AT&T Service Specification

Service: *Vendor Event Listener*

|  |  |
| --- | --- |
| Document Number | xxxx |
| Revision | 3.1 |
| Revision Date | 12/1/2016 |
| Author | Rich Erickson |

© 2015, AT&T Intellectual Property.  All other rights reserved. AT&T and AT&T logos are trademarks of AT&T Intellectual Property. All marks, trademarks, and product names used in this document are the property of their respective owners.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:  This product includes software developed by the AT&T.
4. Neither the name of AT&T nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY AT&T INTELLECTUAL PROPERTY ''AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL AT&T INTELLECTUAL PROPERTY BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;  LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This document and the information contained herein (collectively, the "**Information**") is provided to you (both the individual receiving this document and any legal entity on behalf of which such individual is acting) ("**You**" and "**Your**") by AT&T, on behalf of itself and its affiliates ("**AT&T**") for informational purposes only. AT&T is providing the Information to You because AT&T believes the Information may be useful to You. The Information is provided to You solely on the basis that You will be responsible for making Your own assessments of the Information and are advised to verify all representations, statements and information before using or relying upon any of the Information. Although AT&T has exercised reasonable care in providing the Information to You, AT&T does not warrant the accuracy of the Information and is not responsible for any damages arising from Your use of or reliance upon the Information. You further understand and agree that AT&T in no way represents, and You in no way rely on a belief, that AT&T is providing the Information in accordance with any standard or service (routine, customary or otherwise) related to the consulting, services, hardware or software industries.

AT&T DOES NOT WARRANT THAT THE INFORMATION IS ERROR-FREE. AT&T IS PROVIDING THE INFORMATION TO YOU "AS IS" AND "WITH ALL FAULTS." AT&T DOES NOT WARRANT, BY VIRTUE OF THIS DOCUMENT, OR BY ANY COURSE OF PERFORMANCE, COURSE OF DEALING, USAGE OF TRADE OR ANY COLLATERAL DOCUMENT HEREUNDER OR OTHERWISE, AND HEREBY EXPRESSLY DISCLAIMS, ANY REPRESENTATION OR WARRANTY OF ANY KIND WITH RESPECT TO THE INFORMATION, INCLUDING, WITHOUT LIMITATION, ANY REPRESENTATION OR WARRANTY OF DESIGN, PERFORMANCE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, OR ANY REPRESENTATION OR WARRANTY THAT THE INFORMATION IS APPLICABLE TO OR INTEROPERABLE WITH ANY SYSTEM, DATA, HARDWARE OR SOFTWARE OF ANY KIND. AT&T DISCLAIMS AND IN NO EVENT SHALL BE LIABLE FOR ANY LOSSES OR DAMAGES OF ANY KIND, WHETHER DIRECT, INDIRECT, INCIDENTAL, CONSEQUENTIAL, PUNITIVE, SPECIAL OR EXEMPLARY, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, LOSS OF GOODWILL, COVER, TORTIOUS CONDUCT OR OTHER PECUNIARY LOSS, ARISING OUT OF OR IN ANY WAY RELATED TO THE PROVISION, NON-PROVISION, USE OR NON-USE OF THE INFORMATION, EVEN IF AT&T HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSSES OR DAMAGES.

.

[1 Introduction 1](#_Toc468361608)

[1.1 Audience 1](#_Toc468361609)

[1.2 Terminology 1](#_Toc468361610)

[1.3 Naming Standards for Event Types 2](#_Toc468361611)

[1.4 Support for Protocols Other Than HTTPS 2](#_Toc468361612)

[1.5 Versioning 2](#_Toc468361613)

[2 Security 4](#_Toc468361614)

[2.1.1 Sample Request and Response 4](#_Toc468361615)

[2.1.1.1 Sample Request 4](#_Toc468361616)

[2.1.1.2 Sample Success Response 4](#_Toc468361617)

[3 Resource Structure 6](#_Toc468361618)

[4 Generic Event Format 7](#_Toc468361619)

[4.1 Datatype: codecsInUse 7](#_Toc468361620)

[4.2 Datatype: codecSelected 7](#_Toc468361621)

[4.3 Datatype: codecSelectedTranscoding 7](#_Toc468361622)

[4.4 Datatype: command 7](#_Toc468361623)

[4.5 Datatype: commandList 8](#_Toc468361624)

[4.6 Datatype: commandListEntry 8](#_Toc468361625)

[4.7 Datatype: commonEventHeader 8](#_Toc468361626)

[4.8 Datatype: counter 9](#_Toc468361627)

[4.9 Datatype: cpuUsage 9](#_Toc468361628)

[4.10 Datatype: endOfCallVqmSummaries 9](#_Toc468361629)

[4.11 Datatype: errors 10](#_Toc468361630)

[4.12 Datatype: event 10](#_Toc468361631)

[4.13 Datatype: eventDomainThrottleSpecification 11](#_Toc468361632)

[4.14 Datatype: eventDomainThrottleSpecificationList 11](#_Toc468361633)

[4.15 Datatype: eventInstanceIdentifier 11](#_Toc468361634)

[4.16 Datatype: eventList 12](#_Toc468361635)

[4.17 Datatype: eventThrottlingState 12](#_Toc468361636)

[4.18 Datatype: faultFields 12](#_Toc468361637)

[4.19 Datatype: featuresInUse 13](#_Toc468361638)

[4.20 Datatype: field 13](#_Toc468361639)

[4.21 Datatype: filesystemUsage 13](#_Toc468361640)

[4.22 Datatype: gtpPerFlowMetrics 13](#_Toc468361641)

[4.23 Datatype: latencyBucketMeasure 16](#_Toc468361642)

[4.24 Datatype: marker 17](#_Toc468361643)

[4.25 Datatype: measurementForVfScalingFields 17](#_Toc468361644)

[4.26 Datatype: measurementGroup 18](#_Toc468361645)

[4.27 Datatype: mobileFlowFields 19](#_Toc468361646)

[4.28 Datatype: otherFields 20](#_Toc468361647)

[4.29 Datatype: requestError 20](#_Toc468361648)

[4.30 Datatype: serviceEventsFields 20](#_Toc468361649)

[4.31 Datatype: signalingFields 21](#_Toc468361650)

[4.32 Datatype: stateChangeFields 21](#_Toc468361651)

[4.33 Datatype: suppressedNvPairs 22](#_Toc468361652)

[4.34 Datatype: syslogFields 22](#_Toc468361653)

[4.35 Datatype: thresholdCrossingAlertFields 24](#_Toc468361654)

[4.36 Datatype: vNicUsage 25](#_Toc468361655)

[5 Exceptions 26](#_Toc468361656)

[5.1 RESTful Web Services Exceptions 26](#_Toc468361657)

[5.2 Service Exceptions 26](#_Toc468361658)

[5.3 Policy Exceptions 27](#_Toc468361659)

[6 RESTful Web Services Definition 29](#_Toc468361660)

[6.1 REST Operation Overview 29](#_Toc468361661)

[6.1.1 REST Operation Summary 29](#_Toc468361662)

[6.1.2 Api Version 29](#_Toc468361663)

[6.1.3 Commands Toward Event Source Clients 29](#_Toc468361664)

[6.2 Operation: publishAnyEvent 30](#_Toc468361665)

[6.2.1 Functional Behavior 30](#_Toc468361666)

[6.2.2 Call Flow 31](#_Toc468361667)

[6.2.3 Input Parameters 31](#_Toc468361668)

[6.2.4 Output Parameters 32](#_Toc468361669)

[6.2.5 HTTP Status Codes 33](#_Toc468361670)

[6.2.6 Sample Request and Response 33](#_Toc468361671)

[6.2.6.1 Sample Request 33](#_Toc468361672)

[6.2.6.2 Sample Success Response #1 34](#_Toc468361673)

[6.2.6.3 Sample Success Response #2 34](#_Toc468361674)

[6.2.6.4 Sample Error Responses 35](#_Toc468361675)

[6.3 Operation: publishSpecificTopic 36](#_Toc468361676)

[6.3.1 Functional Behavior 36](#_Toc468361677)

[6.3.2 Call Flow 37](#_Toc468361678)

[6.3.3 Input Parameters 37](#_Toc468361679)

[6.3.4 Output Parameters 38](#_Toc468361680)

[6.3.5 HTTP Status Codes 39](#_Toc468361681)

[6.3.6 Sample Request and Response 40](#_Toc468361682)

[6.3.6.1 Sample Request 40](#_Toc468361683)

[6.3.6.2 Sample Success Response #1 40](#_Toc468361684)

[6.3.6.3 Sample Success Response #2 40](#_Toc468361685)

[6.3.6.4 Sample Error Responses 41](#_Toc468361686)

[6.4 Operation: publishEventBatch 42](#_Toc468361687)

[6.4.1 Functional Behavior 42](#_Toc468361688)

[6.4.2 Call Flow 43](#_Toc468361689)

[6.4.3 Input Parameters 43](#_Toc468361690)

[6.4.4 Output Parameters 44](#_Toc468361691)

[6.4.5 HTTP Status Codes 45](#_Toc468361692)

[6.4.6 Sample Request and Response 45](#_Toc468361693)

[6.4.6.1 Sample Request 45](#_Toc468361694)

[6.4.6.2 Sample Success Response #1 46](#_Toc468361695)

[6.4.6.3 Sample Success Response #2 47](#_Toc468361696)

[6.4.6.4 Sample Error Responses 48](#_Toc468361697)

[6.5 Operation: provideThrottlingState 49](#_Toc468361698)

[6.5.1 Functional Behavior 49](#_Toc468361699)

[6.5.2 Call Flow 49](#_Toc468361700)

[6.5.3 Input Parameters 49](#_Toc468361701)

[6.5.4 Output Parameters 50](#_Toc468361702)

[6.5.5 HTTP Status Codes 50](#_Toc468361703)

[6.5.6 Sample Request and Response 51](#_Toc468361704)

[6.5.6.1 Sample Request 51](#_Toc468361705)

[6.5.6.2 Sample Success Response 52](#_Toc468361706)

[6.5.6.3 Sample Error Responses 52](#_Toc468361707)

Figures

[Figure 1 – REST Resource Structure 6](#_Toc468226663)

[Figure 2 - publishAnyEvent Call Flow 31](#_Toc468226664)

[Figure 3 - publishSpecificTopic Call Flow 37](#_Toc468226665)

[Figure 4 – publishEventBatch Call Flow 43](#_Toc468226666)

[Figure 5 - provideClientThrottlingState Call Flow 49](#_Toc468226667)

Tables

[Table 1 - Service Exceptions 27](#_Toc468226668)

[Table 2 - Policy Exceptions 28](#_Toc468226669)

[Table 3 - REST Operation Summary 29](#_Toc468226670)

# Introduction

This document describes the RESTful interface for AT&T’s Generic Event Listener. The Generic Event Listener is capable of receiving any event sent in AT&T’s Common Event Format. The Common Event Format is a JSON structure consisting of a required Common Event Header Block accompanied by zero or more event domain blocks. A JSON Schema of the Common Event Format is provided later in this document.

It should be understood that events are well structured packages of information, identified by an eventType, which are asynchronously communicated to subscribers who are interested in the eventType. Events can convey measurements, faults, CDRs, alerts, geolocation updates, and much more. Even something as unusual as SIP signaling messages could be captured by a system and published as a SIP Signaling Event to interested subscribers. Events are simply a way of communicating well-structured packages of information to one or more instances of an Event Listener service.

This document describes a RESTful connectionless push event listener that is capable of receiving single events or batches of events. In future, additional documents may describe other transports which make use of persistent TCP connections for high volumes of streaming events.

## Audience

General audience for API service specifications are:

* Security Architecture and Governance
* TD and D2 Architecture
* Testing Organizations
* Service Development Teams
* OSS/BSS Systems Teams
* AT&T Solution Providers
* AT&T VNF Providers

## Terminology

Topics, also known as Event Types, use the generic event format but may require that specific fields be present including specific name-value pairs in the extensible structures provided within the generic event structure.

Events are instances of topics.

## Naming Standards for Event Types

To prevent naming collisions, eventTypes sent as part of the Datatype: commonEventHeader, should follow the following convention:

{DomainAbbreviation}\_{ServiceOrResourceOrSystemName}\_{DescriptionOfInfoBeingConveyed}

Domain abbreviations are derived from the ‘domain’ field in the Datatype: commonEventHeader, as specified below:

* ‘Fault’ for the fault domain
* ‘Heartbeat’ for the heartbeat domain
* ‘Mfvs’ for the measurementsForVfScaling domain
* ‘MobileFlow’ for the mobileFlow domain
* ‘Other’ for the other domain
* SstateChange’ for the stateChange domain
* ‘Syslog’ for the syslog domain
* ‘Tca’ for the thresholdCrossingAlert domain

Examples of eventTypes following the naming standards are provided below:

* Fault\_MobileCallRecording\_PilotNumberPoolExhaustion
* Heartbeat\_vIsbcMmc
* Other\_Mso\_L3toHlsInstantiationStage1Complete
* Syslog\_vdbe
* Tca\_vdbe\_CpuThresholdExceeded

Any questions about eventType naming should be resolved as part of service and resource onboarding to ASDC (AT&T Service Design and Creation).

## Support for Protocols Other Than HTTPS

This API specification describes an HTTPS RESTful interface using the JSON content-type.

Alternative specifications may be provided in future using Websockets, which would establish a permanent TCP socket, or Apache Avro which provides a binary format over an RPC protocol to be defined. Both would leverage the JSON schema provided in this document.

## Versioning

Three types of version numbers supported by this specification:

* The API specification itself is versioned. Going forward, the major number of the specification version will be incremented whenever any change could break an existing client (e.g., a field name is deleted or changed). All other changes to the spec (e.g., a field name is added or text changes are made to the specification itself) will increment only the minor number. Note that the major number appears in REST resource URLs as v# (where ‘#’ is the major number).
* The JSON schema is versioned. Going forward, the major number of the JSON schema will be incremented whenever any change could break an existing client (e.g., a field name is deleted or changed). All other changes to the schema (e.g., a field name is added or text changes are made to the field descriptions) will increment only the minor number.
* The field blocks are versioned. Field blocks include the commonEventHeader and the domain blocks (e.g., the faultFields block). Going forward, the major number of each field block will be incremented whenever any change to that block could break an existing client (e.g., a field name is deleted or changed). All other changes to that block (e.g., a field name is added or text changes are made to the field descriptions) will increment only the minor number.

# Security

Event sources must identify themselves to the Generic Event Listener.

Event source credentials are passed using HTTP [Basic Authentication](http://tools.ietf.org/html/rfc2617).

Credentials must not be passed on the query string. Credentials must be sent in an Authorization header as follows:

1. The username and password are formed into one string as “username:password”
2. The resulting string is Base64 encoded to produce the encoded credential.
3. The encoded credential is communicated in the header after the string “Authorization: Basic “

Because the credentials are merely encoded but not encrypted, HTTPS (rather than HTTP) should be used. HTTPS will also encrypt and protect event contents.

Examples are provided below.

### Sample Request and Response

#### Sample Request

|  |
| --- |
| POST /eventListener/v3 HTTPS/1.1  Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==  content-type: application/json  content-length: 12345 {  "event": {  "commonEventHeader": {  “domain”: “heartbeat”,  "eventType": "Heartbeat\_vIsbcMmc",  "eventId": "ab305d54-85b4-a31b-7db2-fb6b9e546015",  "sequence": "0",  "priority": "Normal",  “reportingEntityId”: “cc305d54-75b4-431b-adb2-eb6b9e541234”,  “reportingEntityName”: “EricssonOamVf”,  "sourceId": "de305d54-75b4-431b-adb2-eb6b9e546014",  “sourceName”: “EricssonECE”,  “functionalRole”: “SCF”,  “startEpochMicrosec”: “1413378172000000”,  “lastEpochMicrosec”: “1413378172000000”  }  }  } |

#### Sample Success Response

|  |
| --- |
| HTTPS/1.1 202 Accepted |

# Resource Structure

REST resources are defined with respect to a ServerRoot:

ServerRoot = https://{ Domain}:{Port}/{optionalRoutingtPath}

The resource structure is provided below:

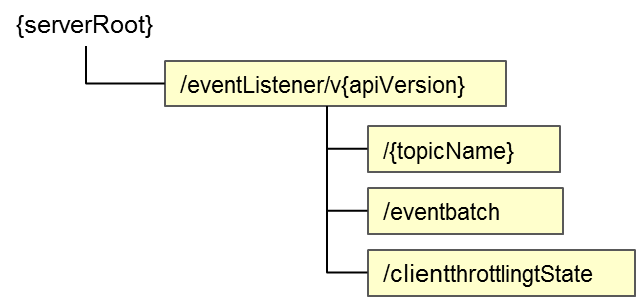


Figure 1 – REST Resource Structure

# Generic Event Format

A JSON schema describing the Generic Event Format is provided below and is reproduced in the tables that follow.



## Datatype: codecsInUse

The codecsInUse datatype consists of the following fields describing the number of times an identified codec was used over the measurementInterval:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| codecIdentifer | string | Yes | Description of the codec |
| numberInUse | number | Yes | Number of such codecs in use |

## Datatype: codecSelected

The codecSelected datatype describes the codec selected for the call; it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| codec | string | No | Codec selected for the call (e.g., PCMA, G729A…) |

## Datatype: codecSelectedTranscoding

The codecSelectedTranscoding datatype describes the codecs selected for the call, when transcoding is occurring; it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| calleeSideCodec | string | Yes | Callee codec |
| callerSideCodec | string | Yes | Caller codec |

## Datatype: command

The command datatype is used by an event collector to request changes in the behavior of an event source (for more information, see 6.1.3); it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| commandType | string | Yes | Enumeration: ‘throttllingSpecification’, ‘provideThrottlingState’, ‘measurementIntervalChange’ |
| eventDomainThrottle Specification | eventDomainThrottleSpecification | No | If commandType is ‘throttlingSpecification’, the fields to suppress within an event domain |
| measurementInterval | number | No | If commandType is ‘measurementIntervalChange’, the measurementInterval duration to use in seconds |

## Datatype: commandList

The commandList datatype is an array of commands from an event collector toward an event source; it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| commandListEntry | commandListEntry | Yes | List of commands from an event collector toward an event source |

## Datatype: commandListEntry

The commandListEntry datatype references a command object; it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| command | command | Yes | References a command object |

## Datatype: commonEventHeader

The commonEventHeader datatype consists of the following fields common to all events:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| version | number | No | Version of the event header (currently: 1.2) |
| eventType | string | No | Unique event topic name |
| domain | string | Yes | Event domain enumeration: ‘fault’, ‘heartbeat’, ‘measurementsForVfScaling’, ‘mobileFlow’, ‘other’, ‘serviceEvents’, ‘signaling’, ‘stateChange’, ‘syslog’, ‘thresholdCrossingAlert’ |
| eventId | string | Yes | Event key that is unique to the event source |
| sourceId | string | No | UUID identifying the entity experiencing the event issue (note: will eventually be supplied through event enrichment) |
| sourceName | string | Yes | Name of the entity experiencing the event issue |
| functionalRole | string | Yes | Function of the event source e.g., eNodeB, MME, PCRF |
| reportingEntityId | string | No | UUID identifying the entity reporting the event, for example an OAM VM |
| reportingEntityName | string | Yes | Name of the entity reporting the event, for example, an OAM VM (note: will eventually be supplied through event enrichment) |
| priority | string | Yes | Processing priority enumeration: ‘High’, ‘Medium’, ‘Normal’, ‘Low’ |
| startEpochMicrosec | number | Yes | the earliest unix time aka epoch time associated with the event from any component--as microseconds elapsed since 1 Jan 1970 not including leap seconds |
| lastEpochMicrosec | number | Yes | the latest unix time aka epoch time associated with the event from any component--as microseconds elapsed since 1 Jan 1970 not including leap seconds |
| sequence | integer | Yes | Ordering of events communicated by an event source instance (or 0 if not needed) |

## Datatype: counter

The counter datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| name | string | Yes | Name of the counter |
| value | string | Yes | Current value of the counter |
| threshholdCrossed | string | Yes | Last threshold that was crossed |
| criticality | string | Yes | Enumeration: ‘CRIT’, ‘MAJ’ |

## Datatype: cpuUsage

The cpuUsage datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| cpuIdentifer | string | Yes | CPU Identifier |
| percentUsage | number | Yes | CPU usage in percent |

## Datatype: endOfCallVqmSummaries

The endOfCallVqmSummaries datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| adjacencyName | string | No | Adjacency name |
| endpointDescription | string | No | Enumeration: ‘Caller’, ‘Callee’ |
| endpointJitter | number | No | Endpoint jitter |
| endpointRtpOctetsDiscarded | number | No | Endpoint RTP octets discarded |
| endpointRtpOctetsReceived | number | No | Endpoint RTP octets received |
| endpointRtpOctetsSent | number | No | Endpoint RTP octets sent |
| endpointRtpPacketsDiscarded | number | No | Endpoint RTP packets discarded |
| endpointRtpPacketsReceived | number | No | Endpoint RTP packets received |
| endpointRtpPacketsSent | number | No | Endpoint RTP packets sent |
| localJitter | number | No | Local jitter |
| localRtpOctetsDiscarded | number | No | Local RTP octets discarded |
| localRtpOctetsReceived | number | No | Local RTP octets received |
| localRtpOctetsSent | number | No | Local RTP octets sent |
| localRtpPacketsDiscarded | number | No | Local RTP packets discarded |
| localRtpPacketsReceived | number | No | Local RTP packets received |
| localRtpPacketsSent | number | No | Local RTP packets sent |
| mosCqe | number | No | Decimal range from 1 to 5 (1 decimal place) |
| packetsLost | number | No | Packets lost |
| packetLossPercent | number | No | Calculated percentage packet loss based on endpoint RTP packets lost (as reported in RTCP) and local RTP packets sent. Direction is based on endpoint description (Caller, Callee). Decimal (2 decimal places) |
| rFactor | number | No | rFactor from 0 to 100 |
| roundTripDelay | number | No | Round trip delay in milliseconds |

## Datatype: errors

The errors datatype provides receive and transmit errors for the measurements domain; it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| receiveDiscards | number | Yes | Receive discards |
| receiveErrors | number | Yes | Receive errors |
| transmitDiscards | number | Yes | Transmit discards |
| transmitErrors | number | Yes | Transmit errors |

## Datatype: event

The event datatype consists of the following fields which constitue the generic or common event format:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| commonEventHeader | commonEventHeader | Yes | Fields common to all events |
| faultFields | faultFields | No | Fields specific to fault events |
| measurementsForVfScalingFields | measurementsForVfScalingFields | No | Fields specific to measurementsForVfScaling events |
| mobileFlowFields | mobileFlowFields | No | Fields specific to mobility flow events |
| otherFields | field [ ] | No | Fields specific to other types of events |
| serviceEventsFields | serviceEventsFields | No | Fields specific to service events |
| signalingFields | signalingFields | No | Fields specific to signaling events |
| stateChangeFields | stateChangeFields | No | Fields specific to state change events |
| syslogFields | syslogFields | No | Fields specific to syslog events |
| threholdCrossingAlertFields | thresholdCrossingAlertFields | No | Fields specific to threshold crossing alert events |

## Datatype: eventDomainThrottleSpecification

The eventDomainThrottleSpecification datatype specifies what fields to suppress within an event domain; it consists of the following fields common to all events:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| eventDomain | string | Yes | Event domain enum from the commonEventHeader domain field |
| suppressedFieldNames | string [ ] | No | List of optional field names in the event block that should not be sent to the Event Listener |
| suppressedNvPairsList | suppressedNvPairs [ ] | No | Optional list of specific NvPairsNames to suppress within a given Name-Value Field |

## Datatype: eventDomainThrottleSpecificationList

The eventDomainThrottleSpecificationList datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| eventDomainThrottleSpecificationList | eventDomainThrottleSpecification [ ] | Yes | Array of eventDomainThrottleSpecifications |

## Datatype: eventInstanceIdentifier

The eventInstanceIdentifier datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| eventFriendlyName | string | No | Event instance friendly name |
| eventId | string | Yes | Event Identifier |
| productId | string | No | Product identifier |
| subsystemId | string | No | Subsystem identifier |
| vendorId | string | Yes | Vendor identifier |

## Datatype: eventList

The eventList datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| eventList | event [ ] | Yes | Array of events |

## Datatype: eventThrottlingState

The eventThrottlingState datatype reports the throttling in force at the event source; it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| eventThrottlingMode | string | Yes | Enumeration: ‘normal’, ‘throttled’ |
| eventDomainThrottleSpecificationList | eventDomainThrottleSpecificationList | No | A list of eventDomainThrottleSpecifications currently in force at the event source, if the eventManagerMode is ‘throttled’ |

## Datatype: faultFields

The faultFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| faultFieldsVersion | number | No | Version of the faultFields block (currently: 1.1) |
| eventSeverity | string | Yes | Event severity or priority enumeration: ‘CRITICAL’, ‘MAJOR’, ‘MINOR’, ‘WARNING’, ‘NORMAL’ |
| eventSourceType | string | Yes | Examples: ‘other’, ‘router’, ‘switch’, ‘host’, ‘card’, ‘port’, ‘slotThreshold’, ‘portThreshold’, ‘virtualMachine’, ‘virtualNetworkFunction’ |
| alarmCondition | string | Yes | Alarm condition reported by the device |
| specificProblem | string | Yes | Short description of the alarm or problem |
| vfStatus | string | Yes | Virtual function status enumeration: ‘Active’, ‘Idle’, ‘Preparing to terminate’, ‘Ready to terminate’, ‘Requesting Termination’ |
| alarmtInterfaceA | string | No | Card, port, channel or interface name of the device generating the alarm |
| alarmAdditional Information | field [ ] | No | Additional alarm information |

## Datatype: featuresInUse

The featuresInUse datatype consists of the following fields which describe the number of times an identified feature was used over the measurementInterval:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| featureIdentifer | string | Yes | Description of the feature |
| feautureUtilization | Number | Yes | Number of times the identified feature was used |

## Datatype: field

The field datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| Name | string | Yes | Name of the field |
| Value | string | Yes | Value of the named field |

## Datatype: filesystemUsage

The filesystemUsage datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| filesystemName | string | Yes | File system name |
| blockConfigured | number | Yes | Configured block storage capacity in GB |
| blockIops | number | Yes | Block storage input-output operations per second |
| blockUsed | number | Yes | Used block storage capacity in GB |
| ephemeralConfigured | number | Yes | Configured ephemeral storage capacity in GB |
| ephemeralIops | number | Yes | Ephemeral storage input-output operations per second |
| ephemeralUsed | number | Yes | Used ephemeral storage capacity in GB |

## Datatype: gtpPerFlowMetrics

The gtpPerFlowMetrics datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| avgBitErrorRate | number | Yes | Average bit error rate |
| avgPacketDelayVariation | number | Yes | Average packet delay variation or jitter in milliseconds for received packets: Average difference between the packet timestamp and time received for all pairs of consecutive packets |
| avgPacketLatency | number | Yes | Average delivery latency |
| avgReceiveThroughput | number | Yes | Average receive throughput |
| avgTransmitThroughput | number | Yes | Average transmit throughput |
| durConnectionFailedStatus | number | No | Duration of failed state in milliseconds, computed as the cumulative time between a failed echo request and the next following successful error request, over this reporting interval |
| durTunnelFailedStatus | number | No | Duration of errored state, computed as the cumulative time between a tunnel error indicator and the next following non-errored indicator, over this reporting interval |
| flowActivatedBy | string | No | Endpoint activating the flow |
| flowActivationEpoch | number | Yes | Time the connection is activated in the flow (connection) being reported on, or transmission time of the first packet if activation time is not available |
| flowActivationMicrosec | number | Yes | Integer microseconds for the start of the flow connection |
| flowActivationTime | Dateto,e | No | Time the connection is activated in the flow being reported on, or transmission time of the first packet if activation time is not available; with RFC 2822 compliant format: ‘Sat, 13 Mar 2010 11:29:05 -0800’ |
| flowDeactivatedBy | string | No | Endpoint deactivating the flow |
| flowDeactivationEpoch | number | Yes | Time for the start of the flow connection, in integer UTC epoch time aka UNIX time |
| flowDeactivationMicrosec | number | Yes | Integer microseconds for the start of the flow connection |
| flowDeactivationTime | datetime | Yes | Transmission time of the first packet in the flow connection being reported on; with RFC 2822 compliant format: ‘Sat, 13 Mar 2010 11:29:05 -0800’ |
| flowStatus | string | Yes | Connection status at reporting time as a working / inactive / failed indicator value |
| gtpConnectionStatus | string | No | Current connection state at reporting time |
| gtpTunnelStatus | string | No | Current tunnel state at reporting time |
| ipTosCountList | associative array | No | Array of key: value pairs where the keys are drawn from the IP Type-of-Service identifiers which range from '0' to '255', and the values are the count of packets that had those ToS identifiers in the flow |
| ipTosList | string | No | Array of unique IP Type-of-Service values observed in the flow where values range from '0' to '255' |
| largePacketRtt | number | No | large packet round trip time |
| largePacketThreshold | number | No | large packet threshold being applied |
| maxPacketDelayVariation | number | Yes | Maximum packet delay variation or jitter in milliseconds for received packets: Maximum of the difference between the packet timestamp and time received for all pairs of consecutive packets |
| maxReceiveBitRate | number | No | maximum receive bit rate" |
| maxTransmitBitRate | number | No | maximum transmit bit rate |
| mobileQciCosCountList | associative array | No | array of key: value pairs where the keys are drawn from LTE QCI or UMTS class of service strings, and the values are the count of packets that had those strings in the flow |
| mobileQciCosList | string | No | Array of unique LTE QCI or UMTS class-of-service values observed in the flow |
| numActivationFailures | number | Yes | Number of failed activation requests, as observed by the reporting node |
| numBitErrors | number | Yes | number of errored bits |
| numBytesReceived | number | Yes | number of bytes received, including retransmissions |
| numBytesTransmitted | number | Yes | number of bytes transmitted, including retransmissions |
| numDroppedPackets | number | Yes | number of received packets dropped due to errors per virtual interface |
| numGtpEchoFailures | number | No | Number of Echo request path failures where failed paths are defined in 3GPP TS 29.281 sec 7.2.1 and 3GPP TS 29.060 sec. 11.2 |
| numGtpTunnelErrors | number | No | Number of tunnel error indications where errors are defined in 3GPP TS 29.281 sec 7.3.1 and 3GPP TS 29.060 sec. 11.1 |
| numHttpErrors | number | No | Http error count |
| numL7BytesReceived | number | Yes | number of tunneled layer 7 bytes received, including retransmissions |
| numL7BytesTransmitted | number | Yes | number of tunneled layer 7 bytes transmitted, excluding retransmissions |
| numLostPackets | number | Yes | number of lost packets |
| numOutOfOrderPackets | number | Yes | number of out-of-order packets |
| numPacketErrors | number | Yes | number of errored packets |
| numPacketsReceivedExclRetrans | number | Yes | number of packets received, excluding retransmission |
| numPacketsReceivedInclRetrans | number | Yes | number of packets received, including retransmission |
| numPacketsTransmittedInclRetrans | number | Yes | number of packets transmitted, including retransmissions |
| numRetries | number | Yes | number of packet retrie |
| numTimeouts | number | Yes | number of packet timeouts |
| numTunneledL7BytesReceived | number | Yes | number of tunneled layer 7 bytes received, excluding retransmissions |
| roundTripTime | number | Yes | Round Trip time |
| tcpFlagCountList | associative array | No | Array of key: value pairs where the keys are drawn from TCP Flags and the values are the count of packets that had that TCP Flag in the flow |
| tcpFlagList | string | No | Array of unique TCP Flags observed in the flow |
| timeToFirstByte | number | Yes | Time in milliseconds between the connection activation and first byte received |

## Datatype: latencyBucketMeasure

The latencyBucketMeasure datatype consists of the following fields which describe the number of counts falling within a defined latency bucket:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| countsInTheBucket | number | Yes | Number of counts falling within a defined latency bucket |
| highEndOfLatencyBucket | number | No | High end of bucket range (typically in ms) |
| lowEndOfLatencyBucket | number | No | Low end of bucket range (typically in ms) |

## Datatype: marker

The marker datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| phoneNumber | string | No | Phone number |

## Datatype: measurementForVfScalingFields

The measurementForVfScalingFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| measurementsForVfScalingFieldsVersion | number | No | Version of the measurementsForVfScalingFields block (currently: 1.1) |
| additionalMeasurements | measurementGroup [ ] | No | Additional measurement fields |
| aggregateCpuUsage | number | No | Aggregate CPU usage of the VM on which the VNFC reporting the event is running |
| codecUsageArray | codecsInUse [] | No | Array of codecs in use |
| concurrentSessions | number | No | Peak concurrent sessions for the VM or VNF (depending on the context) over the measurementInterval |
| configuredEntities | number | No | Depending on the context over the measurementInterval: peak total number of users, subscribers, devices, adjacencies, etc., for the VM, or peak total number of subscribers, devices, etc., for the VNF |
| cpuUsageArray | cpuUsage [] | No | Usage of an array of CPUs |
| errors | errors | No | Receive and transmit errors for the measurements domain |
| featureUsageArray | featuresInUse [] | No | Array of features in use |
| filesystemUsageArray | filesystemUsage [] | No | Filesystem usage of the VM on which the VNFC reporting the event is running |
| latencyDistribution | latencyBucketMeasure [ ] | No | Array of integers representing counts of requests whose latency in milliseconds falls within per-VNF configured ranges; where latency is the duration between a service request and its fulfillment. |
| meanRequestLatency | number | No | Mean seconds required to respond to each request for the VM on which the VNFC reporting the event is running |
| measurementInterval | number | Yes | Interval over which the measurements are being reported in seconds |
| memoryConfigured | number | No | Memory configured in the VM on which the VNFC reporting the event is running |
| memoryUsed | number | No | Memory usage of the VM on which the VNFC reporting the event is running |
| numberOfMediaPortsInUse | Number | No | Number of media ports in use |
| requestRate | number | No | Peak rate of service requests per second to the VNF over the measurementInterval |
| vnfcScalingMetric | number | No | Represents busy-ness of the VNF from 0 to 100 as reported by the VNFC |
| vNicUsageArray | vNicUsage [ ] | No | Usage of an array of virtual network interface cards |

## Datatype: measurementGroup

The measurementGroup datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| name | string | Yes | Name for the measurement Group |
| measurements | field [ ] | Yes | Name value pair measurements |

## Datatype: mobileFlowFields

The mobileFlowFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| mobileFlowFieldsVersion | number | No | Version of the mobileFlowFields block (currently: 1.1) |
| applicationType | string | No | Application type inferred |
| appProtocolType | string | No | Application protocol |
| appProtocolVersion | string | No | Application version |
| cid | string | No | Cell Id |
| connectionType | string | No | Abbreviation referencing a 3GPP reference point e.g., S1-U, S11, etc |
| ecgi | string | No | Evolved Cell Global Id |
| flowDirection | string | Yes | Flow direction, indicating if the reporting node is the source of the flow or destination for the flow |
| gtpPerFlowMetrics | gtpPer FlowMetrics | Yes | Mobility GTP Protocol per flow metrics |
| gtpProtocolType | string | No | GTP protocol |
| gtpVersion | string | No | GTP protocol version |
| httpHeader | string | No | HTTP request header, if the flow connects to a node referenced by HTTP |
| imei | string | No | IMEI for the subscriber UE used in this flow, if the flow connects to a mobile device |
| imsi | string | No | IMSI for the subscriber UE used in this flow, if the flow connects to a mobile device |
| ipProtocolType | string | Yes | IP protocol type e.g., TCP, UDP, RTP... |
| ipVersion | string | Yes | IP protocol version e.g., IPv4, IPv6 |
| lac | string | No | Location area code |
| mcc | string | No | Mobile country code |
| mnc | string | No | Mobile network code |
| msisdn | string | No | MSISDN for the subscriber UE used in this flow, as an integer, if the flow connects to a mobile device |
| otherEndpointIpAddress | string | Yes | IP address for the other endpoint, as used for the flow being reported on |
| otherEndpointPort | string | Yes | IP Port for the reporting entity, as used for the flow being reported on |
| otherFunctionalRole | string | No | Functional role of the other endpoint for the flow being reported on e.g., MME, S-GW, P-GW, PCRF... |
| rac | string | No | Routing area code |
| radioAccessTechnology | string | No | Radio Access Technology e.g., 2G, 3G, LTE |
| reportingEndpointIpAddr | string | Yes | IP address for the reporting entity, as used for the flow being reported on |
| reportingEndpointPort | string | Yes | IP port for the reporting entity, as used for the flow being reported on |
| sac | string | No | Service area code |
| samplingAlgorithm | string | No | Integer identifier for the sampling algorithm or rule being applied in calculating the flow metrics if metrics are calculated based on a sample of packets, or 0 if no sampling is applied |
| tac | string | No | Transport area code |
| tunnelId | string | No | Tunnel identifier |
| vlanId | string | No | VLAN identifier used by this flow |

## Datatype: otherFields

The otherFields datatype is simply a field [ ]

## Datatype: requestError

The requestError datatype defines the standard request error data structure:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| messageId | string | Yes | Unique message identifier of the format ‘ABCnnnn’ where ‘ABC’ is either ‘SVC’ for Service Exceptions or ‘POL’ for Policy Exception. Exception numbers may be in the range of 0001 to 9999 where 0001 to 2999 are defined by OMA (see section 5.1) and 3000-9999 are available and undefined. |
| text | string | Yes | Message text, with replacement variables marked with %n, where n is an index into the list of <variables> elements, starting at 1 |
| url | string | No | Hyperlink to a detailed error resource e.g., an HTML page for browser user agents |
| variables | string | No | List of zero or more strings that represent the contents of the variables used by the message text |

## Datatype: serviceEventsFields

The serviceEventsFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| serviceEventsFieldsVersion | number | No | Version of the serviceEventsFields block (currently: 1.0) |
| additionalFields | field [ ] | No | Additional service events fields |
| codecSelected | codecSelected | No | Codec selected |
| codecSelectedTranscoding | codecSelectedTranscoding | No | Codec selected transcoding |
| correlator | string | No | This is the same for all events on this call |
| endOfCallVqmSummaries | endOfCallVqmSummaries | No | End of call Vqm summaries |
| eventInstanceIdentifier | eventInstanceIdentifier | Yes | Event instance identifier |
| marker | marker | No | marker |
| midCallRtcp | field [ ] | No | Additional service events fields |

## Datatype: signalingFields

The signalingFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| signalingFieldsVersion | number | No | Version of the signalingFields block (currently: 1.0) |
| compressedSip | string | No | The full SIP request/response including headers and bodies |
| correlator | string | No | This is the same for all events on this call |
| eventInstanceIdentifier | eventInstanceIdentifier | Yes | Event instance identifier |
| localIpAddress | string | No | Ip address on VNF |
| localPort | string | No | Port on VNF |
| remoteIpAddress | string | No | IP address of peer endpoint |
| remotePort | string | No | Port of peer endpoint |
| summarySip | string | No | The SIP Method or Response (‘INVITE’, ‘200 OK’, ‘BYE’, etc) |

## Datatype: stateChangeFields

The stateChangeFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| stateChangeFieldsVersion | number | No | Version of the stateChangeFields block (currently: 1.1) |
| additionalFields | field [ ] | No | Additional stateChange fields if needed |
| newState | string | Yes | New state of the entity: ‘inService’, ‘maintenance’, ‘outOfService’ |
| oldState | string | Yes | Previous state of the entity: ‘inService’, ‘maintenance’, ‘outOfService’ |
| stateInterface | string | Yes | Card or port name of the entity that changed state |

## Datatype: suppressedNvPairs

The suppressedNvPairs datatype is a list of specific NvPairsNames to suppress within a given Name-Value Field (for event throttling); it consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| nvPairFieldName | string | Yes | Name of the field within which are the nvpair names to suppress |
| suppressedNvPairNames | string [ ] | Yes | Array of nvpair names to suppress (within the nvpairFieldName) |

## Datatype: syslogFields

The syslogFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| syslogFieldsVersion | number | No | Version of the syslogFields block (currently: 1.2) |
| additionalFields | field [ ] | No | Additional syslog fields if needed |
| eventSourceHost | string | No | Hostname of the device |
| eventSourceType | string | Yes | Examples: ‘other’, ‘router’, ‘switch’, ‘host’, ‘card’, ‘port’, ‘slotThreshold’, ‘portThreshold’, ‘virtualMachine’, ‘virtualNetworkFunction’ |
| syslogFacility | number | No | Numeric code from 0 to 23 for facility:  0 kernel messages  1 user-level messages  2 mail system  3 system daemons  4 security/authorization messages  5 messages generated internally by syslogd  6 line printer subsystem  7 network news subsystem  8 UUCP subsystem  9 clock daemon  10 security/authorization messages  11 FTP daemon  12 NTP subsystem  13 log audit  14 log alert  15 clock daemon (note 2)  16 local use 0 (local0)  17 local use 1 (local1)  18 local use 2 (local2)  19 local use 3 (local3)  20 local use 4 (local4)  21 local use 5 (local5)  22 local use 6 (local6)  23 local use 7 (local7 ) |
| syslogMsg | string | Yes | Syslog message |
| syslogPri | number | No | 0-192  Combined Severity and Facility |
| syslogProc | string | No | Identifies the application that originated the message |
| syslogProcId | number | No | A change in the value of this field indicates a discontinuity in syslog reporting |
| syslogSData | string | No | Syslog structured data consisting of a structured data Id followed by a set of key value pairs (see below for an example)  \*\*Note: SD-ID may not be present if syslogSdId is populated |
| syslogSdId | string | No | 0-32 char in format name@number,  ie ourSDID@32473 |
| syslogSev | String | No | Numerical Code for Severity  (derived from syslogPri: remaider of syslogPri / 8)  0 Emergency: system is unusable  1 Alert: action must be taken immediately  2 Critical: critical conditions  3 Error: error conditions  4 Warning: warning conditions  5 Notice: normal but significant condition  6 Informational: informational messages  7 Debug: debug-level messages |
| syslogTag | string | Yes | MsgId indicating the type of message such as ‘TCPOUT’ or ‘TCPIN’; ‘NILVALUE’ should be used when no other value can be provided |
| syslogVer | number | No | IANA assigned version of the syslog protocol specification (typically ‘1’) |

Example of syslogSData:

STRUCTURED-DATA = NILVALUE / 1\*SD-ELEMENT

SD-ELEMENT = "[" SD-ID \*(SP SD-PARAM) "]"

SD-PARAM = PARAM-NAME "=" %d34 PARAM-VALUE %d34

SD-ID = SD-NAME

PARAM-NAME = SD-NAME

PARAM-VALUE = UTF-8-STRING ; characters '"', '\' and

; ']' MUST be escaped.

SD-NAME = 1\*32PRINTUSASCII

; except '=', SP, ']', %d34 (")

## Datatype: thresholdCrossingAlertFields

The thresholdCrossingAlertFields datatype consists of the following fields:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| thresholdCrossing FieldsVersion | number | No | Version of the thresholdCrossingAlertFields block (currently: 1.2) |
| additionalFields | field [ ] | No | Additional threshold crossing alert fields if needed |
| additionalParameters | counter [ ] | Yes | Array of performance counters |
| alertAction | string | Yes | Enumeration: ‘SET’, ‘CONT’, ‘CLEAR’ |
| alertDescription | string | Yes | Unique short alert description (e.g., NE-CPUMEM) |
| alertType | string | Yes | Enumeration: ‘CARD-ANOMALY’, ‘INTERFACE-ANOMALY’, ELEMENT-ANOMALY’, ‘SERVICE-ANOMALY’ |
| alertValue | string | No | Calculated API value (if applicable) |
| associatedAlertIdList | string [ ] | No | List of eventIds associated with the event being reported |
| collectionTimestamp | string | Yes | Time when the performance collector picked up the data; with RFC 2822 compliant format: ‘Sat, 13 Mar 2010 11:29:05 -0800’ |
| dataCollector | string | No | Specific performance collector instance used |
| elementType | string | No | Type of network element (internal AT&T field) |
| eventSeverity | string | Yes | Event severity or priority enumeration: ‘CRITICAL’, ‘MAJOR’, ‘MINOR’, ‘WARNING’, ‘NORMAL’ |
| eventStartTimestamp | string | Yes | Time closest to when the measurement was made; with RFC 2822 compliant format: ‘Sat, 13 Mar 2010 11:29:05 -0800’ |
| interfaceName | string | No | Physical or logical port or card (if applicable) |
| networkService | string | No | Network name (internal AT&T field) |
| possibleRootCause | string | No | Reserved for future use |

## Datatype: vNicUsage

The vNicUsage datatype consists of the following fields which describe the usage of an of an identified virtual network interface card:

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| broadcastPacketsIn | number | No | Number of broadcast packets received |
| broadcastPacketsOut | number | No | Number of broadcast packets sent |
| bytesIn | number | Yes | Per identified vNic in megabytes |
| bytesOut | number | Yes | Per identified vNic in megabytes |
| multicastPacketsIn | number | No | Number of multicast packets received |
| multicastPacketsOut | number | No | Number of multicast packets sent |
| packetsIn | number | Yes | Total number of packets received |
| packetsOut | number | Yes | Total number of packets sent |
| unicastPacketsIn | number | No | Number of unicast packets received |
| unicastPacketsOut | number | No | Number of unicast packets sent |
| vNicIdentifier | string | Yes | vNic identification |

# Exceptions

## RESTful Web Services Exceptions

RESTful services generate and send exceptions to clients in response to invocation errors. Exceptions send HTTP status codes (specified later in this document for each operation). HTTP status codes may be followed by an optional JSON exception structure described below. Two types of exceptions may be defined: service exceptions and policy exceptions.

| **Field Name** | **Data Type** | **Required?** | **Description** |
| --- | --- | --- | --- |
| messageId | xs:string | Yes | Unique message identifier of the format ‘ABCnnnn’ where ‘ABC’ is either ‘SVC’ for Service Exceptions or ‘POL’ for Policy Exception.  Exception numbers may be in the range of 0001 to 9999 where :   * 0001 to 2999 are defined by OMA (see OMA’s [Common definitions for RESTful Network APIs](http://technical.openmobilealliance.org/Technical/release_program/docs/REST_NetAPI_Common/V1_0-20120417-C/OMA-TS-REST_NetAPI_Common-V1_0-20120417-C.pdf) for details) * 3000-9999 are available and undefined |
| text | xs:string | Yes | Message text, with replacement variables marked with %n, where n is an index into the list of <variables> elements, starting at 1 |
| variables | xs:string [0..unbounded] | No | List of zero or more strings that represent the contents of the variables used by the message text. |
| url | xs:anyUrl | No | Hyperlink to a detailed error resource (e.g., an HTML page for browser user agents). |

## Service Exceptions

When a service is not able to process a request, and retrying the request with the same information will also result in a failure, and the issue is not related to a service policy issue, then the service will issue a fault using the service exception fault message. Examples of service exceptions include invalid input, lack of availability of a required resource or a processing error.

A service exception uses the letters 'SVC' at the beginning of the message identifier. ‘SVC’ service exceptions used by the Generic Event Listener API are defined below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *MessageId* | *Description / Comment* | *Text* | *Variables* | *Parent HTTP Code* |
| SVC0001 | General service error (see SVC2000) | <custom error message> | None | 400 |
| SVC0002 | Bad parameter | Invalid input value for message part %1 | %1: message part | 400 |
| SVC1000 | No server resources | No server resources available to process the request | None | 500 |
| SVC2000 | More elaborate version of SVC0001 | The following service error occurred: %1. Error code is %2. | %1: human readable description of the error  %2: error code | 400 |

Table 1 - Service Exceptions

## Policy Exceptions

When a service is not able to complete because the request fails to meet a policy criteria, then the service will issue a fault using the policy exception fault message. To clarify how a policy exception differs from a service exception, consider that all the input to an operation may be valid as meeting the required input for the operation (thus no service exception), but using that input in the execution of the service may result in conditions that require the service not to complete. Examples of policy exceptions include privacy violations, requests not permitted under a governing service agreement or input content not acceptable to the service provider.

A Policy Exception uses the letters 'POL' at the beginning of the message identifier. ‘POL’ policy exceptions used by the Generic Event Listener API are defined below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *MessageId* | *Description / Comment* | *Text* | *Variables* | *Parent HTTP Code* |
| POL0001 | General policy error (see POL2000) | A policy error occurred. | None | 401 |
| POL1009 | User not provisioned for service | User has not been provisioned for service | None | 401 |
| POL1010 | User suspended from service | User has been suspended from service | None | 401 |
| POL2000 | More elaborate version of POL0001 | The following policy error occurred: %1. Error code is %2. | %1: human readable description of the error  %2: error code | 401 |
| POL9003 | Message size exceeds limit | Message content size exceeds the allowable limit | None | 400 |

Table 2 - Policy Exceptions

# RESTful Web Services Definition

## REST Operation Overview

### REST Operation Summary

| **Operation Action** | **HTTP**  **Verb** | **Resource URL relative to {ServerRoot}** |
| --- | --- | --- |
| publishAnyEvent | POST | /eventListener/v{apiVersion} |
| publishSpecificTopic | POST | /eventListener/v{apiVersion/{topicName} |
| publishEventBatch | POST | /eventListener/v{apiVersion}/eventBatch |
| provideClientThrottlingState | POST | /eventListener/v{apiVersion}/clientThrottlingState |

Table 3 - REST Operation Summary

### Api Version

apiVersion is used to describe the major version number of the event listener API (which is the same as the major version number of this specification). When this number changes, the implication is: clients of older versions will break in some way, if they try to use the new API without modification (e.g., unmodified v1 clients would not be able to use v2 without error).

### Commands Toward Event Source Clients

This specification supports commands from event consumers back toward event source clients. This enables the event consumer (e.g., AT&T event collectors) to command event sources to change their measurement intervals or throttle the information they are sending to the event consumer. Note that commands are sent as part of the synchronous response to events sent by the event source toward the event consumer. This is done so that the event source does not need to host a service to listen for commands from events consumers. The following commands are currently supported:

| **Command** | **Description** |
| --- | --- |
| measurementIntervalChange | Commands the event source to change its measurementInterval to the number provided (in seconds). If ‘0’ is provided, the event source should return to its default measurementInterval. |
| provideThrottlingState | Commands the event source to invoke the provideThrottlingState operation on the event consumer. |
| throttlingSpecification | Commands the event source to throttle events as specified by the provided eventDomainThrottlingSpecification. This specification identifies the fields to suppress within the domain and even supports identification of subfields to suppress within objects or name-value pair structures. Note that required fields should not be suppressed and may result in errors being thrown by the event consumer back toward the event source when events without the required fields are sent to the event consumer. Other notes for event sources:   * the default throttling state is \*off\* for all domains * the throttling state for a domain is altered only by receipt of an eventDomainThrottleSpecification for that domain * the presence of the optional suppressedFieldNames replaces any existing list of suppressed field names * if suppressedFieldNames is not provided, then any existing list of suppressed field names shall be discarded * the presence of the optional suppressedNvPairsList replaces the any existing list of suppressed name-value pairs * if suppressedNvPairsList is not provided, then any existing list of suppressed name-value pairs shall be discarded |

## Operation: publishAnyEvent

### Functional Behavior

Allows authorized clients to publish any single event to the generic event listener.

* Supports only secure HTTPS (one way SSL) access.
* Uses the HTTP verb POST
* Supports JSON content types
* Provides HTTP response codes as well as Service and Policy error messages
* Allows the event collector to use the HTTP response to command the event source to throttle event messages it may send in the future.

### Call Flow

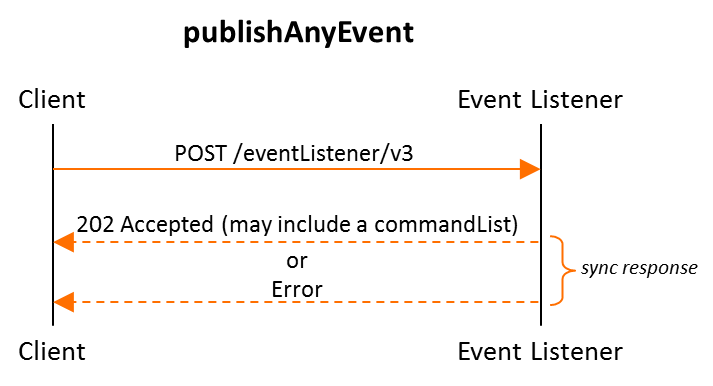


Figure 2 - publishAnyEvent Call Flow

### Input Parameters

Header Fields (note: all parameter names shall be treated as case-insensitive):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Accept | string | No | Determines the format of the body of the response. Valid values are:   * application/json |
| Authorization | string | Yes | The username and password are formed into one string as “username:password”. This string is then Base64 encoded to produce the encoded credential which is communicated in the header after the string “Authorization: Basic “. See examples below. If the Authorization header is missing, then an HTTP 400 Invalid Request message shall be returned. If the string supplied is invalid, then an HTTP 401 Unauthorized message shall be returned. |
| Content-length | integer | No | Note that content length is limited to 1Megabyte. |
| Content-type | string | Yes | Must be set to one of the following values:   * application/json |

Body Fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Event | event | Yes | Contains the JSON structure of the common event format. |

### Output Parameters

Header fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Content-length | integer | No | Used only in error conditions. |
| Content-type | string | No | Used only in error conditions |
| Date | datetime | Yes | Date time of the response in GMT |

Body Fields (for success responses without a commandList): no content is provided and the header fields are not required.

Body Fields (for success responses with one or more commands from the event collector toward the event source):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| commandList | commandList | No | Array of commands (e.g., measurement Interval changes and/or what fields to suppress within specified event domains and/or a request to report the state of event throttling by event domain that is currently in force in the event source). Note: for ‘provideThrottlingState’ commands, the client should subsequently provide the throttling state by calling the provideThrottlingState operation. |

Body Fields (for error Responses):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| requestError | requestError | Yes (for errors) | Used only in error conditions. |

### HTTP Status Codes

|  |  |  |
| --- | --- | --- |
| *Code* | *Reason Phrase* | *Description* |
| 202 | Accepted | The request has been accepted for processing |
| 400 | Bad Request | Many possible reasons not specified by the other codes (e.g., missing required parameters or incorrect format). The response body may include a further exception code and text. HTTP 400 errors may be mapped to SVC0001 (general service error), SVC0002 (bad parameter), SVC2000 (general service error with details) or PO9003 (message content size exceeds the allowable limit). |
| 401 | Unauthorized | Authentication failed or was not provided. HTTP 401 errors may be mapped to POL0001 (general policy error) or POL2000 (general policy error with details). |
| 404 | Not Found | The server has not found anything matching the Request-URI. No indication is given of whether the condition is temporary or permanent. |
| 405 | Method Not Allowed | A request was made of a resource using a request method not supported by that resource (e.g., using PUT on a REST resource that only supports POST). |
| 500 | Internal Server Error | The server encountered an internal error or timed out; please retry (general catch-all server-side error).HTTP 500 errors may be mapped to SVC1000 (no server resources). |

### Sample Request and Response

#### Sample Request

|  |
| --- |
| POST /eventListener/v3 HTTPS/1.1  Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==  content-type: application/json  content-length: 12345 {  "event": {  "commonEventHeader": {  “domain”: “fault”,  "eventType": "Fault\_MobileCallRecording\_PilotNumberPoolExhaustion",  "eventId": "ab305d54-85b4-a31b-7db2-fb6b9e546015",  "sequence": "0",  "priority": "High",  “reportingEntityId”: “cc305d54-75b4-431b-adb2-eb6b9e541234”,  “reportingEntityName”: “EricssonOamVf”,  "sourceId": "de305d54-75b4-431b-adb2-eb6b9e546014",  “sourceName”: “EricssonECE”,  “functionalRole”: “SCF”,  “startEpochMicrosec”: “1413378172000000”,  “lastEpochMicrosec”: “1413378172000000”  },  "faultFields": {  "alarmCondition": "PilotNumberPoolExhaustion",  "eventSourceType": "other",  "specificProblem": "Calls cannot complete because pilot numbers are unavailable"  "eventSeverity": "CRITICAL",  “vfStatus”: “Active”  }  }  } |

#### Sample Success Response #1

For success responses without a provided commandList:

|  |
| --- |
| HTTPS/1.1 202 Accepted |

#### Sample Success Response #2

For success responses with a provided commandList:

|  |
| --- |
| HTTPS/1.1 202 Accepted  content-type: application/json  content-length: nnn  date: Sat, 04 Jul 2015 02:03:15 GMT  {      “commandList”: [          {              “commandListEntry”: {                  “command”: {                      “commandType”: “throttlingSpecification”,                      “eventDomainThrottleSpecification”: {                          “eventDomain”: “fault”,                          “suppressedFieldNames”: [                              “alarmInterfaceA”,                              “alarmAdditionalInformation”                          ]                      }                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “throttlingSpecification”,                      “eventDomainThrottleSpecification”: {                          “eventDomain”: “thresholdCrossingAlert”,                          “suppressedFieldNames”: [                              “associatedAlertIdList”,                              “possibleRootCause”                          ],                          “suppressedNvPairs” {                              “nvPairFieldName”: additionalParameters”,                              “suppressedNvPairNames”: [                                  “someCounterName”,                                  “someOtherCounterName”                              ]                          }                      }                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “measurementInternalChange”,                      “measurementInterval”: “600”                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “provideThrottlingState”                  }              }          }      ]  } |

#### Sample Error Responses

##### Sample Policy Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “policyException”: {  “messageId”: “POL9003”,  “text”: “Message content size exceeds the allowable limit”,  }  }  } |

##### Sample Service Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “serviceException”: {  “messageId”: “SVC2000”,  “text”: “Missing Parameter: %1. Error code is %2”  “variables”: [  “severity”,  “400”  ]  }  }  } |

## Operation: publishSpecificTopic

### Functional Behavior

Allows authorized clients to publish any single event to the generic event listener.

* Supports only secure HTTPS (one way SSL) access.
* Uses the HTTP verb POST
* Supports JSON content types
* Provides HTTP response codes as well as Service and Policy error messages
* Allows the event collector to use the HTTP response to command the event source to throttle event messages it may send in the future.

### Call Flow

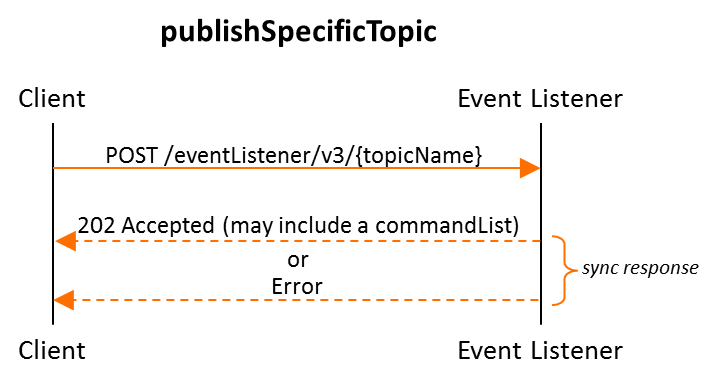


Figure 3 - publishSpecificTopic Call Flow

### Input Parameters

Querystring parameters:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| TopicName | string | Yes | Specifies the specific event topic which the event body must contain. |

Header Fields (note: all parameter names shall be treated as case-insensitive):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Accept | string | No | Determines the format of the body of the response. Valid values are:   * application/json |
| Authorization | string | Yes | The username and password are formed into one string as “username:password”. This string is then Base64 encoded to produce the encoded credential which is communicated in the header after the string “Authorization: Basic “. See examples below. If the Authorization header is missing, then an HTTP 400 Invalid Request message shall be returned. If the string supplied is invalid, then an HTTP 401 Unauthorized message shall be returned. |
| Content-length | integer | No | Note that content length is limited to 1Megabyte. |
| Content-type | string | Yes | Must be set to one of the following values:   * application/json |

Body Fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Event | event | Yes | Contains the JSON structure of the common event format. |

### Output Parameters

Header fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Content-length | integer | No | Used only in error conditions. |
| Content-type | string | No | Used only in error conditions |
| Date | datetime | Yes | Date time of the response in GMT |

Body Fields (for success responses without a commandList): no content is provided and the header fields are not required.

Body Fields (for success responses with one or more commands from the event collector toward the event source):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| commandList | commandList | No | Array of commands (e.g., measurement Interval changes and/or what fields to suppress within specified event domains and/or a request to report the state of event throttling by event domain that is currently in force in the event source). Note: for ‘provideThrottlingState’ commands, the client should subsequently provide the throttling state by calling the provideThrottlingState operation. |

Body Fields (for error Responses):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| requestError | requestError | Yes (for errors) | Used only in error conditions. |

### HTTP Status Codes

|  |  |  |
| --- | --- | --- |
| *Code* | *Reason Phrase* | *Description* |
| 202 | Accepted | The request has been accepted for processing |
| 400 | Bad Request | Many possible reasons not specified by the other codes (e.g., missing required parameters or incorrect format). The response body may include a further exception code and text. HTTP 400 errors may be mapped to SVC0001 (general service error), SVC0002 (bad parameter), SVC2000 (general service error with details) or PO9003 (message content size exceeds the allowable limit). |
| 401 | Unauthorized | Authentication failed or was not provided. HTTP 401 errors may be mapped to POL0001 (general policy error) or POL2000 (general policy error with details). |
| 404 | Not Found | The server has not found anything matching the Request-URI. No indication is given of whether the condition is temporary or permanent. |
| 405 | Method Not Allowed | A request was made of a resource using a request method not supported by that resource (e.g., using PUT on a REST resource that only supports POST). |
| 500 | Internal Server Error | The server encountered an internal error or timed out; please retry (general catch-all server-side error).HTTP 500 errors may be mapped to SVC1000 (no server resources). |

### Sample Request and Response

#### Sample Request

|  |
| --- |
| POST /eventListener/v3/Fault\_MobileCallRecording\_PilotNumberPoolExhaustion HTTPS/1.1  Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==  content-type: application/json  content-length: 12345 {  "event": {  "commonEventHeader": {  “domain”: “fault”,  "eventType": "Fault\_MobileCallRecording\_PilotNumberPoolExhaustion",  "eventId": "ab305d54-85b4-a31b-7db2-fb6b9e546015",  "sequence": "0",  "priority": "High",  “reportingEntityId”: “cc305d54-75b4-431b-adb2-eb6b9e541234”,  “reportingEntityName”: “EricssonOamVf”,  "sourceId": "de305d54-75b4-431b-adb2-eb6b9e546014",  “sourceName”: “EricssonECE”,  “functionalRole”: “SCF”,  “startEpochMicrosec”: “1413378172000000”,  “lastEpochMicrosec”: “1413378172000000”  },  "faultFields": {  "alarmCondition": "PilotNumberPoolExhaustion",  "eventSourceType": "other",  "specificProblem": "Calls cannot complete because pilot numbers are unavailable"  "eventSeverity": "CRITICAL",  “vfStatus”: “Active”  }  }  } |

#### Sample Success Response #1

For success responses without a provided commandList:

|  |
| --- |
| HTTPS/1.1 202 Accepted |

#### Sample Success Response #2

For success responses with a provided commandList:

|  |
| --- |
| HTTPS/1.1 202 Accepted  content-type: application/json  content-length: nnn  date: Sat, 04 Jul 2015 02:03:15 GMT  {      “commandList”: [          {              “commandListEntry”: {                  “command”: {                      “commandType”: “throttlingSpecification”,                      “eventDomainThrottleSpecification”: {                          “eventDomain”: “fault”,                          “suppressedFieldNames”: [                              “alarmInterfaceA”,                              “alarmAdditionalInformation”                          ]                      }                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “throttlingSpecification”,                      “eventDomainThrottleSpecification”: {                          “eventDomain”: “thresholdCrossingAlert”,                          “suppressedFieldNames”: [                              “associatedAlertIdList”,                              “possibleRootCause”                          ],                          “suppressedNvPairs” {                              “nvPairFieldName”: additionalParameters”,                              “suppressedNvPairNames”: [                                  “someCounterName”,                                  “someOtherCounterName”                              ]                          }                      }                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “measurementInternalChange”,                      “measurementInterval”: “600”                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “provideThrottlingState”                  }              }          }      ]  } |

#### Sample Error Responses

##### Sample Policy Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “policyException”: {  “messageId”: “POL9003”,  “text”: “Message content size exceeds the allowable limit”,  }  }  } |

##### Sample Service Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “serviceException”: {  “messageId”: “SVC2000”,  “text”: “Missing Parameter: %1. Error code is %2”  “variables”: [  “severity”,  “400”  ]  }  }  } |

## Operation: publishEventBatch

### Functional Behavior

Allows authorized clients to publish any single event to the generic event listener.

* Supports only secure HTTPS (one way SSL) access.
* Uses the HTTP verb POST
* Supports JSON content types
* Provides HTTP response codes as well as Service and Policy error messages
* Allows the event collector to use the HTTP response to command the event source to throttle event messages it may send in the future.

### Call Flow

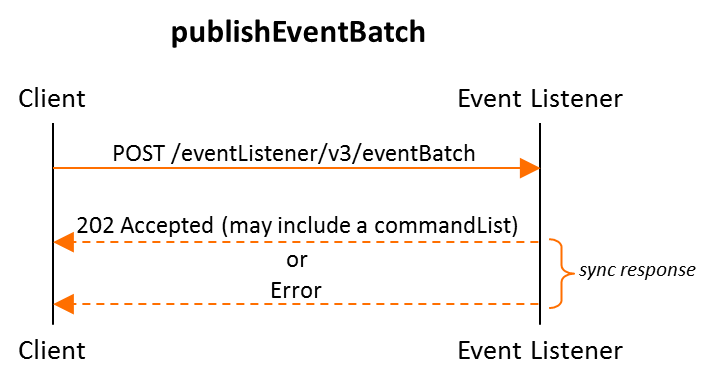


Figure 4 – publishEventBatch Call Flow

### Input Parameters

Header Fields (note: all parameter names shall be treated as case-insensitive):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Accept | string | No | Determines the format of the body of the response. Valid values are:   * application/json |
| Authorization | string | Yes | The username and password are formed into one string as “username:password”. This string is then Base64 encoded to produce the encoded credential which is communicated in the header after the string “Authorization: Basic “. See examples below. If the Authorization header is missing, then an HTTP 400 Invalid Request message shall be returned. If the string supplied is invalid, then an HTTP 401 Unauthorized message shall be returned. |
| Content-length | integer | No | Note that content length is limited to 1Megabyte. |
| Content-type | string | Yes | Must be set to one of the following values:   * application/json |

Body Fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| eventList | eventList | Yes | Array of events conforming to the common event format. |

### Output Parameters

Header fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Content-length | integer | No | Used only in error conditions. |
| Content-type | string | No | Used only in error conditions |
| Date | datetime | Yes | Date time of the response in GMT |

Body Fields (for success responses without a commandList): no content is provided and the header fields are not required.

Body Fields (for success responses with one or more commands from the event collector toward the event source):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| commandList | commandList | No | Array of commands (e.g., measurement Interval changes and/or what fields to suppress within specified event domains and/or a request to report the state of event throttling by event domain that is currently in force in the event source). Note: for ‘provideThrottlingState’ commands, the client should subsequently provide the throttling state by calling the provideThrottlingState operation. |

Body Fields (for error Responses):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| requestError | requestError | Yes (for errors) | Used only in error conditions. |

### HTTP Status Codes

|  |  |  |
| --- | --- | --- |
| *Code* | *Reason Phrase* | *Description* |
| 202 | Accepted | The request has been accepted for processing |
| 400 | Bad Request | Many possible reasons not specified by the other codes (e.g., missing required parameters or incorrect format). The response body may include a further exception code and text. HTTP 400 errors may be mapped to SVC0001 (general service error), SVC0002 (bad parameter), SVC2000 (general service error with details) or PO9003 (message content size exceeds the allowable limit). |
| 401 | Unauthorized | Authentication failed or was not provided. HTTP 401 errors may be mapped to POL0001 (general policy error) or POL2000 (general policy error with details). |
| 404 | Not Found | The server has not found anything matching the Request-URI. No indication is given of whether the condition is temporary or permanent. |
| 405 | Method Not Allowed | A request was made of a resource using a request method not supported by that resource (e.g., using PUT on a REST resource that only supports POST). |
| 500 | Internal Server Error | The server encountered an internal error or timed out; please retry (general catch-all server-side error).HTTP 500 errors may be mapped to SVC1000 (no server resources). |

### Sample Request and Response

#### Sample Request

|  |
| --- |
| POST /eventListener/v3/eventBatch HTTPS/1.1  Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==  content-type: application/json  content-length: 12345 {  "eventList": [  {  "commonEventHeader": {  “domain”: “fault”,  "eventType": "Fault\_MobileCallRecording\_PilotNumberPoolExhaustion",  "eventId": "ab305d54-85b4-a31b-7db2-fb6b9e546015",  "sequence": "0",  "priority": "High",  “reportingEntityId”: “cc305d54-75b4-431b-adb2-eb6b9e541234”,  “reportingEntityName”: “EricssonOamVf”,  "sourceId": "de305d54-75b4-431b-adb2-eb6b9e546014",  “sourceName”: “EricssonECE”,  “functionalRole”: “SCF”,  “startEpochMicrosec”: “1413378172000000”,  “lastEpochMicrosec”: “1413378172000000”  },  "faultFields": {  "alarmCondition": "PilotNumberPoolExhaustion",  "eventSourceType": "other",  "specificProblem": "Calls cannot complete - pilot numbers are unavailable"  "eventSeverity": "CRITICAL",  “vfStatus”: “Active”  }  },  {  "commonEventHeader": {  “domain”: “fault”,  "eventType": "Fault\_MobileCallRecording\_RecordingServerUnreachable",  "eventId": "ab305d54-85b4-a31b-7db2-fb6b9e546025",  "sequence": "0",  "priority": "High",  “reportingEntityId”: “cc305d54-75b4-431b-adb2-eb6b9e541234”,  “reportingEntityName”: “EricssonOamVf”,  "sourceId": "de305d54-75b4-431b-adb2-eb6b9e546014",  “sourceName”: “EricssonECE”,  “functionalRole”: “SCF”,  “startEpochMicrosec”: “1413378172000010”,  “lastEpochMicrosec”: “1413378172000010”  },  "faultFields": {  "alarmCondition": "RecordingServerUnreachable",  "eventSourceType": "other",  "specificProblem": "Recording server unreachable"  "eventSeverity": "CRITICAL",  “vfStatus”: “Active”  }  }  ]  } |

#### Sample Success Response #1

For success responses without a provided commandList:

|  |
| --- |
| HTTPS/1.1 202 Accepted |

#### Sample Success Response #2

For success responses with a provided commandList:

|  |
| --- |
| HTTPS/1.1 202 Accepted  content-type: application/json  content-length: nnn  date: Sat, 04 Jul 2015 02:03:15 GMT  {      “commandList”: [          {              “commandListEntry”: {                  “command”: {                      “commandType”: “throttlingSpecification”,                      “eventDomainThrottleSpecification”: {                          “eventDomain”: “fault”,                          “suppressedFieldNames”: [                              “alarmInterfaceA”,                              “alarmAdditionalInformation”                          ]                      }                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “throttlingSpecification”,                      “eventDomainThrottleSpecification”: {                          “eventDomain”: “thresholdCrossingAlert”,                          “suppressedFieldNames”: [                              “associatedAlertIdList”,                              “possibleRootCause”                          ],                          “suppressedNvPairs” {                              “nvPairFieldName”: additionalParameters”,                              “suppressedNvPairNames”: [                                  “someCounterName”,                                  “someOtherCounterName”                              ]                          }                      }                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “measurementInternalChange”,                      “measurementInterval”: “600”                  }              }          },          {              “commandListEntry”: {                  “command”: {                      “commandType”: “provideThrottlingState”                  }              }          }      ]  } |

#### Sample Error Responses

##### Sample Policy Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “policyException”: {  “messageId”: “POL9003”,  “text”: “Message content size exceeds the allowable limit”,  }  }  } |

##### Sample Service Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “serviceException”: {  “messageId”: “SVC2000”,  “text”: “Missing Parameter: %1. Error code is %2”  “variables”: [  “severity”,  “400”  ]  }  }  } |

## Operation: provideThrottlingState

### Functional Behavior

Allows authorized event source clients to report the state of event throttling by event domain that is currently in force in the event source.

* Supports only secure HTTPS (one way SSL) access.
* Uses the HTTP verb POST
* Supports application/json content types
* Provides HTTP response codes as well as Service and Policy error messages

### Call Flow

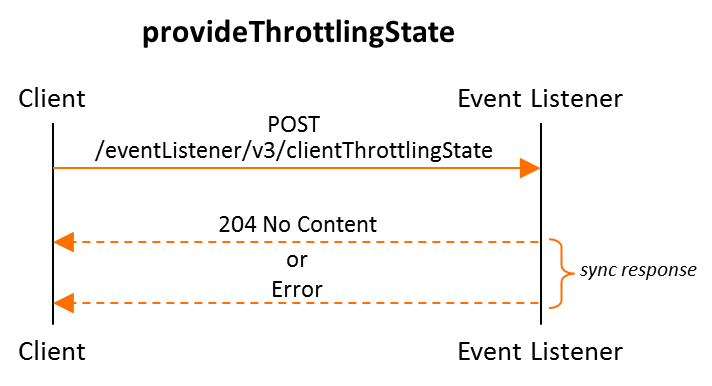


Figure 5 - provideClientThrottlingState Call Flow

### Input Parameters

Header Fields (note: all parameter names shall be treated as case-insensitive):

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Accept | string | No | Determines the format of the body of the response. Valid values are:   * application/json |
| Authorization | string | Yes | The username and password are formed into one string as “username:password”. This string is then Base64 encoded to produce the encoded credential which is communicated in the header after the string “Authorization: Basic “. See examples below. If the Authorization header is missing, then an HTTP 400 Invalid Request message shall be returned. If the string supplied is invalid, then an HTTP 401 Unauthorized message shall be returned. |
| Content-length | integer | No | Note that content length is limited to 1Megabyte. |
| Content-type | string | Yes | Must be set to one of the following values:   * application/json |

Body Fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| eventThrottlingState | eventThrottlingState | Yes | Consists of an eventThrottlingrMode enumeration which can be ‘normal’ or ‘throttled’ followed by an optional array of eventDomainThrottlingSpecification structures |

### Output Parameters

The only output parameters are an HTTP response code and message.

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| Content-length | integer | No | Used only in error conditions. |
| Content-type | string | No | Used only in error conditions. |

Body Fields:

| **Parameter** | **Data Type** | **Required?** | **Brief description** |
| --- | --- | --- | --- |
| requestError | requestError | No | Used only in error conditions. |

### HTTP Status Codes

|  |  |  |
| --- | --- | --- |
| *Code* | *Reason Phrase* | *Description* |
| 204 | No Content | The throttling state update message has been accepted. |
| 400 | Bad Request | Many possible reasons not specified by the other codes (e.g., missing required parameters or incorrect format). The response body may include a further exception code and text. HTTP 400 errors may be mapped to SVC0001 (general service error), SVC0002 (bad parameter), SVC2000 (general service error with details) or PO9003 (message content size exceeds the allowable limit). |
| 401 | Unauthorized | Authentication failed or was not provided. HTTP 401 errors may be mapped to POL0001 (general policy error) or POL2000 (general policy error with details). |
| 404 | Not Found | The server has not found anything matching the Request-URI. No indication is given of whether the condition is temporary or permanent. |
| 405 | Method Not Allowed | A request was made of a resource using a request method not supported by that resource (e.g., using PUT on a REST resource that only supports POST). |
| 409 | Locked | The request could not be completed due to a conflict with the current state of the resource. |
| 500 | Internal Server Error | The server encountered an internal error or timed out; please retry (general catch-all server-side error).HTTP 500 errors may be mapped to SVC1000 (no server resources). |
| 503 | Service Unavailable | The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. The implication is that this is a temporary condition which will be alleviated after some delay. |
| 504 | Gateway Timeout | The server, while acting as a gateway or proxy, did not receive a timely response from the upstream process. |

### Sample Request and Response

#### Sample Request

|  |
| --- |
| POST /eventListener/v3/clientThrottlingState HTTPS/1.1  Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==  content-type: application/json  content-length: nnn  accept: application/json  {  “eventThrottlingState”: {  “eventThrottlingMode”: “throttled”,  “eventDomainThrottleSpecificationList”: [  {  “eventDomainThrottleSpecification”: {  “eventDomain”: “fault”,  “suppressedFieldNames”: [  “alarmInterfaceA”,  “alarmAdditionalInformation”  ]  }  },  {  “eventDomainThrottleSpecification”: {  “eventDomain”: “thresholdCrossingAlert”,  “suppressedFieldNames”: [  “associatedAlertIdList”,  “possibleRootCause”  ],  “suppressedNvPairsList”: [  {  “suppressedNvPairs” {  “nvPairFieldName”: additionalParameters”,  “suppressedNvPairNames”: [  “someCounterName”,  “someOtherCounterName”  ]  }  }  ]  }  }  ]  }  } |

#### Sample Success Response

|  |
| --- |
| HTTPS/1.1 204 No Content |

#### Sample Error Responses

##### Sample Policy Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “policyException”: {  “messageId”: “POL9003”,  “text”: “Message content size exceeds the allowable limit”,  }  }  } |

##### Sample Service Exception

|  |
| --- |
| HTTPS/1.1 400 Bad Request  content-type: application/json  content-length: 12345  Date: Thu, 04 Jun 2009 02:51:59 GMT  {  “requestError”: {  “serviceException”: {  “messageId”: “SVC2000”,  “text”: “Missing Parameter: %1. Error code is %2”  “variables”: [  “severity”,  “400”  ]  }  }  } |