[**Revision History**](#_8um1j0mfoc1u) **1**

[**Objectives**](#_2icsfajrxxgm) **1**

[**Requirements**](#_bxq1r2x5xp8n) **1**

[**Design**](#_bbng58hacbjd) **2**

[**Controller Action Configuration**](#_okjci0yqdngd) **2**

[**GET /mib usage**](#_aqt1jl8luysl) **3**

[**GET /action usage**](#_wf4d1qnr1k5v) **8**

[**POST /trap usage**](#_f4t9xz3bb605) **11**

# **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Author | Date | Summary of Changes |
| V1.0 | John Southerland | 9/1/2020 | Initial version |
| V1.1 | John Southerland | 9/10/2020 | Updated MIB to include an unknow event type OID, as well as text strings for both the event type and the sub event type |

**Table 1.1 – Revision History**

# Objectives

In order to support sending SNMP traps as an action from AppDynamics SaaS Controllers, we require a proxy running on the customer’s premise network that can fire traps in response to Policy Actions during a Health Rule Violation or Event on the SaaS Controller. To facilitate this, a proxy is needed that can take a request from the controller and send it on as an SNMP trap. The goal of this work effort is to produce just such a proxy, test it for functional capability, and assist the customer in deploying it on premise.

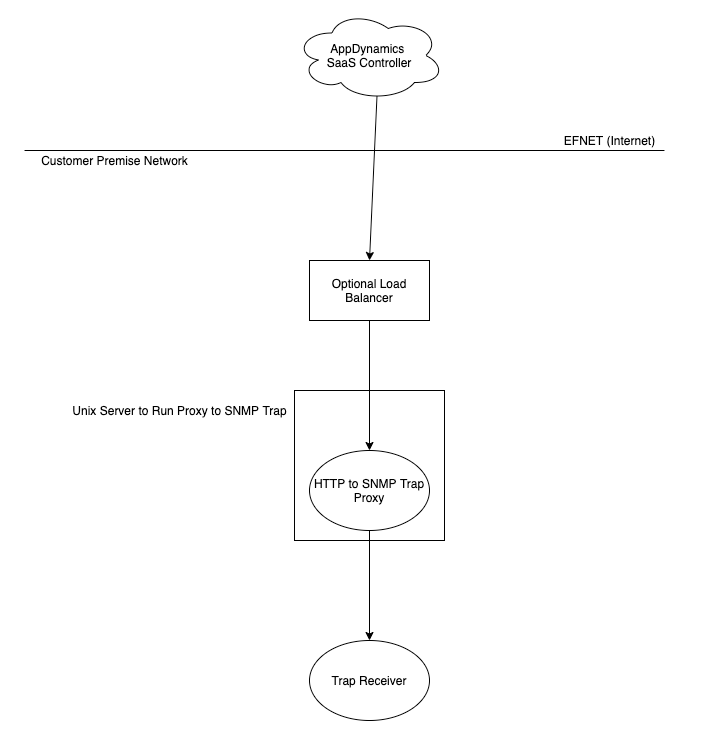
# Requirements

In order to ensure that the SNMP Trap forwarder works for the customer, it must conform to the following requirements:

1. It must be freestanding, and not require any custom hardware or software, where possible
2. It must support SSL encryption on the public facing web listener
3. It must support SNMP V2 traps, version 1 isn’t needed, but wouldn’t hurt as a fallback
4. It must support at least one userid and password for authentication
5. Must support sending to multiple receivers for high availability

# Design

In order to deliver a server quickly, we will make use of standard Java 1.8 JVM with an external library to assist in the SNMP trap generation. The high level design is very basic, a java program will listen on a port for HTTP requests and as requests are received, they will be parsed and forwarded as SNMP traps to the receiver configured in the request.



We will be dependent on the external library: snmp4j v 2.8.4 found here: <https://www.agentpp.com/>

# Controller Action Configuration

On the appdynamics saas controller, an action needs to be created making a restful call to this web application. The body of the post method is documented in the POST /trap usage.

# GET /mib usage

This rest call will print out the mib to be imported into the trap receiver. This is version 1:

AppDynamics-TRAP-MIB DEFINITIONS ::= BEGIN

IMPORTS

MODULE-IDENTITY, enterprises,

OBJECT-TYPE, NOTIFICATION-TYPE, *Integer32*,

*Counter64*

FROM SNMPv2-SMI

OBJECT-GROUP, NOTIFICATION-GROUP

FROM SNMPv2-CONF;

appdynamics MODULE-IDENTITY

LAST-UPDATED "2020091000000Z"

ORGANIZATION "AppDynamics Cisco"

CONTACT-INFO

"AppDynamics

Author: John.Southerland@appdynamics.com or josouthe@cisco.com

Last Ditch Email: it@appdynamics.com"

DESCRIPTION

"This MIB contains the definition of the SNMP Traps associated with Events and Health Rule Violations

sent by the AppDyanmics Controller -> REST Web Proxy -> a trap receiver.

This will need to be imported into your own trap receiver software in order to map the OIDs to values"

REVISION "2020091000000Z"

DESCRIPTION "First Revision, John Southerland john.southerland@appdynamics.com"

::= { enterprises 40684 }

events OBJECT IDENTIFIER ::= { appdynamics 138 }

eventsTable OBJECT-TYPE

SYNTAX SEQUENCE OF *eventEntry*

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION "table of events triggered from AppDynamics Controller"

::= { events 1 }

eventEntry OBJECT-TYPE

SYNTAX *eventEntry*

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION "data structure for an event object"

INDEX { eventId }

::= { eventsTable 1 }

*eventEntry* ::= SEQUENCE {

eventId *OCTET STRING*,

eventType *INTEGER*,

guid *OCTET STRING*,

eventTypeKey OCTET STRING,

eventTimeStamp *OCTET STRING*,

displayName *OCTET STRING*,

summaryMessage *OCTET STRING*,

eventMessage *OCTET STRING*,

applicationName OCTET STRING,

applicationId *INTEGER*,

tierName *OCTET STRING*,

tierId *INTEGER*,

nodeName *OCTET STRING*,

nodeId *INTEGER*,

databaseName *OCTET STRING*,

databaseId *INTEGER*,

severity *INTEGER*,

severityImageURL *OCTET STRING*,

accountName *OCTET STRING*,

policyName *OCTET STRING*,

actionName *OCTET STRING*,

controllerURL *OCTET STRING*,

deepLinkURL *OCTET STRING*,

notes *OCTET STRING*

}

eventId OBJECT-TYPE

SYNTAX OCTET STRING

MAX-ACCESS read-only

STATUS current

DESCRIPTION "ID of event in controller"

::= { eventEntry 1 }

eventType OBJECT-TYPE

SYNTAX *INTEGER* {

HealthRuleViolationEvent (1),

AnomalyViolationEvent (2),

SlowTransactionEvent (3),

CodeProblemEvent (4),

ApplicationChangeEvent (5),

ServerCrashEvent (6),

AppDynamicsConfigWarningEvent (7),

DiscoveryEvent (8),

SyntheticAvailabilityEvent(9),

SyntheticPerformanceEvent(10),

MibileCrashEvent(11),

ErrorEvent(12),

UnknownEvent(13)

}

MAX-ACCESS read-only

STATUS current

DESCRIPTION "The type of event to forward."

::= { eventEntry 2 }

guid OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "GUI of event in controller"

::= { eventEntry 3 }

eventTypeKey OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "event type description"

::= { eventEntry 4 }

eventTimeStamp OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "timestamp of event"

::= { eventEntry 5 }

displayName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "descriptive name of event"

::= { eventEntry 6 }

summaryMessage OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "summary text message of event"

::= { eventEntry 7 }

eventMessage OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "detailed message of event"

::= { eventEntry 8 }

applicationName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd application name"

::= { eventEntry 9 }

applicationId OBJECT-TYPE

SYNTAX *INTEGER*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd application id"

::= { eventEntry 10 }

tierName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd tier name"

::= { eventEntry 11 }

tierId OBJECT-TYPE

SYNTAX *INTEGER*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd tier id"

::= { eventEntry 12 }

nodeName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd node name"

::= { eventEntry 13 }

nodeId OBJECT-TYPE

SYNTAX *INTEGER*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd node id"

::= { eventEntry 14 }

databaseName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd database name"

::= { eventEntry 15 }

databaseId OBJECT-TYPE

SYNTAX *INTEGER*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd database id"

::= { eventEntry 16 }

severity OBJECT-TYPE

SYNTAX *INTEGER* {

INFO (1),

WARNING (2),

CRITICAL (3)

}

MAX-ACCESS read-only

STATUS current

DESCRIPTION "severity of event"

::= { eventEntry 17 }

severityImageURL OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "fun little image to go with severity"

::= { eventEntry 18 }

accountName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd account name"

::= { eventEntry 19 }

policyName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd policy name"

::= { eventEntry 20 }

actionName OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd action name"

::= { eventEntry 21 }

controllerURL OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd controller url for linking back to the UI"

::= { eventEntry 22 }

deepLinkURL OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "appd event url for linking back to the UI"

::= { eventEntry 23 }

notes OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "notes for this event"

::= { eventEntry 24 }

eventTypeString OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "String Text of Event Type"

::= { eventEntry 25 }

eventSubTypeString OBJECT-TYPE

SYNTAX *OCTET STRING*

MAX-ACCESS read-only

STATUS current

DESCRIPTION "Event Type has a subtype for many events, this is the raw text of that"

::= { eventEntry 26 }

EventNotification NOTIFICATION-TYPE

OBJECTS {

eventId,

eventType,

guid,

eventTypeKey,

eventTimeStamp,

displayName,

summaryMessage,

eventMessage,

applicationName,

applicationId,

tierName,

tierId,

nodeName,

nodeId,

databaseName,

databaseId,

severity,

severityImageURL,

accountName,

policyName,

actionName,

controllerURL,

deepLinkURL,

Notes,

eventTypeString,

eventSubTypeString

}

STATUS current

DESCRIPTION "alert notification, trap, for an event"

::= { events 2 }

END

# GET /action usage

The GET /action method will print the body needed in the action for the application policy action:

Receiver:"tcp:ipaddress/port"\n #Edit this to be accurate for your receiver

Version:"2"\n #Edit this to be accurate for your receiver

Community:"public"\n #Edit this to be accurate for your receiver

#if ($latestEvent.id)

eventId:${latestEvent.id}\n

#else

eventId:""\n

#end

#if ($latestEvent.eventType)

eventType:"${latestEvent.eventType}"\n

#else

eventType:""\n

#end

#if ($latestEvent.guid)

guid:"${latestEvent.guid}"\n

#else

guid:""\n

#end

#if ($latestEvent.eventTypeKey)

eventTypeKey:"${latestEvent.eventTypeKey.replace('"', "'")}"\n

#else

eventTypeKey:""\n

#end

#if ($latestEvent.eventTime)

eventTimeStamp:"${latestEvent.eventTime}"\n

#else

eventTimeStamp: ""\n

#end

#if ($latestEvent.displayName)

displayName:"${latestEvent.displayName.replace('"', "'")}"\n

#else

displayName:""\n

#end

#if ($latestEvent.summaryMessage)

summaryMessage:"${latestEvent.summaryMessage.replace('"', "'")}"\n

#else

summaryMessage:""\n

#end

#if ($latestEvent.eventMessage)

eventMessage:"${latestEvent.eventMessage.replace('"', "'")}"\n

#else

eventMessage:""\n

#end

#if ($latestEvent.application.name)

applicationName:"${latestEvent.application.name.replace('"', "'")}"\n

#else

applicationName:""\n

#end

#if ($latestEvent.application.id)

applicationId:${latestEvent.application.id}\n

#else

applicationId:""\n

#end

#if ($latestEvent.tier.name)

tierName:"${latestEvent.tier.name.replace('"', "'")}"\n

#else

tierName:""\n

#end

#if ($latestEvent.tier.id)

tierId:${latestEvent.tier.id}\n

#else

tierId:""\n

#end

#if ($latestEvent.node.name)

nodeName:"${latestEvent.node.name.replace('"', "'")}"\n

#else

nodeName:""\n

#end

#if ($latestEvent.node.id)

nodeId:${latestEvent.node.id}\n

#else

nodeId:""\n

#end

#if ($latestEvent.db.name)

databaseName:"${latestEvent.db.name.replace('"', "'")}"\n

#else

databaseName:""\n

#end

#if ($latestEvent.db.id)

databaseId:${latestEvent.db.id}\n

#else

databaseId:""\n

#end

#if ($topSeverity)

severity:"${topSeverity}"\n

#else

severity:""\n

#end

#if ($topSeverityImage)

severityImageURL:"${topSeverityImage.deepLink}"\n

#else

severityImageURL:""\n

#end

#if ($account.name)

accountName:"${account.name.replace('"', "'")}"\n

#else

accountName:""\n

#end

#if ($policy.name)

policyName:"${policy.name.replace('"', "'")}"\n

#else

policyName:""\n

#end

#if ($action.name)

actionName:"${action.name.replace('"', "'")}"\n

#else

actionName:""\n

#end

#if ($controllerUrl)

controllerUrl:"${controllerUrl}"\n

#else

controllerUrl:""\n

#end

#if ($latestEvent.deepLink)

deepLink:"${latestEvent.deepLink}"\n

#else

deepLink:""\n

#end

#if ($notes)

notes:"${notes.replace('"', "'")}"\n

#else

notes:""\n

#end

# POST /trap usage

The POST /trap method is the method that fires a trap using the action body above from the controller to the receiver.

TODO add a curl command that will issue a trap