Boundary Event SDK

User Guide

Boundary,Inc. **2014-07-11**

Table of Contents

1 Boundary Event SDK

Boundary Event SDK

Boundary Event SDK enables the rapid integration of foreign events into the Boundary event management platform.

The following sections provide information on getting the Boundary Event SDK up and running.

2 Overview

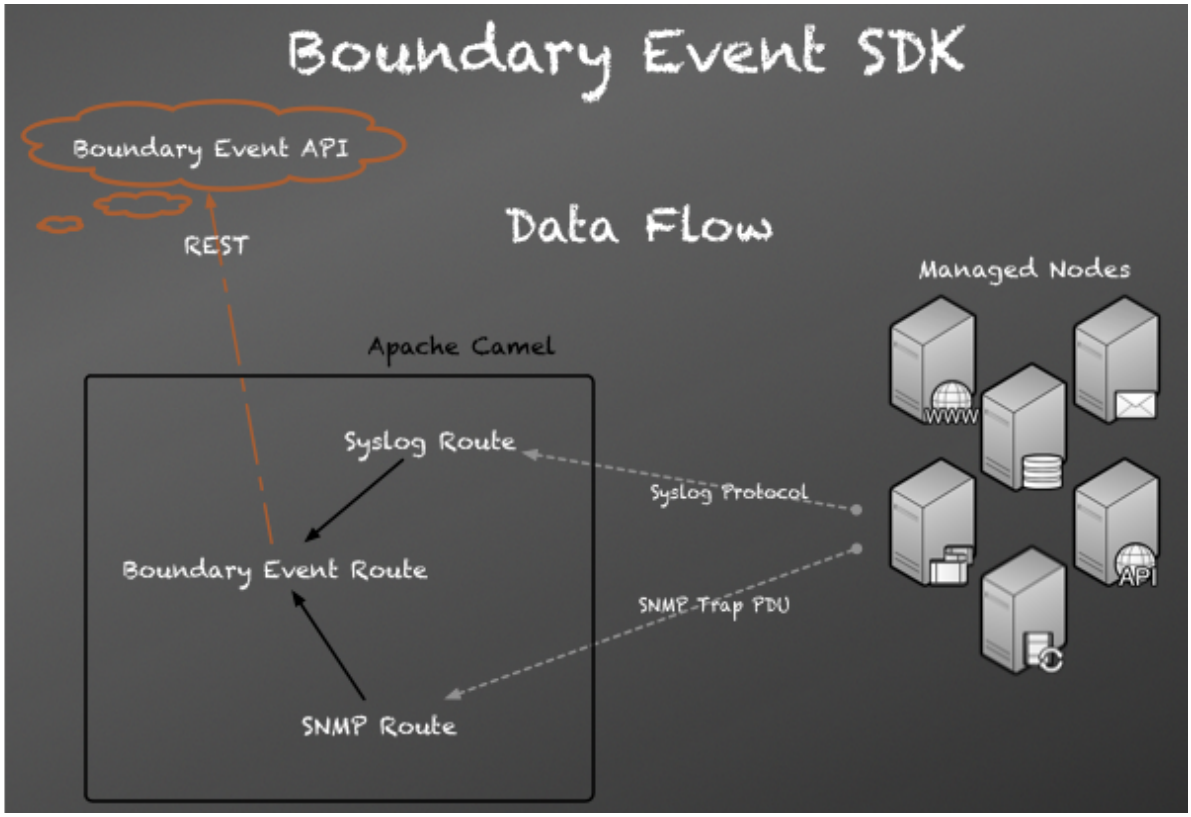
Overview

Boundary Event SDK provides a framework for introducing foreign messages and events into the Boundary platform. It has been developed using the open source framework Apache Camel.

A route in Apache Camel is a series of message processing steps which are loosely coupled. The Boundary Event SDK implementation consists of defining *route builders* that convert foreign messages to Boundary events. One route communicates with the foreign system using its native communication protocol (Sockets, REST, SOAP,etc) and translates the foreign system message into a Boundary *RawEvent* object. While another route recieves the RawEvent object where by it is translated into a *JSON* document and then sent the Boundary Event API via a REST call.

Two routes: one communicating with the foreign source and the other communicating to the Boundary Event API work cooperatively as an *Event Adapter*. This separation of concerns allows the creation of new Event Adapters in a rapid fashion since half the integration, posting of events to the Boundary Event API is already developed and tested.

The diagram belows shows a graphical representation of the flow of data from the managed nodes via foreign message protocols (SNMP trap PDUs, and Syslog protocol), through the Boundary Event SDK and ending at the Boundary Event APIs. The Boundary Event SDK hides the need to have deep technical knowledge of REST or JSON to run the pre-built adapters, since the pre-built adapters are configured in a declarative fashion using an XML file. Speciallized event adapters can be created but this requires that you have some knowledge of programming in Java, or Python.



3 User Guide

User Guide

This document describes how to install and use the Boundary Event SDK to integrate Syslog and SNMP into your Boundary environment.

3.1 Prerequisites

- Boundary Organizational Id
- Boundary API Key
- License key from snmp4j if you plan on receiving traps from the enterprise sub-branch and wish OIDs to mapped to readable strings.

3.2 Installation

1. Download the distribution from Boundary Event SDK Github.
2. Extract the archive:

```
$ cd tar xvf boundary-event-sdk-X.YY.ZZ-dist.tar.gz
```

3.3 Configuration

This sections provides the generatl configuration for the Boundary Event SDK. Specific instructions for each of the event adapters can found here.

3.3.1 Environment Variables

Boundary SDK requires the following the environment variables be set prior to running:

- `BOUNDARY_SDK_HOME` - Path to the extracted the Boundary SDK distribution
- `BOUNDARY_API_KEY` - Boundary API Key
- `BOUNDARY_ORG_ID` - Boundary Organization ID

with the following variables specific to the SNMP Trap Adapter:

- `BOUNDARY_MIB_REPOSITORY` - Stores the compiled MIBs for SNMP trap adapter
- `BOUNDARY_MIB_LICENSE` - License string for SNMP4J-SMI

These environment variables can be either set external or by setting directly in `BOUNDARY_SDK_HOME/etc/boundary-event-sdk`.

3.3.2 Set Runtime Environment

1. Add the required environment variables to the environment as discussed above.
2. Change directory to the distribution:

```
$ cd boundary-event-sdk-XX.YY.ZZ
```

3. Source the environment:

```
$ source etc/boundary-event-sdk
```

4. Verify the environment by running this command which displays the enviroment variables used by SDK and adapters:

```
$ benv
```

3.3.3 Starting the Event Daemon

1. Issue the command:

```
$ bin/beventd start
```

3.3.4 Stopping the Event Daemon

1. Issue the command:

```
$ bin/eventd stop
```

3.3.5 Log Files

The event daemon(`beventd`) logs output to `BOUNDARY_SDK_HOME/logs/beventd.log

4 Adapters

Boundary Event Adapters

Current list of event adapters powered by the Boundary Event SDK.

- SNMP Event Adapter
- Syslog Event Adapter

5 SNMP

Boundary SNMP Route

The SNMP route has the ability receive v1/v2c SNMP traps and translate into Boundary Events.

5.1 SNMP4J-SMI

The SNMP route uses the SNMP4J-SMI library to decode OIDs. For open source use free license is granted which restricts usage to standard MIB modules which are not under the enterprise OID sub-tree. Licenses can be purchased from www.snmp4j.org for $49 US, at the time of this writing.

5.2 Configuration

The sections the follow describe the configuration of the SNMP route.

5.2.1 Environment Variables

SNMP adapter parameters `mibRepository` and `license` can be set by providing values in the `BOUNDARY_SDK_HOME/etc/boundary-event-sdk` file.

5.2.2 Parameters

These parameters are configured in the `BOUNDARY_SDK_HOME/event-application.xml`.

- `bindAddress` - Address to bind the socket to, defaults to 0.0.0.0
- `license` - License string purchased from SNMP4J-SMI
- `mibRepository` - Indicates the end point to send the transformed SNMP trap.
- `port` - Port number to listen for SNMP traps (default is 1162)
- `routeId` - Name to identify the route in the logs
- `startOrder` - Ordering of when this route is started in relationship to other routes
- `toUri` - Indicates the end point to send the transformed SNMP trap.

5.2.3 Example Configuration

```
        <!-- This route recieves v1 and v2c SNMP traps -->
        <bean id="snmp-route" class="com.boundary.sdk.event.snmp.SNMPRouteBuilder">
                <property name="routeId" value="SNMP"/>
                <property name="startUpOrder" value="130"/>
                <property name="port" value="1162"/>
                <property name="mibRepository" value="#{systemEnvironment['BOUNDARY
                <property name="license" value="#{systemEnvironment['BOUNDARY_MIB_L
                <property name="toUri" value="seda:boundary-event"/>
        </bean>
```

5.2.4 Event Mapping

This section describes the mapping of the SNMP trap to a Boundary event.

A SNMP v1 trap consists of the following fields:

- Facility
- Hostname/IP Address

- Severity
- Message
- Timestamp

A SNMP v2 trap consists of list of varbinds.

5.2.5 Field Mapping

SNMP Field	Boundary Event Field	Boundary Field Type	Boundary Fingerprint Field?	Boundary Tag?
	createdAt	standard	NO	NO
trap message	message	standard	YES	NO
hostname	source.ref	standard	YES	YES
	source.type = `host`	standard	YES	YES
severity	severity(mapped)	standard	NO	NO
varbind	varbind name	property	NO	NO
	title (trapName +text + hostname)	standard	NO	NO

5.2.6 References

- Mauro, Douglas R.; Schmidt, Kevin J. *Essential SNMP*,2nd Ed. California: Sebastopol, OReily Media,Inc., 2009. Print.

5.2.7 Future Enhancements

- Generalized mapping and transformation of SNMP trap varbinds message fields to Boundary event fields
- Support for perform SNMP GETs.

6 Syslog

Boundary Syslog Route

The Syslog route nables the UDP receipt of forwarded syslog messagesg daemon into Boundary events.

This adapters adheres to standard set forth in RFC 3164.

6.1 Configuration

6.1.1 Forwarding Syslog Messages

The Syslog Event Route requires that syslog messages be forwarded:

- Using the `UDP` transport
- Sent in the RFC 3164 format.

There are various open source and commericial implementations of syslog including:

- `syslogd`
- Kiwi Syslog
- RSYSLOG
- syslog-ng

The syslog implementations liste above have the capability to forward syslog messages using UDP and syslog format (RFC 3164).

The configuration to forward syslog messages typically in this format:

```
*.*     @<hostname>:<port>
```

NOTE: Different implementations have different names for the configuration file location (typically `/etc/syslog.conf` or `/etc/rsylog.conf`), consult the documentation for your implementation to determine the path to the configuration file.

For example if the Boundary Event SDK was running on the host `ren.stimpy.com` and the Syslog adapter parameter `port` was configured to *1514* then the typical syslog configuration to forward *all* of the syslog messages would be the following:

```
*.*    @ren.stimpy.com:1514
```

If only a subset of the syslog messages are to be forwarded this possible by combined filtering using *facility* or *severity*. Examples below show only a subset of syslog messages:

Forward all syslog messages of severity of `info` to the Boundary Event SDK on the host *192.168.0.1* on port *1514*.

```
*.info @192.168.0.1:1514
```

Forward all `daemon` syslog messages to the Boundary Event SDK on the *localhost* on port *10514*:

```
daemon.*          @127.0.0.1:10514
```

6.1.2 Aggregating Syslog Messages

It is good practice to designate a system or systems to be serve the role as *loghost*. The loghost is responsible for aggregating the syslog messages sent from other hosts and then forwarding the required subset to the Syslog Message route which is co-located on the same host. Systems that are to forward Syslog messsage should be configured to forward messages using TCP rather than. Most modern implementations of Syslog allow forwarding of message via TCP. Consult your deployed Syslog implementation documentation for exact details and configuration on how to forward syslog messages via TCP.

6.1.3 Parameters

These parameters that application in the `etc/event-application.xml` file.

- `bindAddress` - Address to bind the socket to, defaults to 0.0.0.0
- `port` - Port number to listen for Syslog message (default is 1514).
- `routeId` - Name of the route instance appears in logs.
- `startOrder` - Ordering of when this route is started in relationship to other routes.
- `toUri` - Indicates the end point to send the transformed syslog message.

6.1.4 Example Configuration

```
<bean id="syslog-route" class="com.boundary.sdk.event.syslog.SysLogRouteBuilder">
   <property name="routeId" value="SYSLOG"/>
   <property name="startUpOrder" value="120"/>
   <property name="port" value="1514"/>
   <property name="toUri" value="seda:boundary-event"/>
</bean>
```

6.2 Event Mapping

This section describes the mapping of the Syslog message to a Boundary event.

A syslog message consists of the following fields:

- Facility
- Hostname/IP Address
- Severity
- Message
- Timestamp

6.2.1 Field Mapping

The table below provides a guide of how the Syslog message fields are mapped to a Boundary event.

Syslog Field	Boundary Event Field	Boundary Field Type	Boundary Fingerprint Field?	Boundary Tag?
facility	facility	property	YES	YES
timestamp	createdAt	standard	NO	NO
message	message	standard/property	YES	NO

hostname	source.ref	standard/property	YES	YES
	source.type	standard	YES	YES
severity	severity(mapped)	standard	NO	NO
severity	status(mapped)	standard	NO	NO
	title (text + hostname)	standard	NO	NO

6.2.2 Severity Mapping

Mapping of Syslog severity to Boundary event severity is given by the table below. Mapping can be customized by modification of a java property file (`BOUNDARY_SDK_HOME/syslog/etc/syslog.severity.properties`).

Syslog Severity	Boundary Event Severity
EMERG	CRITICAL
ALERT	CRITICAL
CRIT	CRITICAL
ERR	ERROR
WARNING	WARNING
NOTICE	INFO
INFO	INFO
DEBUG	INFO

6.2.3 Status Mapping

Boundary event status of a syslog message is determined by the Syslog severity. The table below shows the default mapping, which customized by modification of a java property file (`BOUNDARY_SDK_HOME/syslog/etc/syslog.status.properties`).

Syslog Severity	Boundary Event Status
EMERG	OPEN
ALERT	OPEN
CRIT	OPEN
ERR	OPEN
WARNING	OPEN
NOTICE	OK
INFO	OK
DEBUG	OK

6.3 Future Enhancements

- Generalized mapping and transformation of Syslog message fields to Boundary event fields

- Support for syslog format as specified by RFC 5424

7 Development

Boundary Event SDK Development

- Building

8 Building

Building

This page provides instructions on how to build the Boundary Event SDK.

8.1 Prerequisites

- Maven 3.2.1 or later
- Java JDK 1.7
- gcc 4.2.1 or later
- Git 1.7.1 or later

8.2 Instructions

8.2.1 Building the software

1. Clone the distribution

   ```
   $ git clone https://github.com/boundary/boundary-event-sdk
   ```
2. Change directory to the cloned repository

   ```
   $ cd boundary-event-sdk
   ```
3. Install additional components to build the distribution by running:

   ```
   $ bash setup.sh
   ```
4. Source the environment

   ```
   $ source env.sh
   ```
5. Build and package the distribution

   ```
   $ mvn assembly:assembly
   ```

8.2.2 Running the Boundary Event SDK in development

8.2.3 Building the documentation

1. Issue command to create the documentation:

   ```
   $ mvn site
   ```
2. Start up the site:

   ```
   $ mvn site:run
   ```
3. Documentation can be viewed from your local browser http://localhost:8080