Document Title	Specification of Intrusion Detection System Manager for Adaptive Platform
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	978

Document Status	published
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	R21-11

Document Change History			
Date	**Release**	**Changed by**	**Description**
2021-11-25	R21-11	AUTOSAR Release Management	• No content changes
2020-11-30	R20-11	AUTOSAR Release Management	• initial release

Disclaimer

Table of Contents

1 Introduction and functional overview

This specification describes the functionality, API and the configuration for the AUTOSAR Adaptive Functional Cluster IdsM.

2 Acronyms and Abbreviations

2.1 Acronyms

Acronym	Description:
Filter Chain	A set of consecutive filters which is applied to Security Events-
Intrusion Detection System	An Intrusion Detection System is a security control which detects and processes security events.
Intrusion Detection System Manager	The Intrusion Detection System Manager handles security events reported by security sensors.
Intrusion Detection System Reporter	The Intrusion Detection System Reporter handles qualified security events received from Idsm instances.
Security Extract	The Security Extract specifies which security events are handled by IdsM instances and their configuration parameters.
Security Event Type	A security event type can be identified by its security event type ID. Instances of security event types are called security events and share the same security event type ID.
Security Events	Onboard Security Events are instances of security event types which are reported by BSW or SWC to the IdsM.
Security Event Memory	A user defined diagnostic event memory which is independent from the primary diagnostic event memory.
Security Sensors	BSW or SWC which report security events to the Idsm.
Qualified Security Events	Security events which pass their filter chain are regarded as Qualified Security Events.
Security Event Memory	User defined diagnostic event memory which is separated from the main diagnostic event memory.
Security Incident and Event Management	Process for handling a confirmed security incident
Security Operation Centre	Organization of security and domain experts who are analyzing security events and contributing to mitigation of threats.

Table 2.1: Acronyms

2.2 Abbreviations

Abbreviation	Description:
DID	Data Identifier according to Unified Diagnostic Services
DTC	Diagnostics Trouble Code
FC	Functional Cluster
IDS	Intrusion Detection System
IdsM	Intrusion Detection System Manager
IdsR	Intrusion Detection System Reporter
SecXT	Security Extract
SEv	Security Event
QSEv	Qualified Security Event
Sem	Security Event Memory
SIEM	Security Incident and Event Management
SOC	Security Operation Centre
SWCL	Software Cluster

Table 2.2: Abbreviations

3 Related documentation

This document is part of the AUTOSAR IDS specification and covers the software specification for the `Adaptive Platform`. For other aspects of the IDS specification, please refer to the following documents:

- **System Requirements Specification of Intrusion Detection System (RS IDS) [1]:** Specifies IDS system requirements.
- **Protocol Requirements on transmission of qualified security events (PRS IDS) [2]:** Specifies the communication protocol between for the transmission of security events.
- **Security Extract Template [3]:** Specifies the Security Extract.

3.1 Input documents & related standards and norms

[1] Requirements on Intrusion Detection System
AUTOSAR_RS_IntrusionDetectionSystem

[2] Specification of Intrusion Detection System Protocol
AUTOSAR_PRS_IntrusionDetectionSystem

[3] Security Extract Template
AUTOSAR_TPS_SecurityExtractTemplate

[4] Specification of Adaptive Platform Core
AUTOSAR_SWS_AdaptivePlatformCore

[5] Specification of Cryptography
AUTOSAR_SWS_Cryptography

3.2 Further Applicable Specification

AUTOSAR provides a core specification [4] which is also applicable for `Intrusion Detection System Manager`. The chapter "General requirements for all FunctionalClusters" of this specification shall be considered as an additional and required specification for implementation of `Intrusion Detection System Manager`.

4 Constraints and assumptions

There are no known constraints and assumptions.

4.1 Known limitations

There are no known limitations.

5 Dependencies to other Functional Clusters

Security events generated via the `IdsM` API can be accessed using diagnostic services. Security events sent to the `IdsR` can be signed using a key modeled in `FC Crypto`.

5.1 Protocol layer dependencies

Security events generated via the `IdsM` API can be transmitted to the `IdsR` using the protocol specified in PRS IDS [2].

6 Requirements Tracing

The following tables reference the requirements specified in System Requirements Specification of Intrusion Detection System (RS IDS) [1] and links to the fulfillment of these. Please note that if column “Satisfied by” is empty for a specific requirement this means that this requirement is not fulfilled by this document.

Requirement	Description	Satisfied by
[RS_Ids_00100]	Initialization of the IdsM	[SWS_AIDSM_00001] [SWS_AIDSM_00002]
[RS_Ids_00200]	Provide Interface for reporting SEv	[SWS_AIDSM_01201]
[RS_Ids_00300]	Provide configurable filter chains for qualifying SEv	[SWS_AIDSM_00301] [SWS_AIDSM_00303] [SWS_AIDSM_00304] [SWS_AIDSM_00305] [SWS_AIDSM_00306]
[RS_Ids_00301]	Provide multiple filter chains	[SWS_AIDSM_00301]
[RS_Ids_00310]	Configure reporting mode per Security Event Type and IdsM instance	[SWS_AIDSM_00101] [SWS_AIDSM_00201] [SWS_AIDSM_00202]
[RS_Ids_00320]	Support machine state filter	[SWS_AIDSM_00401]
[RS_Ids_00330]	Support sampling filter	[SWS_AIDSM_00501] [SWS_AIDSM_00502]
[RS_Ids_00340]	Support Aggregation filter	[SWS_AIDSM_00600] [SWS_AIDSM_00601] [SWS_AIDSM_00602] [SWS_AIDSM_00603] [SWS_AIDSM_00604] [SWS_AIDSM_00605] [SWS_AIDSM_00606] [SWS_AIDSM_00607]
[RS_Ids_00350]	Support Threshold filter	[SWS_AIDSM_00701] [SWS_AIDSM_00702]
[RS_Ids_00400]	Persist QSEv records	[SWS_AIDSM_01301]
[RS_Ids_00502]	Event Timestamps	[SWS_AIDSM_00801]
[RS_Ids_00503]	Timestamp Sources	[SWS_AIDSM_00802] [SWS_AIDSM_00803] [SWS_AIDSM_00804] [SWS_AIDSM_00805] [SWS_AIDSM_00806] [SWS_AIDSM_00807]
[RS_Ids_00505]	Authenticity of QSEvs	[SWS_AIDSM_01001] [SWS_AIDSM_01002]
[RS_Ids_00510]	The IdsM shall allow to transmit QSEv to the IdsR	[SWS_AIDSM_00901] [SWS_AIDSM_00902]
[RS_Ids_00511]	Limit event rate and traffic	[SWS_AIDSM_01101] [SWS_AIDSM_01103] [SWS_AIDSM_01104]
[RS_Ids_00610]	Configuration of qualification filters for SEv	[SWS_AIDSM_00302]
[RS_Ids_00700]	Reconfiguration during run-time	[SWS_AIDSM_01302] [SWS_AIDSM_01303]
[RS_Ids_00810]	Basic SW security events	[SWS_IdsM_91015]

Requirement	Description	Satisfied by
[RS_Ids_00820]	IdsM Security Events	[SWS_AIDSM_01401] [SWS_AIDSM_01402] [SWS_AIDSM_01403]

7 Functional specification

This chapter specifies the function behavior of the IdsM for the Adaptive Platform.

7.1 Functional cluster life-cycle

Using `ara::core::Intitialize` and `ara::core::Deinitialize`, the application can initialize and deinitialize its ara::idsm library.

[SWS_AIDSM_00001]{DRAFT} ⌈When `ara::core::Intitialize` is called, IdsM shall read in the manifest information and prepare the access structures necessary to generate events from the application.⌋*(RS_Ids_00100)* Access structures may encompass the communication channel between the application process and the stack process (if there is any) or other resource required by the IdsM.

[SWS_AIDSM_00002]{DRAFT} ⌈When `ara::core::Deinitialize` is called, the IdsM shall close all accquired handles and free all access structures.⌋*(RS_Ids_00100)*

The application is expected not to call any API of `IdsM` before `ara::core::Intitialize` or after `ara::core::Deinitialize`.

7.2 Event Generation

`SWCL`s and `FC`s can generate new security events using the `IdsM` API. All event types that can be generated by a `SWCL` are configured in the manifest and linked to a PortPrototype of the SWCL. Generating new events involves three steps:

1. Construct an `InstanceSpecifier` object using the shortName path of the `PortPrototype` referencing the event type as the parameter.
2. Construct an `ara::idsm::EventReporter` object by passing the `InstanceSpecifier`.
3. Call the `ara::idsm::EventReporter::ReportEvent` function on the `ara::idsm::EventReporter` object.

Using the `ara::idsm::EventReporter::ReportEvent` function, an application can optionally provide a timestamp, a counter, and/or context data.

[SWS_AIDSM_00101]{DRAFT} **Security Event Type** ⌈Each `Security Event Type` is represented by one `SecurityEventDefinition` object in the model and shall be uniquely identified by the model parameter `SecurityEventDefinition.id`.⌋*(RS_Ids_00310)*

7.3 Reporting Mode

[SWS_AIDSM_00201]{DRAFT} **Reporting Mode** ⌈IdsM shall determine the default reporting mode of every reported SEv from the SecXT model parameter SecurityEventContextProps.defaultReportingMode.⌋*(RS_Ids_00310)*

[SWS_AIDSM_00202]{DRAFT} **Reporting Mode Options** ⌈IdsM shall handle reported SEv depending on its reporting mode according to Table Table 7.1.⌋*(RS_Ids_00310)*

Reporting Mode Level	*Related Behavior*
OFF	IdsM shall discard the SEv without further processing.
BRIEF	If the SEv has been reported including context data, IdsM shall shall discard the context data from further processing, transmission, and storage.
DETAILED	If the SEv has been reported including context data, IdsM shall keep the context data for potential transmission or persisting of the QSEv.
BRIEF_BYPASSING_FILTERS	IdsM shall report or persist the SEv without context data without further application of any filter chain.
DETAILED_BYPASSING_FILTERS	IdsM shall report or persist the SEv with context data (if provided by the sensor) without further application of any filter chain.

Table 7.1: Reporting Mode Filter Values

7.4 Filter Chain

Filter chains are configured using the SecXT model element SecurityEventFilterChain.

[SWS_AIDSM_00301]{DRAFT} **Filter chain selection** ⌈When a SEv is reported, the IdsM shall apply the filter chain that is mapped to the SecurityEventDefinition of the reported SEv via the SecurityEventContextMapping.⌋*(RS_Ids_00300, RS_Ids_00301)*

[SWS_AIDSM_00302]{DRAFT} **Filter chain evaluation** ⌈IdsM shall evaluate the filter chain after evaluating the reporting mode.⌋*(RS_Ids_00610)*

[SWS_AIDSM_00303]{DRAFT} **Possible Filters** ⌈Each filter chain may consist of the following filters:

- MachineState Filter
- Forward-Every-nth Filter
- Aggregation Filter

- Threshold Filter

⌋(*RS_Ids_00300*)

[SWS_AIDSM_00304]{DRAFT} **Filter chain configuration** ⌈Each filter can be activated by aggregating the respective Filter object at the SecurityEventFilterChain object in the model.⌋(*RS_Ids_00300*)

[SWS_AIDSM_00305]{DRAFT} **Filter chain order** ⌈IdsM shall evaluate all activated filter in the order MachineState Filter, Forward-Every-nth Filter, Aggregation Filter, Threshold Filter.⌋(*RS_Ids_00300*)

[SWS_AIDSM_00306]{DRAFT} **Dropping of SEvs** ⌈If the evaluation of one filter leads to dropping the SEv, IdsM shall not evaluate any additional filter.⌋(*RS_Ids_00300*)

After successful evaluation of the configured filter chain, we define the security event as qualified (QSEv).

7.4.1 Machine State Filter

[SWS_AIDSM_00401]{DRAFT} **Machine State Filter** ⌈If IdsM evaluates the Machine State Filter and the current machine state equals one of the states referenced by SecurityEventStateFilter.blockIfStateActiveAp, then IdsM shall drop the SEv.⌋(*RS_Ids_00320*)

7.4.2 Sampling Filter

[SWS_AIDSM_00501]{DRAFT} **Sampling Filter** ⌈If IdsM evaluates the sampling filter for a SEv, IdsM shall drop all the SEvs but every *n-th* per SecurityEventDefinition, where n is defined by SecurityEventOneEveryNFilter.n.⌋(*RS_Ids_00330*)

An implementation will typically maintain one counter per SecurityEventDefinition that will be incremented when an SEv of given type is evaluated by the sampling filter. If the counter equals n the SEv is not dropped and the counter is reset to 0.

[SWS_AIDSM_00502]{DRAFT} **Sampling Filter Initialization** ⌈IdsM shall initialize the sampling filter for a SEv so that the first received SEv per SecurityEventDefinition is forwarded.⌋(*RS_Ids_00330*) Example: SecurityEventOneEveryNFilter.n is set to 3 for a certain event type, then SEvs 1, 4, 7, ... will be forwarded by the IdsM (1 describing the first SEv reported after reset).

7.4.3 Aggregation Filter

All SEv of a given type occurring within a configured time interval are aggregated into one SEv with an additional counter information attached that indicates how often the event occurred in the time interval.

[SWS_AIDSM_00600]{DRAFT} **Configuration of Aggregation Filter** ⌈The integrator shall configure the parameter SecurityEventAggregationFilter.aggregationIntervalLength to be the duration of the interval during which SEvs of the given type shall be aggregated.⌋*(RS_Ids_00340)*

[SWS_AIDSM_00601]{DRAFT} **No Event Forwarding During Interval** ⌈The aggregation filter shall not forward (i.e., to the next filter) any incoming SEv during the aggregation interval.⌋*(RS_Ids_00340)*

At the end of each aggregation interval, the aggregation filter shall implement the following logic for each Security Event Type:

[SWS_AIDSM_00602]{DRAFT} **End of Interval: No Event** ⌈If no SEv of the same event type has been received by the aggregation filter in the past aggregation interval, no action shall be taken.⌋*(RS_Ids_00340)*

[SWS_AIDSM_00603]{DRAFT} **End of Interval: One or More Events** ⌈If one or more SEv of the same event type have been received by the aggregation filter in the past aggregation interval, a SEv shall be forwarded to the next filter in the chain.⌋*(RS_Ids_00340)*

[SWS_AIDSM_00604]{DRAFT} **End of Interval: Count** ⌈If the SEv is forwarded to the next filter in the filter chain, the count parameter of the SEv shall equal the sum of all count parameters of all SEvs of given event type processed by the aggregation filter in the past time interval.⌋*(RS_Ids_00340)*

[SWS_AIDSM_00605]{DRAFT} **End of Interval: First Context Data** ⌈If the SEv is forwarded to the next filter in the filter chain and if SecurityEventAggregationFilter.contextDataSource equals IDSM_FILTERS_CTX_USE_FIRST, then the context data shall equal the first context data of an SEv of given type that has been received at the aggregation filter in the past time interval.⌋*(RS_Ids_00340)*

[SWS_AIDSM_00606]{DRAFT} **End of Interval: Last Context Data** ⌈If the SEv is forwarded to the next filter in the filter chain and if SecurityEventAggregationFilter.contextDataSource equals IDSM_FILTERS_CTX_USE_LAST, then the context data shall equal the last context data of an SEv of given type that has been received at the aggregation filter in the past time interval.⌋*(RS_Ids_00340)*

[SWS_AIDSM_00607]{DRAFT} **End of Interval: Timestamp** ⌈If the SEv is forwarded to the next filter in the filter chain, the timestamp shall be taken from the same SEv from which the context data comes from (configured via SecurityEventAggregationFilter.contextDataSource).⌋*(RS_Ids_00340)*

Please note that if SecurityEventAggregationFilter.contextDataSource equals IDSM_FILTERS_CTX_USE_LAST, then the reported or stored QSEv will contain the context data of the *last* SEv created in the configured time interval but the timestamp of the *first* SEv created in the configured time interval.

7.4.4 Threshold Filter

[SWS_AIDSM_00701]{DRAFT} **Event Dropping Below Threshold** ⌈The threshold filter shall drop an SEv of given type if the sum of count parameters of all SEvs of given type that were processed by the threshold filter in the current threshold interval is smaller than the configured parameter SecurityEventThresholdFilter.thresholdNumber.⌋*(RS_Ids_00350)*

[SWS_AIDSM_00702]{DRAFT} **Event Forwarding Above Threshold** ⌈The threshold filter shall forward an SEv of given type if the sum of count parameters of all SEvs of given type that were processed by the threshold filter in the current threshold interval is equal to or greater than the configured parameter SecurityEventThresholdFilter.thresholdNumber.⌋*(RS_Ids_00350)*

7.4.5 Qualification

After a SEv has successfully passed the last configured filter of the filter chain, it is considered a QSEv. Depending on the configuration, the QSEv can be transmitted to the IdsR and/or persisted locally.

7.5 Timestamp

Timestamps are optional and can be provided to the IdsM in different ways.

[SWS_AIDSM_00801]{DRAFT} **Timestamps are optional** ⌈If IdsmInstance.timestampFormat is not set, IdsM shall not add a timestamp to a QSEv and shall ignore timestamps provided via the timestamp parameter of the event reporting interface.⌋*(RS_Ids_00502)*

[SWS_AIDSM_00802]{DRAFT} **Timestamps provided by the stack** ⌈If IdsmInstance.timestampFormat equals "'AUTOSAR"' and the ara::idsm::EventReporter::ReportEvent function is called without a timestamp parameter, then Idsm shall add a timestamp from the TimeSync::TimeBaseResource referenced as IdsPlatformInstantiation.timeBase to stored and transmitted QSEvs.⌋*(RS_Ids_00503)*

The format of the timestamp to be added is specified in [2].

[SWS_AIDSM_00803]{DRAFT} **Timestamp provided via event reporting interface** ⌈If IdsmInstance.timestampFormat is set and the ara::idsm::EventReporter::ReportEvent function is called with a timestamp parameter, then Idsm shall use this provided timestamp parameter for transmission or storage of the QSEv.⌋ *(RS_Ids_00503)*

[SWS_AIDSM_00804]{DRAFT} **Timestamp provided via application software** ⌈If IdsmInstance.timestampFormat does not equal "'AUTOSAR"' and the ara::idsm::EventReporter::ReportEvent function is called without a timestamp parameter, then IdsM shall add a timestamp that is provided by a application software through the TimestampProvider callback to the QSEv.⌋*(RS_Ids_00503)*

[SWS_AIDSM_00805]{DRAFT} **Timestamp configured but not provided** ⌈If IdsmInstance.timestampFormat does not equal "'AUTOSAR"', but the ara::idsm::EventReporter::ReportEvent function is called without a timestamp parameter and no TimestampProvider has been registered, then IdsM shall not add a timestamp to the QSEv.⌋*(RS_Ids_00503)*

[SWS_AIDSM_00806]{DRAFT} **Truncation of timestamp parameter** ⌈If the ara::idsm::EventReporter::ReportEvent function is called with a timestamp parameter, then IdsM shall truncate this value by the 2 most-significant bits, i.e., only keep the 62 least-significant bits for further use.⌋*(RS_Ids_00503)*

[SWS_AIDSM_00807]{DRAFT} **Timestamp Provider** ⌈The TimestampProvider SWCL shall register a callback using the function ara::idsm::RegisterTimestampProvider. The callback shall return a timestamp.⌋*(RS_Ids_00503)*

Please note that while the TimestampProvider API is specified, the integration and configuration of the TimestampProvider remains stack-vendor specific.

7.6 Propagation of QSEvs

[SWS_AIDSM_00901]{DRAFT} **QSEv transmission** ⌈If a PlatformModuleEthernetEndpointConfiguration is aggregated at the IdsPlatformInstantiation in the role networkInterface, IdsM shall transmit QSEvs using the IDS protocol defined in [2] to the endpoint configured via the PlatformModuleEthernetEndpointConfiguration.⌋*(RS_Ids_00510)*

[SWS_AIDSM_00902]{DRAFT} **Message ID** ⌈IdsM shall set the Message ID field of the IDS Message Separation Header to all zero (0x00000000).⌋*(RS_Ids_00510)*

7.7 Authenticity of Transmitted QSevs

IdsM can optionally protect the authenticity of transmitted QSEvs using cryptographic signatures.

[SWS_AIDSM_01001]{DRAFT} **Signing QSEv** ⌈If an IdsmSignatureSupportAp is aggregated at the IdsmInstance in the role signatureSupportAp, then IdsM shall attach a cryptographic signature to each QSEv transmitted to the IdsR and to each locally persisted QSEv.⌋*(RS_Ids_00505)*

Over which data the signature shall be computed and how the signature shall be included in the message transmitted to the IdsR is specified in [2]. Which signature primitive and which key shall be used can be configured in using the IdsmSignatureSupportAp model element:

[SWS_AIDSM_01002]{DRAFT} **Primitive and Key** ⌈IdsM shall use the signing algorithm specified in the parameter IdsmSignatureSupportAp.cryptoPrimitive and the key identified by the CryptoKeySlot that is referenced by IdsmSignatureSupportAp in the role keySlot.⌋*(RS_Ids_00505)*

The naming scheme for the signature algorithm to be used is specified in SWS Cryptography [5].

7.8 Rate & Traffic Limitation

[SWS_AIDSM_01101]{DRAFT} **Rate and Traffic Limitation** ⌈Before sending a QSEv to the IdsR, IdsM shall apply rate and traffic limitation that can lead to dropping the QSEv.⌋*(RS_Ids_00511)*

[SWS_AIDSM_01103]{DRAFT} **Rate Limitation** ⌈IdsM shall drop an QSEv from transmission, if its transmission would cause the number of QSEvs transmitted in the current interval, which is specified in IdsmRateLimitation.timeInterval, to exceed the maximum number of transmission configured as IdsmRateLimitation.maxEventsInInterval.⌋*(RS_Ids_00511)*

[SWS_AIDSM_01104]{DRAFT} **Traffic Limitation** ⌈IdsM shall drop an QSEv from transmission, if its transmission would cause the number of bytes transmitted in the current interval, which is specified in IdsmTrafficLimitation.timeInterval, to exceed the maximum number of bytes configured as IdsmTrafficLimitation.maxBytesInInterval.⌋*(RS_Ids_00511)*

7.9 Access Control

The generation of security events is subject to access control, i.e., which event types can be generated by a specific SWCL can be limited through configuration. Access Control is enforced by IAM on the Adaptive Platform.

[SWS_AIDSM_01201]{DRAFT} ⌈IdsM shall restrict the event types a Process can generate to those SecurityEventDefinitions referenced by the Process in the role securityEvent in the manifest.⌋*(RS_Ids_00200)*

The TimestampProvider interface also needs to be subject to access control in order to prevent malicious or compromised applications from providing wrong timestamps to the IdsM In order to support project specific TimestampProvider (e.g., hardware or driver-based), access control to the TimestampProvider is out of scope of this specification and has to be enforced in a project-specific way.

7.10 Diagnostic Access

IdsM allows diagnostic access to support two use-cases: First, persisted events can be read via diagnostic access. Second, a reconfiguration of the reporting mode via diagnostic access is possible.

7.10.1 Access to Persisted Events

Each security event references a diagnostic event, which in turn references a DTC.

[SWS_AIDSM_01301]{DRAFT} **Access to Persisted Events** ⌈If an event has been successfully qualified and the event is configured to be persisted (i.e., SecurityEventContextProps.persistentStorage == 1), then IdsM shall qualify the DTC referenced by the event and add the event data as a snapshot record to it.⌋*(RS_Ids_00400)*

7.10.2 Reconfiguration of Reporting Mode

IdsM standardizes a DID for reading and changing the reporting mode of events during runtime.

[SWS_AIDSM_01302]{DRAFT} **Get current reporting mode** ⌈IdsM shall provide a diagnostic service GetReportingMode(SecurityEventDefinition.id) that returns the current reporting mode of the queried SecurityEventDefinition.⌋*(RS_Ids_00700)*

[SWS_AIDSM_01303]{DRAFT} **Set current reporting mode** ⌈IdsM shall provide a diagnostic service SetReportingMode(SecurityEventDefinition.id, ReportingMode) that sets the reporting mode of the given SecurityEventDefinition.⌋*(RS_Ids_00700)*

7.11 IdsM Provided SEvs

IdsM itself can also be used as a Security Event sensor.

[SWS_AIDSM_01401]{DRAFT} **IdsM Provided SEvs** ⌈The security events reported by IdsM module are listed in [SWS_IdsM_91015].⌋*(RS_Ids_00820)*

Note that the hexadecimal value that corresponds to every `Security Event` is centrally defined in the SecXT.

[SWS_IdsM_91015] Security events for IDSM ⌈

Name	*Description*	*ID*
IDSM_INTERNAL_EVENT_NO_EVENT_BUFFER_AVAILABLE	A SEv cannot be handled because there are no more event buffers available to process the event.	46
IDSM_INTERNAL_EVENT_NO_CONTEXT_DATA_BUFFER_AVAILABLE	The context data of an incoming event cannot be stored because there are no more context data buffers available.	47
IDSM_INTERNAL_EVENT_TRAFFIC_LIMITATION_EXCEEDED	The current traffic exceeds a configured traffic limitation.	48
IDSM_INTERNAL_EVENT_COMMUNICATION_ERROR	An error occurred when sending a QSEv via PDU.	49

⌋*(RS_Ids_00810)*

Please note that the term `buffer` refers to the memory in which event and context data is stored, independent of the concrete implementation.

[SWS_AIDSM_01402]{DRAFT} **Buffer availability** ⌈`IdsM` shall ensure that `IdsM` internal events can be processed even though no buffers are available.⌋*(RS_Ids_00820)* An implementation could achieve this by, e.g., pre-allocating memory buffers for `IdsM` provided events.

[SWS_AIDSM_01403]{DRAFT} **Bypass limitation filter** ⌈`IdsM` internal `SEv`s shall not be filtered by rate and traffic limitation filter.⌋*(RS_Ids_00820)*

8 API specification

8.1 API Common Data Types

[SWS_AIDSM_10201]{DRAFT} ⌈

Kind:	type alias
Symbol:	ContextDataType
Scope:	namespace ara::idsm
Derived from:	ara::core::Span<std::uint8_t>
Syntax:	`using ContextDataType = ara::core::Span<std::uint8_t>;`
Header file:	#include "ara/idsm/common.h"
Description:	ContextDataType used for sending context data to the IdsM .

⌋*()*

[SWS_AIDSM_10202]{DRAFT} ⌈

Kind:	type alias
Symbol:	TimestampType
Scope:	namespace ara::idsm
Derived from:	std::uint64_t
Syntax:	`using TimestampType = std::uint64_t;`
Header file:	#include "ara/idsm/common.h"
Description:	TimestampType used for setting optional sensor-specific timestamp for events.
Notes:	Only 62 least-significant bits are used as timestamp value and stored or transmitted, respectively

⌋()

[SWS_AIDSM_10203]{DRAFT} ⌈

Kind:	type alias
Symbol:	CountType
Scope:	namespace ara::idsm
Derived from:	std::uint16_t
Syntax:	`using CountType = std::uint16_t;`
Header file:	#include "ara/idsm/common.h"
Description:	CountType used for setting optional count for events pre-qualified by sensors .

⌋()

8.2 API Reference

8.2.1 EventReporter

[SWS_AIDSM_10101]{DRAFT} ⌈

Kind:	class
Symbol:	EventReporter
Scope:	namespace ara::idsm
Syntax:	`class EventReporter {...};`
Header file:	#include "ara/idsm/event_reporter.h"
Description:	Class for reporting security events to the IdsM .

⌋()

[SWS_AIDSM_10301]{DRAFT} ⌈

Kind:	function
Symbol:	EventReporter(const ara::core::InstanceSpecifier &eventType)

▽

△

Scope:	class ara::idsm::EventReporter	
Syntax:	`EventReporter (const ara::core::InstanceSpecifier &eventType) noexcept;`	
Parameters (in):	eventType	InstanceSpecifier of the EventDefinition to be reported by this EventReporter object
Exception Safety:	noexcept	
Header file:	#include "ara/idsm/event_reporter.h"	
Description:	Construct a new Event Reporter object. Called by the sensor for each event type using the instance specified of the event type .	

⌋()

[SWS_AIDSM_10302]{DRAFT} ⌈

Kind:	function	
Symbol:	ReportEvent(const CountType=1)	
Scope:	class ara::idsm::EventReporter	
Syntax:	`void ReportEvent (const CountType=1) noexcept;`	
DIRECTION NOT DEFINED	CountType	–
Return value:	None	
Exception Safety:	noexcept	
Header file:	#include "ara/idsm/event_reporter.h"	
Description:	Create a new security event at the IdsM. .	

⌋()

[SWS_AIDSM_10303]{DRAFT} ⌈

Kind:	function	
Symbol:	ReportEvent(const TimestampType timestamp, const CountType=1)	
Scope:	class ara::idsm::EventReporter	
Syntax:	`void ReportEvent (const TimestampType timestamp, const CountType=1) noexcept;`	
Parameters (in):	timestamp	application provided timestamp
DIRECTION NOT DEFINED	CountType	–
Return value:	None	
Exception Safety:	noexcept	
Header file:	#include "ara/idsm/event_reporter.h"	
Description:	Create a new security event with a sensor-provided timestamp at the IdsM. .	

⌋()

[SWS_AIDSM_10304]{DRAFT} ⌈

Kind:	function	
Symbol:	ReportEvent(const ContextDataType &contextData, const CountType=1)	
Scope:	class ara::idsm::EventReporter	
Syntax:	`void ReportEvent (const ContextDataType &contextData, const CountType=1) noexcept;`	
Parameters (in):	contextData	context data
DIRECTION NOT DEFINED	CountType	–
Return value:	None	
Exception Safety:	noexcept	
Header file:	#include "ara/idsm/event_reporter.h"	
Description:	Create a new security event with sensor-provided context data at the IdsM. .	

⌋()

[SWS_AIDSM_10305]{DRAFT} ⌈

Kind:	function	
Symbol:	ReportEvent(const ContextDataType &contextData, const TimestampType timestamp, const CountType=1)	
Scope:	class ara::idsm::EventReporter	
Syntax:	`void ReportEvent (const ContextDataType &contextData, const TimestampType timestamp, const CountType=1) noexcept;`	
Parameters (in):	contextData	context data
	timestamp	application provided timestamp
DIRECTION NOT DEFINED	CountType	–
Return value:	None	
Exception Safety:	noexcept	
Header file:	#include "ara/idsm/event_reporter.h"	
Description:	Create a new security event with sensor-provided context data and with a sensor-provided timestamp at the IdsM. .	

⌋()

8.2.2 TimestampProvider

[SWS_AIDSM_20101]{DRAFT} ⌈

Kind:	function	
Symbol:	RegisterTimestampProvider(std::function< TimestampType()> callback)	
Scope:	namespace ara::idsm	
Syntax:	`void RegisterTimestampProvider (std::function< TimestampType()> callback);`	
Parameters (in):	callback	std::function callback that provides a timestamp to the IdsM

▽

△

Return value:	None
Header file:	#include "ara/idsm/timestamp_provider.h"
Description:	Register a callback for providing timestamps to the IdsM .

⌋()

9 Service Interfaces

IdsM does not provide any service interfaces.

A Mentioned Manifest Elements

For the sake of completeness, this chapter contains a set of class tables representing meta-classes mentioned in the context of this document but which are not contained directly in the scope of describing specific meta-model semantics.

Chapter is generated.

Class	**CryptoKeySlot**			
Package	M2::AUTOSARTemplates::AdaptivePlatform::PlatformModuleDeployment::CryptoDeployment			
Note	This meta-class represents the ability to define a concrete key to be used for a crypto operation. **Tags:** atp.ManifestKind=MachineManifest atp.Status=draft			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
allocateShadow Copy	Boolean	0..1	attr	This attribute defines whether a shadow copy of this Key Slot shall be allocated to enable rollback of a failed Key Slot update campaign (see interface BeginTransaction). **Tags:**atp.Status=draft
cryptoAlgId	String	0..1	attr	This attribute defines a crypto algorithm restriction (kAlgId Any means without restriction). The algorithm can be specified partially: family & length, mode, padding. Future Crypto Providers can support some crypto algorithms that are not well known/ standardized today, therefore AUTOSAR doesn't provide a concrete list of crypto algorithms' identifiers and doesn't suppose usage of numerical identifiers. Instead of this a provider supplier should provide string names of supported algorithms in accompanying documentation. The name of a crypto algorithm shall follow the rules defined in the specification of cryptography for Adaptive Platform. **Tags:**atp.Status=draft
cryptoObject Type	CryptoObjectTypeEnum	0..1	attr	Object type that can be stored in the slot. If this field contains "Undefined" then mSlotCapacity must be provided and larger then 0. **Tags:**atp.Status=draft
keySlotAllowed Modification	CryptoKeySlotAllowed Modification	0..1	aggr	Restricts how this keySlot may be used **Tags:**atp.Status=draft
keySlotContent AllowedUsage	CryptoKeySlotContent AllowedUsage	*	aggr	Restriction of allowed usage of a key stored to the slot. **Tags:**atp.Status=draft
slotCapacity	PositiveInteger	0..1	attr	Capacity of the slot in bytes to be reserved by the stack vendor. One use case is to define this value in case that the cryptoObjectType is undefined and the slot size can not be deduced from cryptoObjectType and cryptoAlgId. "0" means slot size can be deduced from cryptoObject Type and cryptoAlgId. **Tags:**atp.Status=draft
slotType	CryptoKeySlotType Enum	0..1	attr	This attribute defines whether the keySlot is exclusively used by the Application; or whether it is used by Stack Services and managed by a Key Manager Application. **Tags:**atp.Status=draft

Table A.1: CryptoKeySlot

Class	***IdsPlatformInstantiation*** (abstract)			
Package	M2::AUTOSARTemplates::AdaptivePlatform::PlatformModuleDeployment::IntrusionDetectionSystem			
Note	This meta-class acts as an abstract base class for platform modules that implement the intrusion detection system. **Tags:**atp.Status=draft			
Base	*ARObject*, *AdaptiveModuleInstantiation*, *Identifiable*, *MultilanguageReferrable*, *NonOsModule Instantiation*, *Referrable*			
Subclasses	IdsmModuleInstantiation			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
network Interface	PlatformModule EthernetEndpoint Configuration	0..1	ref	This association contains the network configuration that shall be applied to an instance of an IDS entity. **Tags:**atp.Status=draft
timeBase	TimeBaseResource	0..1	ref	This reference identifies the applicable time base resource. **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=systemDesignTime

Table A.2: IdsPlatformInstantiation

Class	**IdsmInstance**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class provides the ability to create a relation between an EcuInstance and a specific class of filters for security events that apply for all security events reported on the referenced EcuInstance. **Tags:** atp.Status=draft atp.recommendedPackage=IdsmInstanceToEcuInstanceMappings			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *IdsCommonElement*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
idsmInstanceId	PositiveInteger	0..1	attr	This attribute is used to provide a source identification in the context of reporting security events.. **Tags:**atp.Status=draft
idsmModule Instantiation	IdsmModule Instantiation	0..1	ref	This reference identifies the meta-class that defines the attributes for the IdsM configuration on a specific machine. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=idsmModuleInstantiation atp.Status=draft
rateLimitation Filter	IdsmRateLimitation	0..1	ref	This reference identifies the applicable rate limitation filter for all security events on the related EcuInstance. **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=preCompileTime

△

Class	**IdsmInstance**			
signature SupportAp	IdsmSignatureSupportAp	0..1	aggr	The existence of this aggregation specifies that the IdsM shall add a signature to the QSEv messages it sends onto the network. The cryptographic algorithm and key to be used for this signature is further specified by the aggregated meta-class specifically for the Adaptive Platform. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=signatureSupportAp atp.Status=draft
timestamp Format	String	0..1	attr	The existence of this attribute specifies that the IdsM shall add a timestamp to the QSEv messages it sends onto the network. I.e., if this attribute does not exist, no timestamp shall be added to the QSEv messages. The content of this attribute further specifies the timestamp format as follows: - "AUTOSAR" defines AUTOSAR standardized timestamp format according to the Synchronized Time-Base Manager - Any other string defines a proprietary timestamp format. Note: A string defining a proprietary timestamp format shall be prefixed by a company-specific name fragment to avoid collisions. **Tags:**atp.Status=draft
trafficLimitation Filter	IdsmTrafficLimitation	0..1	ref	This reference identifies the applicable traffic limitation filter for all security events on the related EcuInstance. **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=preCompileTime

Table A.3: IdsmInstance

Class	**IdsmRateLimitation**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents the configuration of a rate limitation filter for security events. This means that security events are dropped if the number of events (of any type) processed within a configurable time window is greater than a configurable threshold. **Tags:**atp.Status=draft			
Base	*ARObject*, *AbstractSecurityIdsmInstanceFilter*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
maxEventsIn Interval	PositiveInteger	1	attr	This attribute configures the threshold for dropping security events if the number of all processed security events exceeds the threshold in the respective time interval. **Tags:**atp.Status=draft
timeInterval	Float	1	attr	This attribute configures the length of the time interval in seconds for dropping security events if the number of all processed security events exceeds the configurable threshold within the respective time interval. **Tags:**atp.Status=draft

Table A.4: IdsmRateLimitation

Class	**IdsmSignatureSupportAp**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class defines, for the Adaptive Platform, the cryptographic algorithm and key to be used by the IdsM instance for providing signature information in QSEv messages. **Tags:**atp.Status=draft			
Base	*ARObject*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
cryptoPrimitive	String	1	attr	This attribute defines the cryptographic algorithm to be used for providing authentication information in QSEv messages. The content of this attribute shall comply to the "Cryptographic Primitives Naming Convention". **Tags:**atp.Status=draft
keySlot	CryptoKeySlot	0..1	ref	This reference denotes the cryptographic key to be used by the cryptographic algorithm for providing authentication information in QSEv messages. **Tags:**atp.Status=draft

Table A.5: IdsmSignatureSupportAp

Class	**IdsmTrafficLimitation**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents the configuration of a traffic limitation filter for Security Events. This means that security events are dropped if the size (in terms of bandwidth) of security events (of any type) processed within a configurable time window is greater than a configurable threshold. **Tags:**atp.Status=draft			
Base	*ARObject*, *AbstractSecurityIdsmInstanceFilter*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
maxBytesIn Interval	PositiveInteger	0..1	attr	This attribute configures the threshold for dropping security events if the size of all processed security events exceeds the threshold in the respective time interval. **Tags:**atp.Status=draft
timeInterval	Float	0..1	attr	This attribute configures the length of the time interval in seconds for dropping security events if the size of all processed security events exceeds the configurable threshold within the respective time interval. **Tags:**atp.Status=draft

Table A.6: IdsmTrafficLimitation

Class	**PlatformModuleEthernetEndpointConfiguration**			
Package	M2::AUTOSARTemplates::AdaptivePlatform::PlatformModuleDeployment::AdaptiveModule Implementation			
Note	This meta-class defines the attributes for the configuration of a port, protocol type and IP address of the communication on a VLAN. **Tags:** atp.Status=draft atp.recommendedPackage=PlatformModuleEndpointConfigurations			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *Packageable Element*, *PlatformModuleEndpointConfiguration*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***

△

Class	**PlatformModuleEthernetEndpointConfiguration**			
communication Connector	EthernetCommunication Connector	0..1	ref	Reference to the CommunicationConnector (VLAN) for which the network configuration is defined. **Tags:**atp.Status=draft
ipv4MulticastIp Address	Ip4AddressString	0..1	attr	Multicast IPv4 Address to which the message will be transmitted. **Tags:**atp.Status=draft
ipv6MulticastIp Address	Ip6AddressString	0..1	attr	Multicast IPv6 Address to which the message will be transmitted. **Tags:**atp.Status=draft
tcpPort	ApApplicationEndpoint	0..1	ref	This reference allows to configure a tcp port number. **Tags:**atp.Status=draft
udpPort	ApApplicationEndpoint	0..1	ref	This reference allows to configure a udp port number. **Tags:**atp.Status=draft

Table A.7: PlatformModuleEthernetEndpointConfiguration

Class	**Process**			
Package	M2::AUTOSARTemplates::AdaptivePlatform::ExecutionManifest			
Note	This meta-class provides information required to execute the referenced executable. **Tags:** atp.Status=draft atp.recommendedPackage=Processes			
Base	*ARElement*, *ARObject*, *AbstractExecutionContext*, *AtpClassifier*, *CollectableElement*, *Identifiable*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*, *UploadablePackageElement*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
design	ProcessDesign	0..1	ref	This reference represents the identification of the design-time representation for the Process that owns the reference. **Tags:**atp.Status=draft
deterministic Client	DeterministicClient	0..1	ref	This reference adds further execution characteristics for deterministic clients. **Tags:**atp.Status=draft
executable	Executable	0..1	ref	Reference to executable that is executed in the process. **Stereotypes:** atpUriDef **Tags:**atp.Status=draft
functionCluster Affiliation	String	0..1	attr	This attribute specifies which functional cluster the process is affiliated with. **Tags:**atp.Status=draft
numberOf RestartAttempts	PositiveInteger	0..1	attr	This attribute defines how often a process shall be restarted if the start fails. numberOfRestartAttempts = "0" OR Attribute not existing, start once numberOfRestartAttempts = "1", start a second time **Tags:**atp.Status=draft
preMapping	Boolean	0..1	attr	This attribute describes whether the executable is preloaded into the memory. **Tags:**atp.Status=draft

▽

△

Class	**Process**			
processState Machine	ModeDeclarationGroup Prototype	0..1	aggr	Set of Process States that are defined for the process. **Tags:**atp.Status=draft
securityEvent	SecurityEventDefinition	*	ref	The reference identifies the collection of SecurityEvents that can be reported by the enclosing SoftwareCluster. **Stereotypes:** atpSplitable; atpUriDef **Tags:** atp.Splitkey=securityEvent atp.Status=draft
stateDependent StartupConfig	StateDependentStartup Config	*	aggr	Applicable startup configurations. **Tags:**atp.Status=draft

Table A.8: Process

Class	**SecurityEventAggregationFilter**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents the aggregation filter that aggregates all security events occurring within a configured time frame into one (i.e. the last reported) security event. **Tags:**atp.Status=draft			
Base	*ARObject*, *AbstractSecurityEventFilter*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
contextData Source	SecurityEventContext DataSourceEnum	0..1	attr	This attributes defines whether the context data of the first or last time-aggregated security event shall be used for the resulting qualified security event.
minimum IntervalLength	TimeValue	0..1	attr	This attribute represents the configuration of the minimum time window in seconds for the aggregation filter. **Tags:**atp.Status=draft

Table A.9: SecurityEventAggregationFilter

Class	***SecurityEventContextMapping*** (abstract)			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents the ability to create an association between a collection of security events, an IdsM instance which handles the security events and the filter chains applicable to the security events. **Tags:**atp.Status=draft			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *IdsCommonElement*, *IdsMapping*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Subclasses	SecurityEventContextMappingApplication, SecurityEventContextMappingCommConnector, Security EventContextMappingFunctionalCluster			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
filterChain	SecurityEventFilter Chain	0..1	ref	This reference defines the filter chain to be applied to each of the referenced security events (depending on the reporting mode). **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=preCompileTime

▽

△

Class	***SecurityEventContextMapping*** (abstract)			
idsmInstance	IdsmInstance	0..1	ref	This reference defines the IdsmInstance onto which the security events are mapped. **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=systemDesignTime
mappedSecurity Event	SecurityEventContext Props	*	aggr	This aggregation represents (through further references) the SecurityEventDefinitions to be mapped to an Idsm Instance with additional mapping-dependent properties. **Stereotypes:** atpSplitable; atpVariation **Tags:** atp.Splitkey=mappedSecurityEvent.shortName, mapped SecurityEvent.variationPoint.shortLabel atp.Status=draft vh.latestBindingTime=preCompileTime

Table A.10: SecurityEventContextMapping

Class	**SecurityEventContextProps**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class specifies the SecurityEventDefinition to be mapped to an IdsmInstance and adds mapping-dependent properties of this security event valid only for this specific mapping. **Tags:**atp.Status=draft			
Base	*ARObject*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
contextData	SecurityEventContext Data	0..1	aggr	This aggregation represents the definition of optional context data for security events. **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=systemDesignTime
default ReportingMode	SecurityEventReporting ModeEnum	0..1	attr	This attribute defines the default reporting mode for the referenced security event. **Tags:**atp.Status=draft
persistent Storage	Boolean	0..1	attr	This attribute controls whether qualified reportings of the referenced security event shall be stored persistently by the mapped IdsmInstance or not. **Tags:**atp.Status=draft
securityEvent	SecurityEventDefinition	0..1	ref	This reference defines the security event that is mapped and enriched by SecurityEventMappingProps with mapping dependent properties. **Stereotypes:** atpVariation **Tags:** atp.Status=draft vh.latestBindingTime=systemDesignTime
sensorInstance Id	PositiveInteger	0..1	attr	This attribute defines the ID of the security sensor that detects the referenced security event. **Tags:**atp.Status=draft

▽

△

Class	SecurityEventContextProps			
severity	PositiveInteger	0..1	attr	This attribute defines how critical/severe the referenced security event is. Please note that currently, the severity level meanings of specific integer values is not specified by AUTOSAR but left to the party responsible for the IDS system design (e.g. the OEM). **Tags:**atp.Status=draft

Table A.11: SecurityEventContextProps

Class	SecurityEventDefinition			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class defines a security-related event as part of the intrusion detection system. **Tags:** atp.Status=draft atp.recommendedPackage=SecurityEventDefinitions			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *IdsCommonElement*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
eventSymbol Name	SymbolProps	0..1	aggr	This aggregation defines optionally an alternative Event Name for the SecurityEventDefinition in case there is a collision of shortNames. **Stereotypes:** atpSplitable **Tags:** atp.Splitkey=eventSymbolName.shortName atp.Status=draft
id	PositiveInteger	0..1	attr	This attribute represents the numerical identification of the defined security event. The identification shall be unique within the scope of the IDS. **Tags:**atp.Status=draft

Table A.12: SecurityEventDefinition

Class	SecurityEventFilterChain			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents a configurable chain of filters used to qualify security events. The different filters of this filter chain are applied in the follow order: SecurityEventStateFilter, SecurityEventOneEvery NFilter, SecurityEventAggregationFilter, SecurityEventThresholdFilter. **Tags:** atp.Status=draft atp.recommendedPackage=SecurityFilterChains			
Base	*ARElement*, *ARObject*, *CollectableElement*, *Identifiable*, *IdsCommonElement*, *MultilanguageReferrable*, *PackageableElement*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
aggregation	SecurityEvent AggregationFilter	0..1	aggr	This aggregation represents the aggregation filter in the filter chain. **Tags:**atp.Status=draft
oneEveryN	SecurityEventOneEvery NFilter	0..1	aggr	This aggregation represents the sampling filter in the filter chain. **Tags:**atp.Status=draft

▽

△

Class	**SecurityEventFilterChain**			
state	SecurityEventStateFilter	0..1	aggr	This aggregation represents the state filter in the event chain. **Tags:**atp.Status=draft
threshold	SecurityEventThreshold Filter	0..1	aggr	This aggregation represents the threshold filter in the filter chain. **Tags:**atp.Status=draft

Table A.13: SecurityEventFilterChain

Class	**SecurityEventOneEveryNFilter**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents the configuration of a sampling (i.e. every n-th event is sampled) filter for security events. **Tags:**atp.Status=draft			
Base	*ARObject*, *AbstractSecurityEventFilter*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
n	PositiveInteger	0..1	attr	This attribute represents the configuration of the sampling filter, i.e. it configures the parameter "n" that controls how many events (n-1) shall be dropped after a sampled event until a new sample is created. **Tags:**atp.Status=draft

Table A.14: SecurityEventOneEveryNFilter

Class	**SecurityEventStateFilter**			
Package	M2::AUTOSARTemplates::SecurityExtractTemplate			
Note	This meta-class represents the configuration of a state filter for security events. The referenced states represent a block list, i.e. the security events are dropped if the referenced state is the active state in the relevant state machine (which depends on whether the IdsM instance runs on the Classic or the Adaptive Platform). **Tags:**atp.Status=draft			
Base	*ARObject*, *AbstractSecurityEventFilter*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
blockIfState ActiveAp	ModeDeclaration	*	iref	For the AP, this reference defines the machine states of the block list. That means, if a security event (mapped to the filter chain to which the SecurityEventStateFilter belongs to) is reported when the machine is in one of the block listed states, the IdsM shall discard the reported security event. **Tags:**atp.Status=draft **InstanceRef implemented by:**FunctionGroupStateIn FunctionGroupSetInstanceRef

Table A.15: SecurityEventStateFilter

Class	**SecurityEventThresholdFilter**
Package	M2::AUTOSARTemplates::SecurityExtractTemplate

▽

△

Class	**SecurityEventThresholdFilter**			
Note	This meta-class represents the threshold filter that drops (repeatedly at each beginning of a configurable time interval) a configurable number of security events . All subsequently arriving security events (within the configured time interval) pass the filter. **Tags:**atp.Status=draft			
Base	*ARObject*, *AbstractSecurityEventFilter*, *Identifiable*, *MultilanguageReferrable*, *Referrable*			
Attribute	***Type***	***Mult.***	***Kind***	***Note***
intervalLength	TimeValue	0..1	attr	This attribute configures the time interval in seconds for one threshold filter operation. **Tags:**atp.Status=draft
threshold Number	PositiveInteger	0..1	attr	This attribute configures the threshold number, i.e. how many security events in the configured time frame are dropped before subsequent events start to pass the filter. **Tags:**atp.Status=draft

Table A.16: SecurityEventThresholdFilter

B Interfaces to other Functional Clusters (informative)

B.1 Overview

AUTOSAR decided not to standardize interfaces which are exclusively used between Functional Clusters (on platform-level only), to allow efficient implementations, which might depend e.g. on the used Operating System.

This chapter provides informative guidelines how the interaction between Functional Clusters looks like, by clustering the relevant requirements of this document to describe Inter-Functional Cluster (IFC) interfaces. In addition, the standardized public interfaces which are accessible by user space applications (see chapters 8 and 9) can also be used for interaction between Functional Clusters.

The goal is to provide a clear understanding of Functional Cluster boundaries and interaction, without specifying syntactical details. This ensures compatibility between documents specifying different Functional Clusters and supports parallel implementation of different Functional Clusters. Details of the interfaces are up to the platform provider. Additional interfaces, parameters and return values can be added.

B.2 Interface Tables