Yamcs: SLE Plugin

Release 1.5.1-SNAPSHOT

Yamcs
MISSION CONTROL

Space Applications Services, NV/SA
Leuvensesteenweg 325
1932 Sint-Stevens-Woluwe
Belgium
spaceapplications.com
yamcs.org

Aerospace Applications North America, Inc.
16850 Saturn Ln, Ste 100
Houston, TX 77058
United States of America
aerospaceapplications-na.com

Contents

1. About Yamcs: SLE Plugin

This plugin extends Yamcs with links to connect via the CCSDS SLE (Space Link Extension) protocol to SLE-enabled ground stations. These are typically the ground stations of national space agencies.

The supported services are:

- CCSDS Return All Frames (RAF) specified in CCSDS 911.1-B-4[1].
- CCSDS Return Chanel Frames (RCF) specified in 911.2-B-3[2].
- CCSDS Forward CLTU (FCLTU) specified in CCSDS 912.1-B-4[3].

The services are supported by the SLE Internet Protocol for transfer services (ISP1) specified in CCSDS 913.1-B-2[4].

The RAF and RCF services are further divided into:

- **online timely:** used for retrieval of frames with a guaranteed timeliness of data - if the receiver of the data cannot process the data fast enough (or the network link towards the receiver is too slow), the provider will drop data.
- **online complete:** used for retrieval of frames when the receiver wants to receive complete data at the expense of timeliness. The provider will buffer the data if the receiver is too slow.
- **offline:** used for retrieval of frames stored at the provider (ground station).

Usage with Maven

Add the following dependency to your Yamcs Maven project. Replace `x.y.z` with the latest version. See https://mvnrepository.com/artifact/org.yamcs/yamcs-sle

```
<dependency>
  <groupId>org.yamcs</groupId>
  <artifactId>yamcs-sle</artifactId>
  <version>x.y.z</version>
</dependency>
```

[1] https://public.ccsds.org/Pubs/911x1b4.pdf
[2] https://public.ccsds.org/Pubs/911x2b3.pdf
[3] https://public.ccsds.org/Pubs/912x1b4.pdf
[4] https://public.ccsds.org/Pubs/913x1b2.pdf

2. Provider Configuration

SLE Providers are configured in `etc/sle.yaml`. Data link configuration in yamcs.instance.yaml refer to one of the providers defined in this file.

An example of provider configuration is here below:

```
CommonSettings: &CommonSettings
    hashAlgorithm: "SHA-1"
    authLevel: BIND
    versionNumber: 4
    myUsername: "user01"
    myPassword:  01020304ABCD
    initiatorId: "user01"
    responderPortId: "ABC"
    heartbeatInterval: 30
    heartbeatDeadFactor: 3

Providers:
    GS1:
        <<: *CommonSettings
        peerUsername: "prov-user"
        peerPassword: AB0102030405060708090a0b0c0d0e0f

        cltu:
            host: 172.16.1.113
            port: 63800
            serviceInstance: "sagr=9999.spack=ABC-PERM.fsl-fg=1.cltu=cltu1"

        raf-onlt:
            host: 172.16.1.113
            port: 63900
            serviceInstance: "sagr=9999.spack=ABC-PERM.rsl-fg=1.raf=onlt1"

        raf-onlc:
            host: 172.16.1.113
            port: 63901
            responderPortId: "ABC901"
            serviceInstance: "sagr=9999.spack=ABC-PERM.rsl-fg=1.raf=onlc1"
```

The section `Providers` contains a map with each SLE provider (e.g. ground station). This contains a section with general settings followed by a specific cltu, raf-ontl, raf-onlc, rcf-ontl, rcf-onlc sections containing connection parameters for the CLTU service, RAF/RCF online timely, respectively RAF/RCF online complete services.

In the example above the `CommonSettings` block is included into the GS1 provider block using the yaml node achor and reference feature. This feature (pure yaml functionality, not specific to SLE) allows grouping common settings for multiple ground stations into one definition.

2.1 Common Options

These options are specified at the provider level (directly under `GS1` in the example above) as they apply to all services from this provider. All the options which do not have a default, are mandatory. Some options can be specified both at provider and at service level, the service level values overriding the provider level values.

versionNumber (integer)
2, 3, 4 or 5. This is the version number passed in the BIND call. Note that there is no code to handle specifically any version but the SLE versions 2-5 SLE are belived to be compatible as far as the user (i.e. Yamcs) is concerned; the differences are mostly to do with new options being added in the newer versions. Those options are currently not accessible from the Yamcs SLE link. Default: `5`

hashAlgorithm (string)

One of `SHA-1` or `SHA-256`.
The hashAlgorithm is effectively passed to the MessageDigest.getInstance(hashAlgorithm)[5] method in Java. Default: `SHA-256`

authLevel (string)

One of: `ALL`, `BIND` or `NONE`.
The SLE messages can be optionally authenticated by inserting a `invoker-credentials` token in the SLE message. This option specifies which messages sent by Yamcs are authenticated (i.e. contain this token). It also specifies which of the peer messages are expected to be authenticated. Default: `BIND`
Important: the SLE security mechanism does not prevent man-in-the-middle attacks and other type of security attacks and therefore it is advisable to protect the transport at TCP level by other means (e.g. VPN).
For details: see Chapter 8 Security Aspects of the SLE Forward CLTU Transfer Service in the CCSDS specification.

authenticationDelay
The authentication tokens received from the provider contain a timestamp which is compared with the local time to verify that the token has been recently generated. This option configures the maximum allowed delta in seconds. Default: 180 (3 minutes)

myUsername (string)
The username used to build the credentials token part of ISP1 authentication if the authLevel is ALL or BIND.

myPassword (hexadecimal string)
Used together with the myUsername for the ISP1 authentication.

peerUsername (string)
The username of the peer. It is used to verify the peer credential tokens. Depending on the `authLevel`, all messages, the bind return or no message will be authenticated.

peerPassword (hexadecimal string)
Used together with the `peerUsername` for verifying the peer suplied credential token.

initiatorId (string)
This property species the value of the `initiator-identifier` parameter passed as part of the BIND SLE message.

responderPortId (string)
This property species the value of the `responder-port-identifier` parameter passed as part of the BIND SLE message.

heartbeatInterval (integer number of seconds)
This property specifies the value proposed to the peer for the ISP1 heartbeat interval. The SLE provider will check this against its accepted range and it will close the connection if the value sent by Yamcs is not within the accepted range. Default: `30` (seconds)

heartbeatDeadFactor (integer)
Defines the number of heartbeat intervals without a message from peer after which a TCP connection is considered dead and is closed. Default: `3`

[5] https://docs.oracle.com/javase/8/docs/api/java/security/MessageDigest.html#getInstance-java.lang.String

reconnectionIntervalSec (integer)
How soon should reconnect in case the connection to the SLE provider is lost; If negative, do not reconnect. Default: `30` (seconds). The setting is not applicable for the offline TM link, it is set internally to -1.

maxShutdownDelaySec (integer)
When the SLE connection is stopped, a graceful shutdown is performed by sending STOP and UNBIND messages. This option defines the maximum time for the graceful shutdown to take place. If the shutdown is not finished in this time, the connection will be closed nevertheless. Default: `5` (seconds)

unbindReason (string)
The reason to send in the UNBIND call. One of END, SUSPEND or OTHER. Default: `SUSPEND`

2.2 Service-specific Options

These options are specified as part of each SLE service (e.g. under `cltu` in the example above). The options which have the same names with those defined above in the provider section, will override those for the specific service.

host (string)
The hostname or IP address to connect to.

port (integer)
The port number to connect to.

serviceInstance (string)
Used (after transformation to binary form) as `service-instance-identifier` in the SLE BIND call to identify the service requested to the provider. It is a series of `sia=value` separated by dots where sia is a service identifier attribute.

Ask your SLE provider for the value of this parameter.

tmlMaxLength (integer)
The maximum length in bytes of the Transport Mapping Layer (TML) messages. These are the messages defined in the ISP1 standard for transporting SLE data. If a message larger than this length is received, the connection is closed.

On the ESA SLE provider this is configured by the `transfer-buffer-size` parameter which sets the number of frames which can be transferred in one message. The tmlMaxLength should be set to accomodate that number of frames taking into account the frame size and some 70 bytes overhead per frame. Default: `307200` (300KB)

responderPortId (string)
Overrides the setting set at the provider level, see above.

reconnectionIntervalSec (integer)
Overrides the setting set at the provider level, see above.

maxShutdownDelaySec (integer)
Overrides the setting set at the provider level, see above.

unbindReason (string)
Overrides the setting set at the provider level, see above.

3. Links

3.1 TM Link

Online RAF service. This link binds and starts a SLE session as soon as it is enabled. If the connection goes down it will continually try to re-establish it.

3.1.1 Usage

Specify SLE properties in `etc/sle.yaml`, and add a link entry to `etc/yamcs.[instance].yaml`:

```
dataLinks:
    - name: SLE_IN
      class: org.yamcs.sle.TmSleLink
      sleProvider: GS1
      deliveryMode: timely
      startSleOnEnable: false
      # other options
```

3.1.2 Options

sleProvider (string)
: **Required.** The name of a provider defined in the `etc/sle.yaml` configuration file.

service (string)
: One of: `RAF` or `RCF`. Default RAF.

deliveryMode (string)
: **Required.** One of: `complete` or `timely`.

startSleOnEnable (boolean)
: Whether the SLE session should automatically be STARTed when the link is enabled. If `false`, enabling the link will only BIND it. Default: `true`

service (string)
: One of: `RAF` or `RCF`.

  Depending on this setting and the `deliveryMode`, one of the entries raf-ontl, raf-onlc, rcf-ontl, rcf-onlc from sle.yaml will be used.

  If the RCF service is used, the request sent with the SLE START includes the triplet (TransferFrameVersionNumber, SpacecraftId, VirtualChannelId).

  The values for the TransferFrameVersionNumber and SpacecraftId parameters are normally derived from the frame processing configuration but can be overriden by the options below. Overriding them will most likely result in an invalid configuration.

  The value of the VirtualChannelId is by default -1 meaning all VCs are requested but can be restricted to only one VC with the option below.

  Default: `RAF`

rcfTfVersion (integer)
: If `service` is set to `RCF`, this overrides the Transfer Frame Version Number which is otherwise derived from the `frameType` parameter part of the frame processing configuration.

rcfSpacecraftId (integer)
: If `service` is set to `RCF`, this overrides the Spacecraft Id which is otherwise the one specified `spacecraftId` parameter part of the frame processing configuration.

rcfVcId
: If `service` is RCF, this specifies the Virtual Channel requested via RCF. By default it is -1 meaning all Virtual Channels for the defined spacecraft. There is validation that this virtual channel is defined in the `virtualChannels` parameter part of the frame processing configuration.

frameQuality
: If `service` is RAF, this specifies the frame quality requested in the SLE START invocation. Valid values are `goodFramesOnly`, `erredFramesOnly` or `allFrames`. RCF does not support this option, it only delivers good frames. Default: `goodFramesOnly`

reconnectionIntervalSec (integer)
: Select the reconnection interval. If the connection to the SLE provider breaks, Yamcs will attempt to reconnect after that many seconds.

  Default: `30`

Note: Other available link options are general frame processing parameters as specified at https://docs.yamcs.org/yamcs-server-manual/links/ccsds-frame-processing.

3.2 TC Link

CLTU service. Like *TM Link* (page 5), this link tries to bind and start a SLE session for as long as it is enabled.

3.2.1 Usage

Specify SLE properties in `etc/sle.yaml`, and add a link entry to `etc/yamcs.[instance].yaml`:

```
dataLinks:
    - name: SLE_OUT
      class: org.yamcs.sle.TcSleLink
      sleProvider: GS1
      startSleOnEnable: false
      # other options
```

3.2.2 Queing behaviour

This link does not queue any command frames internally. It expects the SLE provider to queue up to `maxPandingFrames` command frames. This is required in order to be able to properly fill the uplink, otherwise the provider may not get the next frame in time and will need to fill in some dummy data (according to the PLOP in effect).

When the link is enabled and the SLE connection is estabilished, the link performs the following steps in a loop:

1. Get a command frame from the multiplexer, waiting if necessary until one becomes available.
2. If the uplink is available and the number of commands in the provider does not exceed the maxPandingFrames send the command to the provider.

3. Otherwise wait for `waitForUplinkMsec` milliseconds. If the condition is still not met, drop the frame. If the frame has the by-pass flag set (BD frame), all the commands inside (which at this time is only one since Yamcs does not send BD frames with multiple commands) will be negatively acknowledged (otherwise the COP1 will take care of the acknowledgments).

The multiplexer will provide commands from different virtual channels and the virtual channel sub-links (one per virtual channel) will have their own queueing in operation. If COP1 is used, then the COP1 specific queueing settings will be used (taking into account the overall number of un-acknowledged frames), otherwise a simple queue size is used. The https://docs.yamcs.org/yamcs-server-manual/links/ccsds-frame-processing provides more details.

Thus there can be a number of commands pending transmission, some in the specific virtual channel sub-links, some in the SLE Provider (ground-station). The `SLE_REQ` command history attribute will be set for the commands that have been sent to the SLE Provider.

Note that if the SLE TC link is disabled, the multiplexer will not be queried for new frames, so the commands may queue in the sub-links, indefinitely. This will only happen however, if the main SLE TC link is disabled and the virtual channel sub-links enabled.

3.2.3 Options

sleProvider (string)
: **Required.** The name of a provider defined in the `etc/sle.yaml` configuration file.

waitForUplinkMsec (integer)
: If a command is received and the uplink is not available, wait this number of milliseconds for the uplink to become available. If 0 or negative, then drop the command immediately. Default: `5000` (5 seconds)

maxPendingFrames:
: Maximum number of pending frames in the SLE provider (waiting or being uplinked). If this number is reached we start rejecting new frames but only after waiting waitForUplinkMsec before each frame. Default: `20`.

startSleOnEnable (boolean)
: Whether the SLE session should automatically be STARTed when the link is enabled. If `false`, enabling the link will only BIND it. Default: `true`

reconnectionIntervalSec (integer)
: Select the reconnection interval. If the connection to the SLE provider breaks, Yamcs will attempt to reconnect after that many seconds.

  Default: `30`

Note: Other available link options are general frame processing parameters as specified at https://docs.yamcs.org/yamcs-server-manual/links/ccsds-frame-processing.

3.3 Offline TM Link

Offline RAF/RCF service. This link does not retrieve any data by itself. Instead it expects requests to be made using the *Request Offline Data* (page 9) HTTP API. As long as it has retrieval requests in the queue, it keeps a connection open and bound to the SLE provider and it starts/stops SLE sessions for each retrieval request. After all the retrievals have been finished, it unbinds and closes the connection to the SLE provider. If a request fails for whatever reason (for example could not connect to SLE provider), it does not reattempt to execute it.

3.3.1 Usage

Specify SLE properties in `etc/sle.yaml`, and add a link entry to `etc/yamcs.[instance].yaml`:

```
dataLinks:
    - name: SLE_OFFLINE_IN
      class: org.yamcs.sle.OfflineTmSleLink
      sleProvider: GS1
      # other options
```

3.3.2 Options

sleProvider (string)
: **Required.** The name of a provider defined in the `etc/sle.yaml` configuration file.

service (string)
: One of: `RAF` or `RCF`.

  Depending on this setting and the `deliveryMode`, one of the entries raf-ontl, raf-onlc, rcf-ontl, rcf-onlc from sle.yaml will be used.

  If the RCF service is used, the request sent with the SLE START includes the triplet (TransferFrameVersionNumber, SpacecraftId, VirtualChannelId).

  The values for the TransferFrameVersionNumber and SpacecraftId parameters are normally derived from the frame processing configuration but can be overriden by the options below. Overriding them will most likely result in an invalid configuration.

  The value of the VirtualChannelId is by default -1 meaning all VCs are requested but can be restricted to only one VC with the option below.

  Default: `RAF`

rcfTfVersion (integer)
: If `service` is set to `RCF`, this overrides the Transfer Frame Version Number which is otherwise derived from the `frameType` parameter part of the frame processing configuration.

rcfSpacecraftId (integer)
: If `service` is set to `RCF`, this overrides the Spacecraft Id which is otherwise the one specified `spacecraftId` parameter part of the frame processing configuration.

rcfVcId
: If `service` is RCF, this specifies the Virtual Channel requested via RCF. By default it is -1 meaning all Virtual Channels for the defined spacecraft. There is validation that this virtual channel is defined in the `virtualChannels` parameter part of the frame processing configuration.

frameQuality
: If `service` is RAF, this specifies the frame quality requested in the SLE START invocation. Valid values are `goodFramesOnly`, `erredFramesOnly` or `allFrames`. RCF does not support this option, it only delivers good frames. Default: `goodFramesOnly`

Note: Other available link options are general frame processing parameters as specified at https://docs.yamcs.org/yamcs-server-manual/links/ccsds-frame-processing.

---

4. HTTP API

4.1 Request Offline Data

Request offline data.

This request can be used to obtain offline data from an SLE provider. The link has to be of type `org.yamcs.sle.OfflineTmSleLink`.

`start` and `stop` are passed as parameters to the RAF-START SLE request. The SLE provider will deliver all frames having their Earth Reception Time (ERT) in the [start, stop] time interval, both ends are inclusive. SLE requires that start is strictly smaller than stop so `start=stop` is not accepted.

The time passed via API is at nanosecond resolution whereas the time passed in the RAF-START SLE request is at microsecond or picosecond resolution (depending on the SLE version used).

Leap seconds

Time passed via the API is according to the Timestamp[6] protobuf message and is passed *unsmeared* to the SLE request. This is different from the normal requests to Yamcs where the time is transformed into internal Yamcs time (but at millisecond resolution!) by performing reverse smearing around the leap seconds.

This means that it is not possible to specify a retrieval that starts or stops on a leap second. For example `2016-12-31T23:59:60Z` will be effectively translated into `2017-01-01T00:00:00Z`.

URI Template

```
POST /api/sle/links/{instance}/{linkName}:requestOfflineData
```

{instance}
: Yamcs instance name

{linkName}
: Link name

Request Body

```
interface RequestOfflineDataRequest {

  // Start of the retrieval interval
  start: string;  // RFC 3339 timestamp

  // End of the retrieval interval
  stop: string;  // RFC 3339 timestamp
}
```

[6] https://protobuf.dev/reference/protobuf/google.protobuf/#timestamp

4.2 Cltu Throw Event

Invoke a CLTU Throw Event.

The link has to be of type `org.yamcs.sle.OfflineTmSleLink`.

URI Template

```
POST /api/sle/links/{instance}/{linkName}:throwEvent
```

{instance}
: Yamcs instance name

{linkName}
: Link name

Request Body

```
// The CCSDS FCLTU standard does not define parameters of the throw
// event invocation but gives some examples in annex F, see below.
//
// In these examples ``eventQualifier`` is an ASCII encoded string.
//
// Enable or disable command modulation of the carrier.
//   eventIdentifier: 1
//   eventQualifier: on|off
//
// Change bit rate to new bit rate defined by <BR> (requested new bit rate is the value of <BR> in bits/
↪second)
//   eventIdentifier:2
//   eventQualifier: br <BR> [7.8125 - 4000.0]
//
// Change modulation index to new modulation index angle defined by <MI>
// (requested new modulation index is the value of <MI> in milli-radians).
//   eventIdentifier: 3
//   eventQualifier: mi <MI> [1 - 1570]
//
// Change both bit rate and modulation index as described above.
//   eventIdentifier: 4
//   eventQualifier: br <BR> mi <MI>
//
// For convenience, the message below allows the event qualifier to be specified
// either as string or as binary. Only one of them can be used at a time.
// The SLE throw message requires a binary; the string will be converted to binary
// using UTF-8 encoding.
interface CltuThrowEventRequest {
  eventIdentifier: number;
  eventQualifier: string;
  eventQualifierBinary: string;  // Base64
}
```

Response Type

```
// If the CLTU Throw event is successful, the return contains only the
// event id. Otherwise the error field will contain the error.
interface CltuThrowEventResponse {

  // Event identifier passed in the request
  eventIdentifier: number;

  // Error, if there was one
  error: string;
}
```

4.3 Get Parameter

Invoke a SLE Get Parameter.

Can be used for all SLE links but some parameters make sense only for certain links.

URI Template

```
GET /api/sle/links/{instance}/{linkName}/{parameterName}
```

{instance}
: Yamcs instance name

{linkName}
: Link name

{parameterName}
: Parameter name

  One from org.yamcs.sle.ParameterName enum. Providing an invalid name will return a list with all parameters.

Request Body

```
interface GetParameterRequest {
}
```

Response Type

```
// The parameter response contains the parameter name and the value.
// The value can be of different types depending on the parameter.
//
// If the parameter is not available on the provider or if the link
// is not connected, the error field will be populated (if the parameter
// name is not correct, an HTTP 400 'Bad Request' error is returned instead).
//
// The parameter value returned by the provider may be null. In this case the
// response will contain only the parameter name with no other field populated
interface GetParameterResponse {

  // Parameter name
  parameterName: string;
  error: string;
  intValue: number;

  //the long value is used for the 32 bits unsigned parameters
  // The only parameter of that type is the CLTU minimumDelayTime which is the
  // minimum guard time the F-CLTU provider will accept between two consecutive
  // CLTUs. It is very likely that the value will be small enough (2147 seconds)
  // to fit into the intValue which means that the longValue will most certainly
  // never be used.
  longValue: string;  // String decimal
  stringValue: string;
}
```