ensure standards compliance and code correctness; all in a single integrated suite of tools and runtime components from a proven and trusted vendor. Fast and predictable networking between the nodes of the information backbone is an essential part of the OpenSplice solution, as is the importance of addressing Information Assurance.

The OpenSplice Secure Networking Service is an optional pluggable service to the OpenSplice Core infrastructure. The Secure Networking Service complements the advanced real-time networking features of the OpenSplice Core module by offering a dedicated crypto-channel *per* network partition.

The OpenSplice Networking service provides essential features for achieving both scalability as well as real-time determinism in mission-critical systems where important high-priority data must ‘pre-empt’ less-important data. The Networking service’s architecture supports multiple runtime-configured *network channels* each representing a pre-emptive priority band which allows for traffic-shaping and data-urgency (Latency-Budget QoS) driven network-packing. The proper network channel is automatically chosen based upon the actual importance (Transport_Priority QoS) of published data. The channel-specific traffic-shaping and bandwidth limitation effectively prevent faulty and/or misbehaving applications from monopolizing the network resources.

The Secure Networking Service extends this real-time ‘network scheduler’ with configurable cryptographic protection implementing transport security with the following properties:

- Information exchanged between nodes of the OpenSplice-based information backbone over unsecure networks cannot be eavesdropped or modified without detection while in transit (data integrity and confidentiality).
- Complete, reliable, and readily-evaluatable separation between the area in which the information is processed in unencrypted form (*RED*, on the node) and the area to which critical (classified) information is not permitted to flow in unencrypted form (*BLACK*, network), is achieved by means of concentration and restriction of network connectivity to exactly one client on each node.
- Authenticity of all information exchanged between nodes.

In addition to transport security, the Secure Networking Service offers pluggable access control. Different modules may be used to enforce access control (for example modules implementing mandatory access control or role-based access control, *etc.*). Currently, OpenSplice supports Mandatory Access Control (MAC). Other modules will be available in a future release. The Mandatory Access Control Module is characterized by the following features:

- Information received *via* the network can only be retrieved on nodes that
 - are accredited for the security level of this information, and
 - host applications that have a need-to-know for the information.

- Information of different classifications can be cryptographically separated while in transit between different nodes, resulting in stronger separation than only labelling, and performing no infiltration or exfiltration between different classifications while in transit.

This infrastructure security solution ensures Information Assurance (IA) for all DDS-based co-operation and information exchange between the DDS nodes over untrusted communication infrastructures. The Secure Networking Service allows the reliable separation of applications with different clearances deployed on different nodes in a way that ensures transparency to the applications, thus supporting full portability.

The OpenSplice Secure Networking Service is the first building block for a complete QoS-enabled IA solution offering end-to-end security between all applications (distributed or co-located), including mandatory access control for all data flowing between applications and detailed security audit of application interactions.

2.2 Getting Started

The Vortex OpenSplice Secure Networking Service is provided along with the OpenSplice commercial edition RTS and HDE product installers.



To activate the Secure Networking Service an OpenSplice licence including the *OPEN-SPLICE_SECURE_NETWORKING* feature is required. Please be sure to read the Release Notes for the latest information.

To install OpenSplice, please follow the instructions and configuration example given in the *Vortex OpenSplice Getting Started Guide*. For an in-depth description of the available configuration settings you may also want to read the *Vortex OpenSplice Deployment Guide* and the *Vortex OpenSplice Configuration Guide*.

If you wish to try out the *Secure Networking Tutorial* you will need to set up at least two networked hosts with OpenSplice. You will also need to build and run the Chat Tutorial application which is described in the *DCPS C Tutorial Guide*.

3

Secure Networking Configuration

This section provides an in-depth description of the OpenSplice security configuration by describing the most important configuration parameters for the Secure Networking Service.

Each configuration parameter will be explained by means of an extensive description together with the tabular summary that contains the following information:

Full path - Describes the location of the item within a complete configuration. Because the configuration is in XML format, an XPath expression is used to point out the name and location of the configuration item.

Format - Describes the format of the value of the configuration item.

Dimension - Describes the unit for the configuration item (for instance *seconds* or *bytes*).

Default value - Describes the default value that is used by the service when the configuration item is not set in the configuration.

Valid values - Describes the valid values for the configuration item. This can be a range or a set of values.

If the configuration parameter is an XML *attribute*, the table also contains the following information:

Required - Describes whether the attribute is required or if it is optional.

If the configuration parameter is an XML *element*, the table also contains the following information:

Occurrences - Describes the range of the possible number of occurrences of the element in the configuration by specifying the minimum and maximum number of occurrences.

3.1 Activating Secure Networking

The OpenSplice Security Service comes as a separate service, replacing the regular networking service.

The secure networking service executable is named `snetworking` on Linux and Solaris and `snetworking.exe` on Windows.

It needs to be configured in the OpenSplice configuration XML file (see below). In addition, a license is required to enable the OpenSplice Secure Networking feature; if a license is not found then an error message is printed and the service will not start.

3.1.1 Configuring the Secure Networking Service

OpenSplice services are configured in an OpenSplice configuration file, which is located in `<OSPL_HOME>/etc/configs/ospl.xml` by default.

Within this configuration file the `<Domain>` element contains a set of Service child elements, which are responsible for starting services like `durability` or `networking`. The default configuration file already contains a `<Service>` element starting the networking service. To configure secure networking only the name of the service's executable in the `<Command>` child element needs to be replaced with the value `snetworking`.

The following snippet shows an example of an OpenSplice configuration starting secure networking service:

```

<OpenSplice>
  <Domain>
    <Name>OpenSplice Security</Name>
    <Database>
      <Size>10485670</Size>
    </Database>
    <Lease>
      <ExpiryTime update_factor="0.5">5.0</ExpiryTime>
    </Lease>
    <Service enabled="true" name="networking">
      <Command>snetworking</Command>
    </Service>
  </Domain>
  ...
</OpenSplice>

```

To check if the service starts correctly the info logfile `ospl-info.log` can be inspected (see also *How to Diagnose Problems - Logging*).

3.2 Secure Networking Configuration Elements

The secure networking configuration expects a root element named `OpenSplice/NetworkingService`. Within this root element, the networking daemon will look for several sub-elements.

The configuration elements for secure networking are listed and explained below.

These elements are an extension of the configuration elements for the Networking Service described in the *Vortex OpenSplice Deployment Guide* and the *Vortex OpenSplice Configuration Guide*.

3.2.1 Element GlobalPartition

This element specifies the global or default networking partition.

Attribute SecurityProfile

In the context of secure networking, the `GlobalPartition` element provides support for the attribute `SecurityProfile`. The attribute is referencing a security profile declared in the context of the `Security` element.

If the given reference is invalid, the global partition configuration is invalid. In this case, the partition will be blocked to prevent unwanted information leaks. A configuration error message will be logged to the `ospl-error.log` file. If the security feature has been enabled, but no profile is declared, then the `NULL` profile is used by default: this means that no security is added to the transport (see *Attribute Enabled*).

Full Path	OpenSplice/NetworkService/Partitioning/GlobalPartition[@SecurityProfile]
Format	string
Dimension	none
Default Value	none
Valid Values	any
Required	no
Remarks	This attribute is referencing a security profile declared in the context of the <code>Security</code> element.

3.2.2 Element NetworkPartition

The Networking configuration can contain a set of networking partitions, which are defined in the context of the `NetworkPartitions` element.

Attribute SecurityProfile

In the context of secure networking, the `NetworkPartition` element provides support for the attribute `SecurityProfile`. The attribute is referencing a security profile declared in the context of the `Security` element.

If the given reference is invalid, the network partition configuration is invalid. In this case the partition will be blocked to prevent unwanted information leaks. A configuration error message will be logged to the `ospl-error.log` file. If the security feature has been enabled but no profile is declared, the `NULL` profile will be used by default.

The ordering of network partition declarations in the OSPL configuration file must be the same for all nodes within the OpenSplice domain.

If certain nodes shall not use one of the network partitions, the network partition in question must be declared as `connected="false"`. In this case the declared security profile would not be evaluated or initialized, and the associated secret cipher keys need not to be defined for the OpenSplice node in question.

Full Path	OpenSplice/NetworkService/NetworkPartitions/NetworkPartition[@SecurityProfile]
Format	string
Dimension	none
Default Value	none
Valid Values	any
Required	no
Remarks	This attribute is referencing a security profile declared in the context of the <code>Security</code> element.

3.2.3 Element Security

The `Security` section defines the parameters relevant for secure networking. Declaring this element in the OSPL configuration file will activate the secure networking feature.

Without any additional security settings, all network partitions of the node would use the `NULL` cipher encoding. If confidentiality and integrity is required for a network partition, the network partition must be associated with a security profile (see `Element SecurityProfile`).

Attribute Enabled

This is an optional attribute.

 If not defined it defaults to `true` and all network partitions, if not specified otherwise, will be encoded using the `NULL` cipher. **The `NULL` cipher does not provide for any level of integrity or confidentiality, and message items will be sent unencrypted.**

If `enabled="false"` the security feature will not be activated, and the node acts like any other OpenSplice node not being security-aware. Security profiles defined in the configuration file will not take effect, but will cause the system to log warnings.

Full Path	OpenSplice/NetworkService/Security[@Enabled]
Format	Boolean
Dimension	none
Default Value	true
Valid Values	true, false
Required	no
Remarks	This attribute is a flag to enable or disable the secure networking.

3.2.4 Element SecurityProfile

This element defines the security profile which can be applied to one or more network partitions. This element is optional.

Attribute Name

This is a mandatory attribute. The name must be unique for all Security Profiles being declared. If the name is not specified, the security profile will be ignored as it cannot be referenced anyway.

Full Path	OpenSplice/NetworkService/Security[@Enabled]
Format	string
Dimension	none
Default Value	none
Valid Values	any, but must be unique amongst all other SecurityProfiles
Required	yes
Remarks	This is a required attribute. The given name can be referenced by NetworkPartition and GlobalPartition elements.

Attribute Cipher

This is a mandatory attribute. Depending on the declared cipher, the cipher key must have a specific length, 128 bits, 192 bits, 256 bits or none at all. The following case-insensitive values are supported by the current implementation:

- **aes128**, implements AES cipher with 128 bit cipher-key (16 Bytes, 32 hexadecimal characters). This cipher will occupy 34 bytes of each UDP packet being sent.
- **aes192**, implements the AES cipher with 192 bit cipher-key (24 Bytes, 48 hexadecimal characters). This cipher will occupy 34 bytes of each UDP packet being sent.
- **aes256**, implements the AES cipher with 256 bit cipher-key (32 Bytes, 64 hexadecimal characters). This cipher will occupy 34 bytes of each UDP packet being sent.
- **blowfish**, implements the Blowfish cipher with 128 bit cipher-key (16 Bytes, 32 hexadecimal characters). This cipher will occupy 26 bytes of each UDP packet being sent.
- **null**, implements the NULL cipher. The only cipher that does not require a cipher-key. This cipher will occupy 4 bytes of each UDP packet being sent.

All ciphers except for the NULL cipher are combined with SHA1 to achieve data integrity. Also, the *rsa-* prefix can be added to the ciphers. In this case, digital signatures using RSA will be available.

Full Path	OpenSplice/NetworkingService/Security/SecurityProfile[@Cipher]
Format	enumeration
Dimension	none
Default Value	none
Valid Values	aes128, aes192, aes256, blowfish, rsa-aes128, rsa-eas192, rsa-eas256, rsa-blowfish, rsa-null, NULL
Required	yes
Remarks	All but NULL cipher require attribute CipherKey to be set with matching key length.

Attribute CipherKey

The `CipherKey` attribute is used to define the secret key required by the declared cipher. The value can be a URI referencing an external file containing the secret key, or the secret key can be defined in-place directly as a string value. The key must be defined as a hexadecimal string, each character representing 4 bits of the key; for example, 1ABC represents the 16-bit key 0001 1010 1011 1100. The key must not follow a well-known pattern and must match *exactly* the key length required by the chosen cipher. If the cipher-keys are malformed, the security

profile in question will be marked as invalid. Moreover, each network partition referring to the invalid Security Profile will not be operational and thus traffic will be blocked to prevent information leaks.

 As all OpenSplice applications require read access to the XML configuration file, for security reasons it is recommended to store the secret key in an external file in the file system, referenced by the URI in the configuration file. The file must be protected against read and write access from other users on the host. Verify that access rights are not given to any other user or group on the host.

 Alternatively, storing the secret key in-place in the XML configuration file will give read/write access to all DDS applications joining the same OpenSplice node. Because of this, the 'in-place' method is strongly discouraged.

Full Path	OpenSplice/NetworkingService/Security/SecurityProfile[@CipherKey]
Format	string
Dimension	none
Default Value	none
Valid Values	Hexadecimal string or file URI matching the pattern: <code>file://.*</code> .
Required	yes
Remarks	All but NULL cipher require attribute CipherKey to be set with matching key length.

3.2.5 Element AccessControl

The optional `AccessControl` element defines settings for access control enforcement and which access control module shall be used.

Attribute enabled

The access control feature will be activated when `enabled="true"`.

Full Path	Security/AccessControl[@enabled]
Format	Boolean
Dimension	none
Default Value	false
Valid Values	false, true

Attribute policy

The `policy` attribute references a file containing the access control policy. Configuration elements of this file are explained in detail in *Access Control Policy Elements*.

Full Path	Security/AccessControl[@policy]
Format	file URI, also see Attribute CipherKey
Dimension	none
Default Value	none
Valid Values	any

3.2.6 Element AccessControlModule

The `AccessControlModule` element defines which access control module will be used. More than one module may be defined. All defined and enabled modules will be used to determine if access should be granted.

Attribute enabled

The module specified in the `type` attribute is used to evaluate access control rules when `enabled="true"`.

Full Path	Security/AccessControl/AccessControlModule[@enabled]
Format	Boolean
Dimension	none
Default Value	true
Valid Values	true, false

Attribute type

The `type` attribute defines the access control model type.

i OpenSplice currently only supports mandatory access control, accordingly the only valid value for this attribute is "MAC".

Full Path	Security/AccessControl/AccessControlModule[@enabled]
Format	string
Dimension	none
Default Value	none
Valid Values	"MAC"
Remarks	This value is referencing an access control module.

3.2.7 Element Authentication

The optional `Authentication` element defines whether additional sender authorization shall be performed. Enabling `Authentication` requires that a cipher, including RSA (such as `rsa-aes256`), is used.

Attribute enabled

`Authentication` is performed when `enabled` is set to `true`.

Full Path	Security/Authentication[@enabled]
Format	Boolean
Dimension	none
Default Value	none
Valid Values	true, false

Element X509Authentication

The `X509Authentication` element defines where keys and certificates required for X509 authentication may be found.

Element Credentials

The `Credentials` element is an optional element. If it is missing, then the node does not sign messages (in other words, does not send credentials).

Element Key The `Key` element references the file containing the key.

Full Path	Security/Authentication/X509Authentication/Credentials/Key
Format	string
Dimension	none
Default Value	none
Valid Values	any file URI
Occurrences (min-max)	1 - 1
Remarks	It is recommended that the absolute path is used. A relative path will be interpreted relative to the directory from which the OpenSplice daemon is started.

Element Cert The `Cert` element references the file containing the certificate.

Full Path	Security/Authentication/X509Authentication/Credentials/Cert
Format	string
Dimension	none
Default Value	none
Valid Values	any file URI
Occurrences (min-max)	1 - 1
Remarks	It is recommended that the absolute path is used. A relative path will be interpreted relative to the directory from which the OpenSplice daemon is started.

Element TrustedCertificates

The `TrustedCertificates` element references a file containing the trusted certificates.

Full Path	Security/Authentication/X509Authentication/TrustedCertificates
Format	string
Dimension	none
Default Value	none
Valid Values	any file URI
Occurrences (min-max)	1 - 1
Remarks	It is recommended that the absolute path is used. A relative path will be interpreted relative to the directory from which the OpenSplice daemon is started.

3.3 Secure Networking Example Configuration

The following XML is an example of a secure networking configuration.

```
<OpenSplice>
  <Domain>
    <Name>OpenSplice Security</Name>
    <Database>
      <Size>10485670</Size>
    </Database>
    <Lease>
      <ExpiryTime update_factor="0.5">5.0</ExpiryTime>
    </Lease>
    <Service enabled="true" name="networking">
      <Command>snetworking</Command>
    </Service>
  </Domain>

  <NetworkService name="networking">
    <Partitioning>
      <GlobalPartition Address="broadcast"
```

```

        SecurityProfile="GlobalProfile"/>
    <NetworkPartitions>
        <NetworkPartition Name="ChatRoomPartition"
            Address="230.230.230.1"
            SecurityProfile="ChatRoomProfile" />
    </NetworkPartitions>
    <PartitionMappings>
        <PartitionMapping DCPSPartitionTopic="ChatRoom.*"
            NetworkPartition="ChatRoomPartition"/>
    </PartitionMappings>
</Partitioning>

<Security enabled="true">
    <SecurityProfile Name="GlobalProfile"
        Cipher="aes128"
        CipherKey="000102030405060708090a0b0c0d0e0f" />
    <SecurityProfile Name="ChatRoomProfile"
        Cipher="blowfish"
        CipherKey="000102030405060708090a0b0c0d0e0f" />
    <SecurityProfile Name="OtherProfile"
        Cipher="rsa-aes128"
        CipherKey=
            "file:///my/shared/secrets/aes128.key" />

    <AccessControl enabled="true" policy="file:///.....">
        <AccessControlModule enabled="true" type="MAC"/>
    </AccessControl/>

    <Authentication enabled="true">
        <X509Authentication>
            <Credentials>
                <Key>
                    file:///usr/osp/securityConfig/ProxyKey.pem
                </Key>
                <Cert>
                    file:///usr/osp/securityConfig/ \
                    ProxyCert.pem
                </Cert>
            </Credentials>
            <TrustedCertificates>
                file:///../../securityConfig/ \
                trustedCerts.pem
            </TrustedCertificates>
        </X509Authentication>
    </Authentication>
</Security>
<Channels>
    <Channel
        enabled="true" name="BestEffort"
        reliable="false" default="true">
        <PortNr>3340</PortNr>
    </Channel>
    <Channel enabled="true" name="Reliable" reliable="true">
        <PortNr>3350</PortNr>
    </Channel>
</Channels>
<Discovery enabled="true">
    <PortNr>3360</PortNr>
</Discovery>
</NetworkService>
</OpenSplice>

```

4

Secure Networking Tutorial

This tutorial demonstrates secure networking features of Vortex OpenSplice.

4.1 Prerequisites

DDS applications on a single host use shared memory for inter process communication. Network communication is needed only in scenarios running DDS applications on multiple hosts. To follow the demonstration of OpenSplice's secure networking features you will need at least two machines connected to the same IP network and OpenSplice installed. Please refer to OpenSplice Getting Started Guide to install OpenSplice on these hosts.

This tutorial is based on the Chat Tutorial delivered with OpenSplice (see `$OSPL_HOME/examples/dcps/standalone/*/Tutorial`).

To demonstrate OpenSplice's secure networking features, no changes to the applications' source code is necessary. Please refer to the Chat Tutorial `README.txt` file and be familiar with building and running the applications (of your preferred programming language) before continuing. Please build and run the Chat Tutorial applications on all hosts. A more in-depth description of the Chat Tutorial can be found in the *DCPS C Tutorial Guide*.

4.2 Preparation

The Chat Tutorial gave a brief introduction to a typical DDS application.

Before activating any security configurations you should know how to start OpenSplice with a customized configuration.



Modifying the default configuration is not recommended because it might affect other users or applications.

4.2.1 Customizing OpenSplice Configuration

Default settings for OpenSplice are read from a configuration file located at `<OSPL_HOME>/etc/configs/ospl.xml`.

To customize settings it is best to run OpenSplice using a *different* configuration file.

Create a file named `ospl.xml` with the content listed below in the Chat Tutorial's directory:

```
<OpenSplice>
  <Domain>
    <Name>OpenSplice Security</Name>
    <Database>
      <Size>10485670</Size>
    </Database>
    <Lease>
      <ExpiryTime update_factor="0.5">5.0</ExpiryTime>
    </Lease>
    <Service enabled="true" name="networking">
```

```

        <Command>snetworking</Command>
    </Service>
</Domain>
<NetworkService name="networking">
    <Partitioning>
        <GlobalPartition Address="broadcast"/>
    </Partitioning>
    <Channels>
        <Channel default="true" enabled="true"
            name="BestEffort" reliable="false">
            <PortNr>3340</PortNr>
        </Channel>
        <Channel enabled="true" name="Reliable"
            reliable="true">
            <PortNr>3350</PortNr>
        </Channel>
    </Channels>
    <Discovery enabled="true">
        <PortNr>3360</PortNr>
    </Discovery>
</NetworkService>
</OpenSplice>

```

Windows Windows users should change the value of the configuration element at `OpenSplice/Domain/Service/Command` to `snetworking.exe`.

Stop OpenSplice if it is already running.

To tell OpenSplice to use this file for configuration, the system variable `OSPL_URI` has to be set with a file URI pointing to the just-created configuration file. For example:

```
% export OSPL_URI=file://ospl.xml
```

will point to a file named `ospl.xml` in the current working directory and

```
% export OSPL_URI=file:///home/user/ospl.xml
```

will point to `ospl.xml` in a `/home/user` directory. Ensure that you have read and write permissions for the file pointed at.

Now start OpenSplice again. You will see that the status message contains a new DDS domain name.

```
% ospl start
% Starting up domain "OpenSplice Security" . Ready
```

Note that when running OpenSplice with the configuration listed above, each OpenSplice node will broadcast heartbeat messages every second, targeting the UDP port 3360.

These messages are used to discover other OpenSplice nodes in the network. Only when a second OpenSplice node has been discovered in the network by heartbeats ChatRoom messages will be broadcasted to the UDP target port 3340 (best effort) and 3350 (reliable). So, do not expect any ChatRoom messages to show up in the network traces as long as you operate with a single OpenSplice node.

4.2.2 Running Chat Tutorial Without Security

Before securing the Chat Tutorial we will run the applications without security activated by using the configuration presented above. This way we can analyse network traffic and identify potential vulnerabilities.

Run the MessageBoard application on one host by executing:

```
% ./exec/MessageBoard
```

On the other host run the Chatter application by executing:

```
% ./exec/Chatter
```

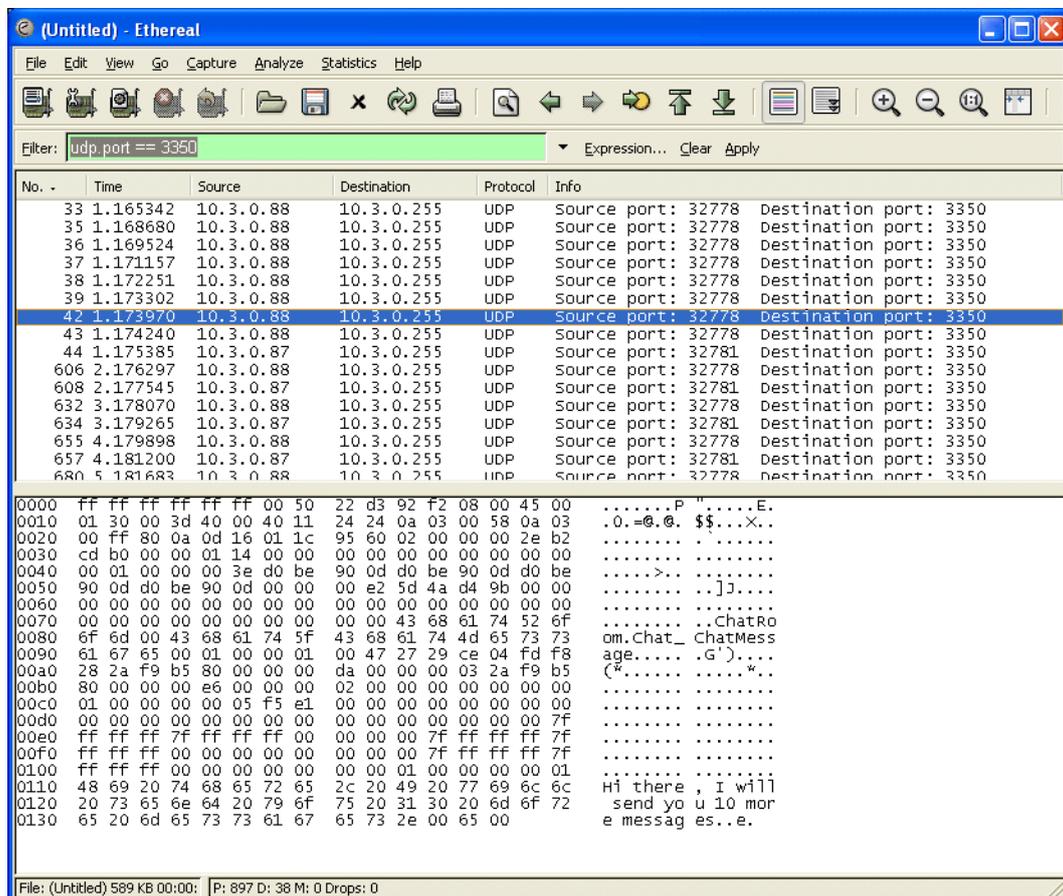
The MessageBoard application will print the following output:

```
% Chatter 1: Hi there, I will send you 10 more messages.
% Chatter 1: Message no. 1
% Chatter 1: Message no. 2
% Chatter 1: Message no. 3
% ...
```

! If running the applications in another shell, ensure that the system variable `OSPL_URI` points to the same configuration file as indicated above when starting OpenSplice. Otherwise the applications won't start up correctly.

To analyse network traffic you may use a tool like Wireshark ¹. Re-run the Chatter application with Wireshark capturing traffic from your network interface. You should see some UDP packets being captured. Analysing packets sent to UDP port 3350 shows the plain text content of our Chat Tutorial messages as you can see in the following screen:

OSPL Sniff Showing Unencrypted Traffic



4.3 Activating Secure Networking

At this point the Chat Tutorial applications should work fine using your own configuration file on both hosts. Up to now all network traffic is sent unencrypted as you can see in the illustration above (OSPL Sniff Showing Unencrypted Traffic).

¹ See <http://www.wireshark.org> for further information.

4.3.1 Simple Setup Using GlobalPartition

To activate encryption, you must shut down the tutorial applications and OpenSplice before modifying the configuration file.

Two steps have to be done for a simple setup.

First, add a new security profile to the network service configuration. This defines details of encryption, like the algorithm and the secret cipher key to be used.

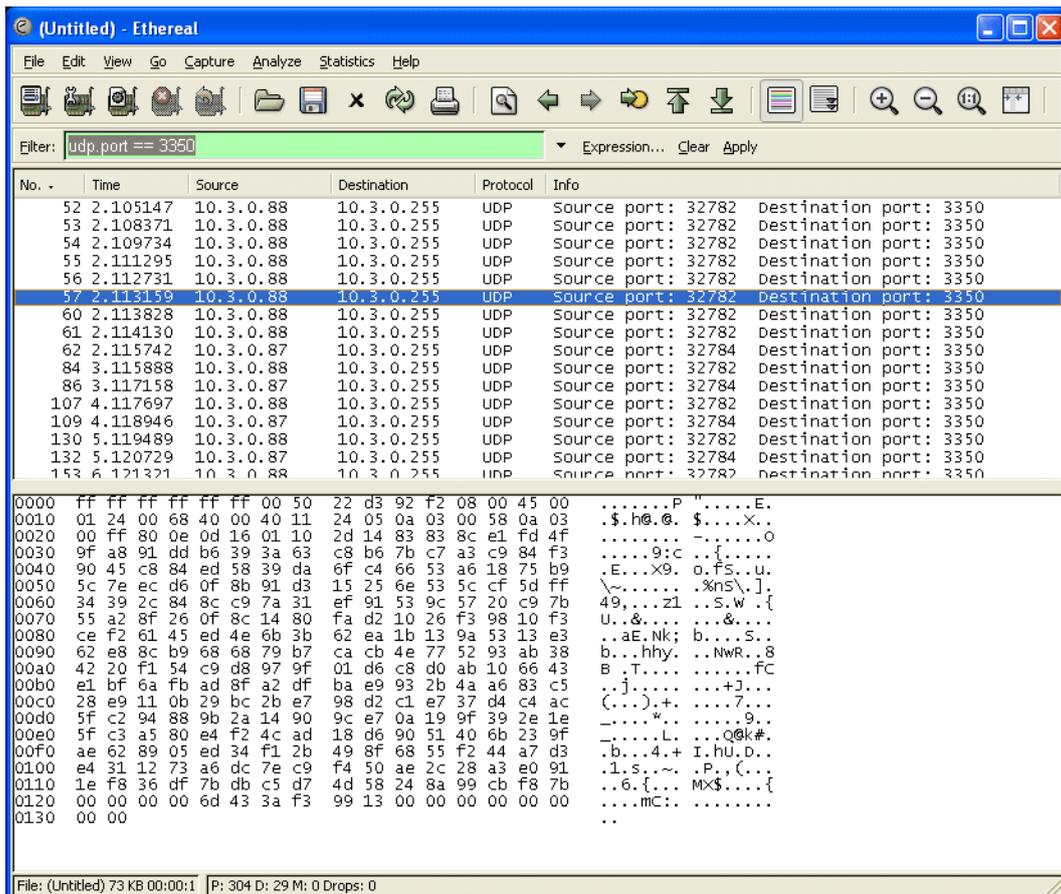
```
<NetworkService name="networking">
  ...
  <Security enabled="true" >
    <SecurityProfile Name="GlobalProfile"
      Cipher="blowfish"
      CipherKey="000102030405060708090a0b0c0d0e0f"
    />
  </Security>
  ...
</NetworkService>
```

Second, you have to associate this security profile with a network partition. Enhance the existing global network partition with a new attribute: `SecurityProfile="GlobalProfile"`. All network traffic sent by the global partition is encrypted using settings from the assigned security profile.

```
<NetworkService name="networking">
  ...
  <Partitioning>
    <GlobalPartition Address="broadcast "
      SecurityProfile="GlobalProfile"/>
    ...
  </Partitioning>
  ...
</NetworkService>
```

As an alternative to modifying the configuration file as described above, you can use the complete example configuration listed below for setup. To avoid problems, ensure that all hosts use the same configuration. Restarting OpenSplice and the Chat Tutorial applications using the modified configuration will enable encrypted network traffic. Again, you may use Wireshark to analyse messages sent through the network. As you can see in the screen shown below, it is not possible to read plain text from chat messages exchanged.

OSPL Sniff Showing Encrypted Traffic



```
<OpenSplice>
  <Domain>
    <Name>OpenSplice Security</Name>
    <Database>
      <Size>10485670</Size>
    </Database>
    <Lease>
      <ExpiryTime update_factor="0.5">5.0</ExpiryTime>
    </Lease>
    <Service enabled="true" name="networking">
      <Command>networking</Command>
    </Service>
  </Domain>
  <NetworkService name="networking">
    <Partitioning>
      <GlobalPartition Address="broadcast"
        SecurityProfile="GlobalProfile"/>
    </Partitioning>
    <Security enabled="true" >
      <SecurityProfile Name="GlobalProfile"
        Cipher="blowfish"
        CipherKey="000102030405060708090a0b0c0d0e0f"/>
    </Security>
    <Channels>
      <Channel default="true" enabled="true"
        name="BestEffort" reliable="false">
        <PortNr>3340</PortNr>
      </Channel>
      <Channel enabled="true" name="Reliable"
        reliable="true">
        <PortNr>3350</PortNr>
      </Channel>
    </Channels>
  </NetworkService>
</OpenSplice>
```

```

    </Channels>
    <Discovery enabled="true">
      <PortNr>3360</PortNr>
    </Discovery>
  </NetworkService>
</OpenSplice>

```

If the cipher key is changed in the configuration of only one host (ensure that the key-length is kept the same) and OpenSplice and the Chat Tutorial applications are restarted you will notice that messages sent by the Chatter application won't reach the MessageBoard running on the other host. On the receiving host a warning message will be logged in the `ospl-info.log` file. This file is located in the start-up directory, by default.

```

### Report Message ###
Type      : WARNING
Context   : networking
nw_plugReceiveChannelReadSocketNonBlocking
File      : ../../code/nw_plugReceiveChannel.c
Line      : 1560
Code      : 0
Description : decoding failed
Node      : shark1.de.primstech.com
Process   : networking (5338)
Thread    : Discovery[@enabled!='false'] b7d1dbb0
Timestamp : 1184225057.899220000 (Thu Jul 12 09:24:17 2007)

```



Before following the instructions in the next section, please restore the correct cipher-keys and restart the OpenSplice services.

4.3.2 Advanced Setup Using Network Partitions

In OpenSplice, DDS topics and DDS partitions can be mapped onto additional network partitions. This allows the separation of network traffic between different applications sharing the same DDS domain. OpenSplice secure networking enables you to use different cipher keys for each network partition to enable confidentiality and integrity of the data exchanged between different applications. For example, the following declaration maps all topics from *ChatRoom* partition onto a dedicated *ChatRoom network partition* using an exclusive multicast address instead of using the *GlobalPartition*:

```

<NetworkService name="networking">
  ...
  <Partitioning>
    <GlobalPartition Address="broadcast"
      SecurityProfile="GlobalProfile"/>
    <NetworkPartitions>
      <NetworkPartition Name="ChatRoomPartition"
        Address="230.230.230.1"
        SecurityProfile="ChatRoomProfile" />
    </NetworkPartitions>
    <PartitionMappings>
      <PartitionMapping DCPSPartitionTopic="ChatRoom.*"
        NetworkPartition="ChatRoomPartition"/>
    </PartitionMappings>
  </Partitioning>
  <Security enabled="true">
    <SecurityProfile Name="GlobalProfile" Cipher="null"/>
    <SecurityProfile Name="ChatRoomProfile"
      Cipher="aes128"
      CipherKey="file:///my/secrets/aes128-chatroom-key.txt"/>
  </Security>
  ...
</NetworkService>

```

With the security feature enabled, the global partition and each network partition can be associated with its own security profile. In our example we refer to *GlobalProfile* and *ChatRoomProfile*. Each security profile defines the cipher to be used. The `NULL` cipher used in *GlobalProfile* may be used for debugging purposes or for single topics/partitions that do not require protection.

The configuration sample above also demonstrates the cipher keys may be stored in an external file too. The key within the file is encoded as a hexadecimal-string, in other words a 16-Byte key is formed of 32 characters, for example: `0f0e0d0c0b0a09080706050403020100`. External files are the preferred way to store keys, to keep them confidential.

 **Permissions for the key files should be set so that nobody except the OpenSplice core services may read the files.** Storing the keys in-place in the configuration file which is read by each DDS application on start up would be a potential leak of secret information.

5

Access Control

The previous sections described how transport security is achieved with the OpenSplice Secure Networking Module. This section explains how access control is realized in OpenSplice and gives a short introduction to the underlying concepts.

5.1 Overview

The OpenSplice Access Control Module is an optional, pluggable service to the OpenSplice Secure Networking Module. It complements the Secure Networking Module by offering different types of access control modules that may be flexibly plugged in (for example, mandatory access control, role-based access control, *etc.*).

i OpenSplice currently supports Mandatory Access Control (MAC) based on the Bell-LaPadula/Biba model. Other Access Control Modules may be available in a future release.

Access Control can be enabled in the OpenSplice secure networking configuration (see *Secure Networking Configuration*). An external file containing the policy rules in XML format can be referenced.

The following section, Mandatory Access Control, introduces mandatory access control. MAC in OpenSplice explains how Mandatory access control is realized in OpenSplice.

5.2 Mandatory Access Control

An access control policy defines whether a user (referred to as a *subject*) is allowed to access an object.

The access control policy is centrally controlled when Mandatory Access Control is in force. Users (subjects) may not override the policy. For example when Mandatory Access Control is in force users may not grant access to objects that would otherwise be restricted.

Mandatory Access Control in OpenSplice ensures data confidentiality as well as data integrity: data may only be created and accessed by authorized parties. The control of information flow is based on the Bell-LaPadula model. This ensures that the information received via the network can only be retrieved on nodes that:

- are accredited for the secrecy level of this information, and
- contain applications that have a need-to-know for the information.

Additionally, care is taken to preserve data integrity (based on the Biba model) by:

- preventing data modification by unauthorized parties
- preventing unauthorized data modification by authorized parties
- maintaining internal and external consistency

The following sections give a short introduction to the Bell-LaPadula and Biba models. MAC in OpenSplice explains how mandatory access control is realized in OpenSplice.

5.2.1 Bell-LaPadula Confidentiality Model

The Bell-LaPadula Confidentiality model describes a set of access control rules which use clearances for subjects (the users) and classifications on objects (resources). The clearance for a user and a classification of an object define a secrecy level and a set of compartments. Secrecy levels are hierarchical and range from the least sensitive, for example “Unclassified” or “Public”, to the most sensitive, for example “Top Secret”.

Compartments are a non-hierarchical list of ID’s (strings) structuring information to enforce the ‘need-to-know’ principle.

The clearance of the subject is compared to the classification of the object in order to determine if a subject is authorized to access a certain object. The Bell-LaPadula model defines two mandatory access control rules:

- a subject at a given secrecy level may not read an object at a higher secrecy level (no *read-up*).
- a subject at a given secrecy level must not write to any object at a lower secrecy level (no *write-down*).

In the Bell-LaPadula model, users can create content only at or above their own secrecy level. For example, secret researchers can create “Secret” or “Top-Secret” files, but may not create “Public” files (no write-down). On the other hand, users can view content only at or below their own secrecy level. For example, secret researchers can view “Public” or “Secret” files, but may not view “Top-Secret” files (no read-up).

Additionally, a subject must have a ‘need-to-know’ for the object it wants to access; in other words, the set of compartments contained in the user’s clearance must be a subset of the set of compartments contained in the object’s classification.

5.2.2 Biba Integrity Model

The Biba Integrity Model describes a set of access control rules designed to ensure data integrity. Resources (objects) and users (subjects) are grouped into ordered levels of integrity.

The Biba model defines two mandatory access control rules:

- a subject at a given level of integrity may not read an object at a lower integrity level (no read down).
- a subject at a given level of integrity must not write to any object at a higher level of integrity (no write up).

In the Biba model, users can only create content at or below their own integrity level (no write up). On the other hand, users can only view content at or above their own integrity level (no read down).

5.3 Access Control in OpenSplice

In OpenSplice, access control is enforced on the network-level for inter-node communication, in other words, the communication between different nodes. One node (host) corresponds to one user. When running multiple DDS applications on one host these will all run under the same User ID.

There are two access control enforcement points:

- for inbound traffic: when reading data from the network, the following checks are carried out:
 - is the reader allowed to receive the data,
 - was the data published by a trusted node (in other words, was the sender allowed to send the data)
- for outbound traffic: when writing data to the network, the following check is carried out:
 - is the user allowed to write data to the network



Note that access control is not enforced for intra-node communication, in other words communication between DDS applications running on the same node.

5.3.1 Objects to be Protected

One access control policy applies to one DDS Domain. The objects to be protected by the access control policy in OpenSplice apply to DDS topics and DDS partitions.

For example, a classification can be attached to a topic. This may result in restricting access in a way that only users with a matching clearance may use this topic for publication of data or for subscription to data of this topic, respectively. The same applies to partitions which are referenced by a resource element of the access control policy.

5.3.2 Secure vs. Insecure

The ‘*secure*’ world is strictly separated from the ‘*insecure*’ world in OpenSplice. If a resource does not have access control information (for example a classification), then no access control will be enforced for this resource. If a user has no access control information (for example a clearance) they may only access ‘insecure’ resources (resources that do not have access control information like a classification). If a user has access control information, then they may not access ‘insecure’ resources that don’t have access control information.

5.3.3 User Authentication

The system authenticates the user’s identity.



OpenSplice currently supports SSL X.509 Certificate Authentication. Other authentication mechanisms (such as user ID - password authentication) may be available in a future release.

X.509 is a standard for digital certificates used by SSL. Certificates include, among other things, information about the identity of the certificate holder (in other words: the user). The user authenticates to the system presenting a certificate. The system accepts the user’s certificate if it was issued by a trusted authority. A list of trusted certificates can be defined in the secure networking configuration (see *Element X509Authentication* and *Element TrustedCertificates*).

The user’s identity is determined by a distinguished name (DN) contained in the user’s certificate. The system retrieves the corresponding user from the access control policy (each user entry in the policy defines the subject’s DN) and applies the defined access control information to this user.

5.4 MAC in OpenSplice

Mandatory Access Control (MAC) in Open Splice combines the Bell-LaPadula and Biba models to ensure confidentiality and data integrity. Each resource (object) has a classification which comprises a secrecy level, an integrity level and a set of compartments that this resource is intended for. Each user (subject) has a clearance which comprises a secrecy level, an integrity level and a set of compartments this user has a ‘need-to-know’ for.

In order to determine if a user is authorized to access a certain resource, in other words, if they can publish a certain topic or subscribe to a topic, the clearance of the user is compared to the classification of the resource. This process comprises the evaluation of the:

- secrecy level
 - subscribing is permitted if the resource’s secrecy level is identical or lower than the user’s secrecy level
 - publishing is permitted if the resource’s secrecy level is identical or higher than the user’s secrecy level
- integrity level
 - subscribing is permitted if the resource’s integrity level is identical or higher
 - publishing is permitted if the resource’s integrity level is identical or lower
- need to know (compartment)

- publish/subscribe is permitted if the user's set of compartments is a subset of the resource's set of compartments

Access is only granted if *all three parts* of the user's clearance (secrecy level, integrity level, and need-to-know) are evaluated positively against the resource's classification.

The following section, *Access Control Configuration* describes access control policy configuration elements for MAC in OpenSplice.

6

Access Control Configuration

This section provides a detailed description of the OpenSplice access control policy configuration.

Each configuration parameter will be explained by means of an extensive description together with the tabular summary that contains the following information:

Full path - Describes the location of the item within a complete configuration. Because the configuration is in XML format, an XPath expression is used to point out the name and location of the configuration item.

Format - Describes the format of the value of the configuration item.

Dimension - Describes the unit for the configuration item (for instance *seconds* or *bytes*).

Default value - Describes the default value that is used by the service when the configuration item is not set in the configuration.

Valid values - Describes the valid values for the configuration item. This can be a range or a set of values.

If the configuration parameter is an XML *attribute*, the table also contains the following information:

Required - Describes whether the attribute is required or if it is optional.

If the configuration parameter is an XML *element*, the table also contains the following information:

Occurrences - Describes the range of the possible number of occurrences of the element in the configuration by specifying the minimum and maximum number of occurrences.

6.1 Access Control Policy Elements

When access control is enabled a file containing the access control policy configuration is referenced in the secure networking configuration.

The access control policy configuration expects a root element named `accessControlPolicy`. Elements defined in an access control policy are listed and explained in the following sections.

6.1.1 Element `secrecyLevels/secrecyLevel`

The access control policy contains a hierarchical list of secrecy levels which are grouped under the `secrecyLevels` element. Typical secrecy levels would be: UNCLASSIFIED, RESTRICTED, CONFIDENTIAL, SECRET, and TOP_SECRET.



Note that the order of defined secrecy levels is important: secrecy levels are listed from weakest to strongest.

Full Path	accessControlPolicy/secretcyLevels/secretcyLevel
Format	string
Dimension	none
Default Value	none
Valid Values	any
Occurrences (min-max)	0 - *
Remarks	Listed from the weakest to the strongest level.

6.1.2 Element integrityLevels/integrityLevel

The access control policy contains a list of integrity levels which are grouped under the `integrityLevels` element.



The order of defined integrity levels is important. Integrity levels are listed from the weakest to the strongest.

Full Path	accessControlPolicy/integrityLevels/integrityLevel
Format	string
Dimension	none
Default Value	none
Valid Values	any
Occurrences (min-max)	0 - *
Remarks	Listed from the weakest to the strongest level.

6.1.3 Element users/user

The `users` section contains a set of users. A user has an *id*, a *clearance*, and a list of *authentication mechanisms*.

Element id

Full Path	accessControlPolicy/users/user/id
Format	string
Dimension	none
Default Value	none
Valid Values	any
Occurrences (min-max)	0 - 1
Remarks	none

Element Clearance

A clearance consists of this user's secrecy level, integrity level, and a set of compartments.

Element secretcyLevel

Defines this user's secrecy level.

Full Path	accessControlPolicy/users/user/clearance/secretLevel
Format	string
Dimension	none
Default Value	none
Valid Values	any, defined in accessControlPolicy/secretLevels
Occurrences (min-max)	0 - 1
Remarks	none

Element integrityLevel

Defines this user's integrity level.

Full Path	accessControlPolicy/users/user/clearance/integrityLevel
Format	string
Dimension	none
Default Value	none
Valid Values	any, defined in accessControlPolicy/integrityLevels
Occurrences (min-max)	0 - 1
Remarks	none

Element compartments/compartment

The `compartments` section contains a set of compartments this user is entitled to access.

Full Path	accessControlPolicy/users/user/clearance/compartments/compartment
Format	string
Dimension	none
Default Value	none
Valid Values	any
Occurrences (min-max)	0 - *
Remarks	none

Element authentication

This element contains a list of authentication mechanisms for this user.

 Currently, OpenSplice supports SSL X.509 Certificate Authentication. Other authentication mechanisms (such as user ID/password authentication) may be available in a future release.

Element x509Authentication

Defines properties of x509 (SSL certificate) authentication.

Element subject

Full Path	accessControlPolicy/users/user/authentication/x509Authentication/subject
Format	string
Dimension	none
Default Value	none
Valid Values	any
Occurrences (min-max)	0 - 1
Remarks	The distinguished name (DN) of the certificate the user transmits to authenticate to the system (single elements of the distinguished name have to be separated by a comma).

 Note that the user's certificate DN must be unique: ensure that multiple users do not share the same client certificate DN.

6.1.4 Element resources/resource

The `resources` section contains a set of resources, in other words, the objects to be protected.

A resource has a resource identification (made up of the resource's type, id, and topic or partitions, respectively) and a classification (containing the resource's secrecy and integrity level and a list of compartments). The classification is used for mandatory access control.

Element type

Defines the type of this resource. A resource can have the type `PARTITION` or `TOPIC`.

Full Path	accessControlPolicy/resources/resource/type
Format	string
Dimension	none
Default Value	none
Valid Values	"PARTITION" or "TOPIC"
Occurrences (min-max)	0 - 1
Remarks	none

Element id

Defines this resource's id.

Full Path	accessControlPolicy/resources/resource/id
Format	string
Dimension	none
Default Value	none
Valid Values	any valid topic or partition name of a DDS domain
Occurrences (min-max)	0 - 1
Remarks	This is related to the type element value.

Element topics/topic

The `topics` section contains a set of topics. This element is only valid if the type of the resource is `TOPIC`. It lists all valid topics that may be part of this partition.

Full Path	accessControlPolicy/resources/resource/topics/topic
Format	string
Dimension	none
Default Value	none
Valid Values	any valid topic of a DDS domain
Occurrences (min-max)	0 - 1
Remarks	Exists only if element type = PARTITION

Element partitions/partition

The `partitions` section contains a set of partitions. This element is only valid if the type of the resource is `PARTITION`. It lists all valid partitions that may be part of this partition.

Full Path	accessControlPolicy/resources/resource/partitions/partition
Format	string
Dimension	none
Default Value	none
Valid Values	any valid partition of a DDS domain
Occurrences (min-max)	0 - *
Remarks	Exists only if element type = TOPIC

Element classification

A classification consists of this resource's secrecy level, integrity level, and a set of compartments.

Element secrecyLevel

Defines this resource's secrecy level.

Full Path	accessControlPolicy/resources/resource/classification/secrecyLevel
Format	string
Dimension	none
Default Value	none
Valid Values	any, defined in <code>accessControlPolicy/secrecyLevels</code>
Occurrences (min-max)	0 - 1
Remarks	none

Element integrityLevel

Defines this resource' integrity level.

Full Path	accessControlPolicy/resources/resource/classification/integrityLevel
Format	string
Dimension	none
Default Value	none
Valid Values	any, defined in <code>accessControlPolicy/integrityLevels</code>
Occurrences (min-max)	0 - 1
Remarks	none

Element compartments/compartment

The compartments section contains a set of compartments this resource is intended for.

Full Path	accessControlPolicy/resources/resource/classification/compartments/compartment
Format	string
Dimension	none
Default Value	none
Valid Values	any
Occurrences (min-max)	0 - *
Remarks	none

6.2 Access Control Example Configuration

The following XML shows an example access control policy.

```
<accessControlPolicy>
  <secrecyLevels> <!-- for MAC -->
    <secrecyLevel>UNCLASSIFIED</secrecyLevel>
    <secrecyLevel>CONFIDENTIAL</secrecyLevel>
    <secrecyLevel>SECRET</secrecyLevel>
    <secrecyLevel>TOP_SECRET</secrecyLevel>
  </secrecyLevels>

  <integrityLevels> <!-- for MAC -->
    <integrityLevel>LEVEL_0</integrityLevel>
    <integrityLevel>LEVEL_1</integrityLevel>
    <integrityLevel>LEVEL_2</integrityLevel>
  </integrityLevels>
  <users>
    <user>
      <id>user1</id>
      <clearance> <!-- for MAC -->
        <secrecyLevel>CONFIDENTIAL</secrecyLevel>
        <integrityLevel>LEVEL_2</integrityLevel>
        <compartments>
          <compartment>US Only</compartment>
          <compartment>Air Force</compartment>
          <compartment>Radar</compartment>
        </compartments>
      </clearance>
      <authentication>
        <x509Authentication>
          <subject>DN</subject>
        </x509Authentication>
      </authentication>
    </user>
    <user>
      <id>user2</id>
      <authentication>
        <x509Authentication>
          <subject>DN2</subject>
        </x509Authentication>
      </authentication>
    </user>
  </users>
  <resources>
    <resource>
      <type>PARTITION</type>
    </resource>
  </resources>
</accessControlPolicy>
```

```
        <id>chat</id>
    <topics>
        <topic>ChatMessage</topic>
        <topic>NamedMessage</topic>
    </topics>
    <classification> <!-- for MAC -->
        <secretLevel>CONFIDENTIAL</secretLevel>
        <integrityLevel>LEVEL_1</integrityLevel>
        <compartments>
            <compartment>US Only</compartment>
            <compartment>Air Force</compartment>
        </compartments>
    </classification>
</resource>
<resource>
    <type>TOPIC</type>
    <id>pingpong</id>
    <partitions>
        <partition>PING</partition>
        <partition>PONG</partition>
    </partitions>
</resource>
<resource>
    <type>TOPIC</type>
    <id>topic1</id>
    <partitions>
    </partitions>
</resource>
</resources>
</accessControlPolicy>
```

7

Troubleshooting

This section provides information for the diagnosis of common (security-related) issues, error messages, and configuration problems.

7.1 Known Issues

This section lists known issues and their resolution.

- **The networking process does not start up on Windows.**
Check for the startup command `snetworking.exe` (cf. *Customizing OpenSplice Configuration* .)
- **OpenSplice does not start up, although the used configuration is correct.**
Make sure that no shared memory file exists from a previous run. This issue shows up mostly on Windows. Remove shared memory files located in `%TEMP%`.

7.2 How to Diagnose Problems - Logging

If you are experiencing problems take a look at the OpenSplice log files.

- Error reports are written to the `ospl-error.log` file located in the start-up directory of the OpenSplice daemon, by default.
- Information and warning reports are written to the `ospl-info.log` file, by default.

Generally, you do not need to worry about the `ospl-info.log` file, but in some cases, warning messages may help to understand the error message contained in the `ospl-error.log` file.

7.2.1 Error Messages

Below are described security-related error messages and their resolution. Error messages are written to the `ospl-error.log` file.

- **Partition x (<partition_name>) - undefined security profile <profile_name>.**
This message indicates a missing security profile where `x` is an internal partition identifier to which a security profile named `<profile_name>` has been assigned in the configuration file. Check if the `<profile_name>` security profile has been defined within the security element of the configuration file.
- **Failed to initialize the security module.**
The Security XML configuration is faulty and some information is missing or incorrect. Generally, this message comes after one or more messages reporting missing or incorrect configuration of XML elements. Check the `ospl-info.log` security-related messages for additional information.

- **Partition x (<partition_name>)** - security profile <profile_name> requires cipher key of <bits_nb> bits.

The partition x has been assigned a faulty security profile with a missing CipherKey attribute. Note that:

- Null cipher requires no cipher key
- Blowfish cipher requires a key of 128 bits (32 hex-characters)
- AES128 cipher requires a key of 128 bits (32 hex-characters)
- AES192 cipher requires a key of 192 bits (48 hex-characters)
- AES256 cipher requires a key of 256 bits (64 hex-characters)

- **Partition x (<partition_name>)** - security profile <profile_name> with invalid cipher <cipher_name>.

The partition x has been assigned to a faulty security profile with an invalid cipher. Check that the <cipher_name> identifier is correctly spelled and supported.



Note that only aes128, aes192, aes256, blowfish, rsa-aes128, rsa-192, rsa-256, rsa-blowfish, rsa-null and NULL ciphers are currently supported.

- **Dropping traffic of partition x due to insufficient cipher key, until re-keying has been done.**

Partition x traffic is temporarily blocked because of faulty cipher key.

- **Receiving message blocked, bad partition encoding, verify if sending node has security feature enabled.**

A received message cannot be assigned to a valid network partition, the message is blocked and not delegated to the Data Reader. This may be caused by the security feature not being activated on all participating OpenSplice nodes in the domain. Make sure that the security feature is enabled in all OSPL configuration files (`Security[@enabled=true]`). Note that the network partition is still able to receive data samples from nodes where the OSPL configuration matches the local network partition configuration.

- **Sending message blocked, bad partition x.**

Message sending on partition x is blocked because of faulty security profile definition. Check the `ospl-info.log` for security-related warning messages indicating missing elements.

7.2.2 Warning Messages

This section describes security-related warning messages and their resolution. Warning messages are written to the `ospl-info.log` file. Note that warning messages can be ignored in the absence of errors.

- **Name attribute of security profile undefined, or empty string.**

The Name attribute of one or more SecurityProfile XML elements is missing in the configuration file.

- **Cipher attribute of security profile <profile_name> undefined.**

The Cipher attribute of one or more SecurityProfile XML elements is missing in the configuration file.

- **CipherKey attribute of security profile <profile_name> not defined.**

The CipherKey attribute of one or more SecurityProfile XML elements is missing in the configuration file. Note that the CipherKey attribute is required only for non-NULl ciphers.

- **SecurityProfile <profile_name> has invalid cipher key <cipher_key>, check length and encoding.**

Check the <profile_name> security profile definition for key length and hexadecimal encoding correctness.

- **Security profile <profile_name> defines unknown cipher <cipher_name>.**

Check the <cipher_name> identifier to see that it is correctly spelled and supported. Note that only aes128, aes192, aes256, blowfish, rsa-aes128, rsa-192, rsa-256, rsa-blowfish, rsa-null and NULL ciphers are currently supported.

8

Contacts & Notices

8.1 Contacts

PrismTech Corporation

400 TradeCenter
Suite 5900
Woburn, MA
01801
USA
Tel: +1 781 569 5819

PrismTech Limited

PrismTech House
5th Avenue Business Park
Gateshead
NE11 0NG
UK
Tel: +44 (0)191 497 9900

PrismTech France

28 rue Jean Rostand
91400 Orsay
France
Tel: +33 (1) 69 015354

Web: <http://www.prismtech.com>

E-mail: info@prismtech.com

8.2 Notices

Copyright © 2016 PrismTech Limited. All rights reserved.

This document may be reproduced in whole but not in part. The information contained in this document is subject to change without notice and is made available in good faith without liability on the part of PrismTech Limited or PrismTech Corporation. All trademarks acknowledged.