Reference Guide

Ayla Rules API Specification

Version: 2.0

Date Released: November 13, 2018

Document Number: AY006URA3-2

Ayla Networks

Referenced Documents

Ayla Networks does not supply all documents that are referenced in this document with the equipment. Ayla Networks reserves the right to decide which documents are supplied with products and services.

Contact Information

Ayla Networks TECHNICAL SUPPORT and SALES

Contact Technical Support: http://help.aylasupport.com
or via email at support@aylanetworks.com

Contact Sales: https://www.aylanetworks.com/company/contact-us

Ayla Networks REGIONAL OFFICES

GREATER CHINA
Shenzhen
Room 310-311
City University of Hong Kong
Research Institute Building
No. 8 Yuexing 1st Road
High-Tech Industrial Park
Nanshan District
Shenzhen, China
Phone: 0755-86581520

HEADQUARTERS
Silicon Valley
4250 Burton Drive, Suite 100
Santa Clara, CA 95054
United States
Phone: +1 408 830 9844
Fax: +1 408 716 2621

JAPAN
Room #701,
No.2 Ueno Building 3-7-18,
Shin-Yokohama, Kohoku Ward
Yokohama City, 222-0033 Japan
Telephone: 045-594-8406

EUROPE
Munich
Building 64.07
Room EG.A.076 / 14b
Rupert-Mayer-Str. 44
D-81379 München
Germany

TAIWAN
Taipei
5F No. 250 Sec. 1
Neihu Road, Neihu District
Taipei 11493, Taiwan

For a Complete Contact List of Our Offices in the US, China, Europe, Taiwan, and Japan:
https://www.aylanetworks.com/company/contact-us

Table of Contents

1 Introduction

The Ayla Cloud Service (ACS) is a secure, distributed, scalable IoT platform, exposing a RESTful API for clients. The Ayla Device and Ayla User Services are the core REST resources that can be managed through the API.

The Ayla Rules Engine is a system of cloud components on the Ayla Platform that implement a Rules framework and enable devices to interact with the outside world.

1.1 About this Document

This document describes the Ayla Rules Engine APIs, including:

- Description of mandatory and optional input
- JSON examples along with their responses
- List of status codes that the API may return

1.2 Intended Audience

This document is intended for developers of web portals or mobile applications familiar with software using RESTful services over HTTPS. Users should be also familiar with the Ayla Cloud Service interaction with devices and users.

Requests to any of the Ayla Services (including ACS) require an access token. To obtain an access token, users need to log in to the Ayla User Service using the Sign-In API. The instructions are in the Sign In section of this document.

1.3 Related Documentation

You may want to refer to the following documents available on support.aylanetworks.com:

- *Ayla Customer Dashboard User's Guide*, AY006UDB3
- *Ayla API Browser Quick Reference Guide*, AY006FAP2
- *Ayla API Quick Start Guide,* AY006UAP2

1.4 API Browser

The Ayla API Browser provides developers with an interactive environment to learn and test Ayla APIs. The browser currently contains a subset of Ayla APIs. In the browser, you can view all pertinent details for the API and exercise the API to communicate directly with the Ayla system as you would from your own applications and devices. Each API request results in a Request URL, Response Body, Response Code, and Response Headers. The pertinent details provided in the browser include:

- Description of mandatory and optional input
- JSON examples along with their responses
- List of status codes that the API may return

To access the API Browser, log in to the Ayla Developer's Portal on https://developer.aylanetworks.com/, and click either the link or icon for the API Browser. If you have not created an account on the portal, refer to the *Ayla Developer Portal User's Guide,* AY006UDP3, on help.aylasupport.com.

NOTE Ayla Rules APIs will be available in the Ayla API Browser in late 2018.

2 Pre-Requisites

Prior to using the APIs, developers should obtain an application ID and application secret from the Apps section of their OEM profile in the OEM Dashboard. They should also know the device service and user service URLs for their region.

All APIs require an HTTP Authorization Header with the following the format:

```
Authorization: auth_token <token>
```

where `<token>` is obtained from the user object returned upon a successful sign-in from a developer site.

All URL parameters should be URL encoded.

3 Actions APIs

In addition to email, SMS, Push Notifications, actions can be creating a datapoint for a specific property on a specific device, notifying groups of people instead of a single individual, or even executing a separate set of actions defined elsewhere. Actions can exist on their own and later be associated to rules. This section provides Action APIs from the Ayla Rules Engine:

POST /rulesservice/v1/actions/

GET /rulesservice/v1/actions/

GET /rulesservice/v1/actions/:action_uuid

PUT /rulesservice/v1/actions/:action_uuid

DELETE /rulesservice/v1/actions/:action_uuid

3.1 POST /rulesservice/v1/actions/

This API creates a new action for the user.

Access Control: The results are filtered depending on the user. Actions can be created by a user with registered devices.

Pre-condition: The device must exist if the user is creating DATAPOINT action.

Post-condition: None

Related Tasks: N/A

Validation: Auth token-based authentication and authorization.

Parameters

Mandatory Input		
Parameter	**Description**	**Type**
name	The name that the user gave to the action.	string
type	The name that the user gave to the action type (for DATAPOINT, URL, or Diagnostic). Refer to GET/rulesservice/v1/action_type.	string
example	HTTP message body. Example: `{ "action" : { "name": "Send Datapoint", "type": "DATAPOINT", "parameters": { "datapoint": "DATAPOINT(TESTDSN_ARE030744_3537,boolean_input_tp) = true", } } }`	boolean

Mandatory Input (continued)		
Parameter	**Description**	**Type**
scheme	This supports two values: 1. Basic | basic IS supported. (Basic HTTP Authentication Scheme) 2. OAuthHelper, which is for OAuth authentication. Contact Ayla Technical Support for assistance with the OAuthHelper scheme. Example: for scheme OAuthHelper `{ "action" : { "name" : "Post to ", "user_uuid" : "e047f3f4-79b1-11e5-94b1-0ea626ed4811", "type" : "URL", "parameters" : { "endpoint" : "http://oauthservice:8090/oauthhelper/token?uuid=f6c62dce&idp=WeChat&clientId=wechat.application-oa2-client", "scheme" : "OauthHelper", "body_parameters": '{"foo":"bar"}' "body" : "Should be equal bar = {{foo}} " } } }`	string
username	This is the username required when the auth scheme is Basic.	string
password	This is the password required when the auth scheme is Basic.	string

Parameters	Contains key / value pairs with information relevant to the action to be performed.
	For Datapoint action, the required parameter key is `datapoint`.
	Example: for integer property
	`"parameters": { "datapoint" : "DATAPOINT(dsn1,prop1) = 70" }`
	Example: for string property
	`"parameters": { "datapoint": "DATAPOINT(VD72f801390000023, input_str) = 'string value'" }`
	Example: boolean property
	`"parameters": { "datapoint": "DATAPOINT(VD72f801390000023, Blue_LED) = true" }`
		Integer, string, or boolean as shown in examples

Mandatory Input (continued)		
Parameter	**Description**	**Type**

Parameters	For URL, the required parameter keys are `endpoint` and `body`.
	Example:
	`"parameters": {`
	`"body": "This is url action body",`
	`"endpoint": "https://webhook.site/3ef8a9ca-b4c2-4223-8a69-234bdc0308d8",`
	`"password": "s33cret",`
	`"scheme": "Basic",`
	`"username": "test_user11"`
	`},`
	`"parameters": {`
	`"body": "This is url action body",`
	`"endpoint": "https://webhook.site/3ef8a9ca-b4c2-4223-8a69-234bdc0308d8"`
	`},`
	For Diagnostic Action, the required parameter key is `diagnostic_state_name`.
	Example:
	`"parameters": {`
	`"diagnostc_state_name": "red"`
	`},`
		Integer, string, or boolean as shown in examples

Optional Input		
Parameter	**Description**	**Type**

permissions	This parameter key is valid for diagnostic actions only.
	NOTE: If permissions are not set for a diagnostic rule, only the user who created the rule has Create, Read, Update, and Delete (CRUD) access to the rule and the diagnostic states created by the rule.
	Example:
	`"permissions": [`
	`{`
	`"role": "OEM:Global IOT:Acme Telecom",`
	`"type": "UPDATE"`
	`},`
	`{`
	`"role": "OEM:Global IOT:Acme Retail",`
	`"type": "READ"`
	`}`
	`]`
		string

JSON Examples: Datapoint Action

- Action to update a property of type boolean

Curl JSON Example

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type:
application/json"  https://rulesservice.aylanetworks.com/rulesservice/v1/action
s -d '{
    "action": {
        "name": "String Action 6",
        "type": "DATAPOINT",
        "parameters": {
            "datapoint": "DATAPOINT(TESTDSN_3537,boolean_property_name) =
true"
        }
    }
}'
```

Response

```
{
    "action": {
        "name": "String Action 6",
        "action_uuid": "c265beb0-bf56-11e7-b118-c142a1baf62e",
        "user_uuid": "user-uuid-foo",
        "type": "DATAPOINT",
        "parameters": {
            "datapoint": "DATAPOINT(TESTDSN_3537,boolean_property_name) =
true"
        }
    }
}
```

- Action to update a property of type integer

Curl JSON Example

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type:
application/json"  https://rulesservice.aylanetworks.com/rulesservice/v1/action
s -d '{
    "action": {
        "name": "String Action 7",
        "type": "DATAPOINT",
        "parameters": {
            "datapoint": "DATAPOINT(TESTDSN_3537,integer_property_name) = 1"
        }
    }
}'
```

Response

```
{
    "action": {
        "name": "String Action 7",
        "action_uuid": "d265beb0-bf56-11e7-b118-c142a1baf62e",
        "user_uuid": "user-uuid-foo",
        "type": "DATAPOINT",
        "parameters": {
            "datapoint": "DATAPOINT(TESTDSN_3537,integer_property_name) = 1"
        }
    }
}
```

JSON Examples: Diagnostic Action

Curl JSON Example

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type:
application/json" https://rulesservice.aylanetworks.com/rulesservice/v1/actions
 -d '{
    "action": {
    "name": "actionName-foo",
    "type": "DIAGNOSTIC",
    "parameters": {
    "diagnostc_state_name": "red"
    },
    "permissions": [
    {
    "role": "OEM:Global IOT:Acme Telecom",
    "type": "UPDATE"
    },
    {
    "role": "OEM:Global IOT:Acme Retail",
    "type": "READ"
    }
    ]
    }
}'
```

Response


```
{
  "action" : {
"name" : "actionName-foo",
"action_uuid" : "529cb330-296f-11e8-8438-df49af59a923",
"user_uuid" : "51ec7601-296f-11e8-8438-df49af59a923",
"type" : "DIAGNOSTIC",
"parameters" : {
  "diagnostc_state_name" : "red"
},
"permissions" : [ {
  "role" : "OEM:Global IOT:Acme Telecom",
  "type" : "UPDATE",
  "created_at" : "2018-03-16T23:11:13.895Z",
  "created_by" : "51ec7601-296f-11e8-8438-df49af59a923"
}, {
  "role" : "OEM:Global IOT:Acme Retail",
  "type" : "READ",
  "created_at" : "2018-03-16T23:11:13.895Z",
  "created_by" : "51ec7601-296f-11e8-8438-df49af59a923"
} ],
"created_at" : "2018-03-16T23:11:13.895Z",
"updated_at" : "2018-03-16T23:11:13.895Z",
"created_by" : "51ec7601-296f-11e8-8438-df49af59a923",
"updated_by" : "51ec7601-296f-11e8-8438-df49af59a923"
  }
}
```


JSON Examples: URL Action

- Generic URL action

NOTE In the example below, the mandatory parameters are in **bold** text.

Curl JSON Example

```
curl -X POST 'https://rulesservice-perf.ayladev.com/rulesservice/v1/actions' -d
'{"action": {"name": "TestAction-656525-802657", "parameters": {"body":
"{\"foo\"=\"bar\"}", "endpoint": "https://webhook.site/3ef8a9ca-b4c2-4223-8a69-
234bdc0308d8", "password": "s33cret", "scheme": "Basic", "username":
"test_user11"}, "type": "URL"}}' -H 'Accept: application/json' -H 'Content-
Type: application/json' -H
'Authorization: auth_token 2663a84a05a2474cb3fb3c7ceaa6e2c7'
```

Response


```
{
  "action": {
    "action_uuid": "160a4ba1-604a-11e8-818a-7f8caf4c7d47",
    "created_at": "2018-05-25T18:33:14.843Z",
    "created_by": "11751ad4-604a-11e8-adcf-0aae5eab8cb0",
    "is_abstract": false,
    "name": "TestAction-656525-802657",
    "parameters": {
      "body": "\{"foo\"=\"bar\"},
      "endpoint": "https://webhook.site/3ef8a9ca-b4c2-4223-8a69-234bdc0308d8",
```

```
        "password": "s33cret",
        "scheme": "Basic",
        "username": "test_user11"
      },
      "type": "URL",
      "updated_at": "2018-05-25T18:33:14.843Z",
      "updated_by": "11751ad4-604a-11e8-adcf-0aae5eab8cb0",
      "user_uuid": "11751ad4-604a-11e8-adcf-0aae5eab8cb0"
    }
}
```

Response Status Code

201 - Success; see response above.

- OTA URL action

NOTE In the example below, the mandatory parameters are in **bold** text.

Curl JSON Example

```
curl -X POST 'https://rulesservice-perf.ayladev.com/rulesservice/v1/actions' -d
'{"action": {"name": "OTA action", "parameters": {"body":
"{{dpl_event}}", "endpoint": "https://otaservice.ayladev.com/foo?bar=thud"},
"type": "URL"}}' -H 'Accept: application/json' -H 'Content-Type:
application/json' -H
'Authorization: auth_token 2663a84a05a2474cb3fb3c7ceaa6e2c7'
```

Response

```
{
  "action": {
    "action_uuid": "160a4ba1-604a-11e8-818a-7f8caf4c7d47",
    "created_at": "2018-05-25T18:33:14.843Z",
    "created_by": "11751ad4-604a-11e8-adcf-0aae5eab8cb0",
    "is_abstract": false,
    "name": "TestAction-656525-802657",
    "parameters": {
      "body": "{{{dpl_event}}}",
     "endpoint": "https://otaservice.ayladev.com/foo?bar=thud"
    },
    "type": "URL",
    "updated_at": "2018-05-25T18:33:14.843Z",
    "updated_by": "11751ad4-604a-11e8-adcf-0aae5eab8cb0",
    "user_uuid": "11751ad4-604a-11e8-adcf-0aae5eab8cb0"
  }
}
```

If the above action is triggered, as a result, the Rule Action Service (RAS) calls the OTA endpoint https://otaservice.ayladev.com/foo with specified URL parameters (`"bar=thud"`) and a body containing the `DPL VERSION` event that triggered the actions, as shown below:

```
{
    "dpl_event": {
        "metadata": {
            "oem_id": "4f9bec12",
            "oem_model": "ledevb-test",
            "dsn": "TESTDSN_235978_000",
            "resource_tags": [],
            "event_type": "version"
        },
        "version_event": {
            "oem_model_version": "sanity-test-version",
            "ayla_model_version": "1.0",
            "ayla_model": "AY001MUS1"
        }
    }
}
```

Response Status Code

201 - Success; see response above.

Response Status Codes

201 – Success; see below:

```
{
   "action" : {
 "action_uuid": "1246881a-9526-11e5-986a-985aeb8b62f8",
   "name": "Property action",
  "type": "DATAPOINT",
  "parameters": {
    "datapoint": "DATAPOINT(TESTDSN_ARE030744_3537,boolean_input_tp) = true",
  },
  "created_at": "2017-05-19 11:42:50Z",
  "updated_at": "2017-05-19 11:42:50Z"
    }
}
```

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

422 – Failure: Validation failures; text indicating list of failures:

For example: Error message for datapoint action:

```
{"errors" : [{"code": "ARE-501",
  "message": "Invalid device xxxx."}]}
```

NOTE Response status codes in the 400-500 range are ARE error codes: `ARE-XXX`

3.2 GET /rulesservice/v1/actions/

This API gets a list of actions created by the user.

Access Control: The results are filtered depending on the user.

Pre-condition: None

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Pagination: Required

Parameters

None

JSON Example

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/actions
```

Response

```
{ "actions": [ { "action_uuid": "0f0881a-9526-11e5-986a-985aeb8b62f8", "name":
"Set Temperature", "type": "DATAPOINT", "parameters": {"datapoint" :
"DATAPOINT(dsn1,prop1) = 70"},  "created_at": "2017-05-19 11:42:50Z",
"updated_at": "2017-05-19 11:42:50Z" },  { "name" : "String Action 7",
"action_uuid" : "c265beb0-bf56-11e7-b118-c142a1baf62e", "user_uuid" : "user-
uuid-foo", "type" : "DATAPOINT", "parameters" : { "datapoint" :
"DATAPOINT(TESTDSN_3537,boolean_input_tp) = true" } } ] }
```

Response Status Code

200 – Success; see below:

```
{
  "actions": [
  {
   "action_uuid": "0f06881a-9526-11e5-986a-985aeb8b62f8",
    "name": "Set Temperature",
    "type": "DATAPOINT",
    "parameters": {
      "datapoint" : "DATAPOINT(dsn1,prop1) = 70"
    },
    "created_at": "2017-05-19 11:42:50Z",
    "updated_at": "2017-05-19 11:42:50Z"
  },
  {
 "action_uuid": "1246881a-9526-11e5-986a-985aeb8b62f8",
   "name": "Send Datapoint",
  "type": "DATAPOINT",
  "parameters": {
    "datapoint": "DATAPOINT(TESTDSN_ARE030744_3537,boolean_input_tp) = true",
  },
```

```
    "created_at": "2017-05-19 11:42:50Z",
    "updated_at": "2017-05-19 11:42:50Z"
     }
  ]
```

401 – Failure: credentials are required to access this resource

403 – Failure: User entitlement issues.

3.3 GET /rulesservice/v1/actions/:action_uuid

This API gets a single action using the ID created by the user.

Access Control: The results are filtered depending on the user.

Pre-condition: The action must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Parameters

Mandatory Input		
Parameter	**Description**	**Type**
id	This is the action_uuid.	string

JSON Example

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/actions/1246881a-9526-
11e5-986a-985aeb8b62f8
```

Response

```
{ "action" : { "name" : "String Action 7", "action_uuid" : "c265beb0-bf56-
11e7-b118-c142a1baf62e", "user_uuid" : "user-uuid-foo", "type" : "DATAPOINT",
"parameters" : { "datapoint" :"DATAPOINT(TESTDSN_3537,boolean_input_tp) =
true" } } }
```

Response Status Code

200 – Success; see below:

```
{{
   "action" : {
 "action_uuid": "1246881a-9526-11e5-986a-985aeb8b62f8",
   "name": "Datapoint action ",
  "type": "DATAPOINT",
  "parameters": {
    "datapoint": "DATAPOINT(TESTDSN_ARE030744_3537,boolean_input_tp) = true",
  },
```

```
    "created_at": "2017-05-19 11:42:50Z",
    "updated_at": "2017-05-19 11:42:50Z"
      }
}
```

401 – Failure: Credentials are required to access this resource

403 – Failure: Access forbidden

404 – Failure: Not found (for example, the action with ID does not exist.)

3.4 PUT /rulesservice/v1/actions/:action_uuid

This API updates a single action using the ID owned by the user.

Access Control: The results are filtered depending on user.

Pre-condition: The action must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Parameters

No parameters.

JSON Example

```
$ curl -X PUT -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/actions/1246881a-9526-
11e5-986a-985aeb8b62f8
```

Response

No response on success.

Response Status Code

401 – Failure: Credential are required to access this resource

403 – Failure: User entitlement issues

404 – Failure: Not found. Available in case the rule ID does not exist.

405 – Failure: Method not allowed

3.5 DELETE /rulesservice/v1/actions/:action_uuid

This API deletes a single action using ID owned by the user.

Access Control: The results are filtered depending on the user. End users can delete their own actions.

Pre-condition: The action must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Parameters

No parameters.

JSON Example

```
$ curl -X DELETE -H "Authorization: auth_token
6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/actions/1246881a-9526-
11e5-986a-985aeb8b62f8
```

Response

No response on success.

Response Status Code

204 - Success

401 - Failure: Credentials are required to access this resource.

403 - Failure: User entitlement issues

404 - Failure: Not found- Available in case the rule ID does not exist.

409 - Failure: Conflict- This occurs if it cannot delete the action because it is used in one or more rule. The error message will contain list of associated rules.

4 Rules APIs

In the Ayla Rules Engine, actions can exist without any association with a rule; however, rules can only be created when associated with one or more actions. This section provides Rules APIs in the Ayla Rules Engine:

4.1 POST /rulesservice/v1/rules/

This API creates a new rule for the user.

Access Control: The results are filtered depending on user. Rules can be created by users with registered devices.

Pre-condition: The device and actions used in the rule must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
name	The name for this rule entered by the user.	string
expression	For syntax, see the section on Ayla Rule Expression Syntax (ARES).	
action_ids	The list of actions associated with a rule.	
Optional Input		

Parameter	Description	Type
description	The description of the rule entered by the user.	string
Example (showing above parameters): `{ "rule" : { "name": "My Rule 1", "description": "Description 1", "expression": "DATAPOINT(dsn1,prop1) < 60 AND DP(dsn2, prop2) == 8", "action_ids": [ "1246881a-9526-11e5-986a-985aeb8b62f8" ] } }`		
is_template	If true, expression must be abstract, for example, defined on a group of devices. NOTE: The `is_template` parameter key is mandatory if the rule is a template.	boolean

JSON Examples

- Generic

Curl JSON Example

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type: application/json" -d '{"rule" : {"name": "My Rule 1",
"description": "Description 1",  "expression": "DATAPOINT(dsn1,prop1) < 60",
"action_ids": [124]}}'
https://rulesservice.aylanetworks.com/rulesservice/v1/rules

{"rule" :{"rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8",,  "name": "My
Rule 1",
  "is_enabled": true, "description": "Description 1", "expression":
"DATAPOINT(dsn1,prop1) < 60", "action_ids": [124],  "created_at": "2017-05-19
11:42:50Z", "updated_at": "2017-05-19 11:42:50Z"}}
```

NOTE For syntax and examples of rules expression, see Ayla Rule Expression Syntax (ARES).

Response

```
{
  "rule" : {
   "rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8",
  "name": "My Rule 1",
  "is_enabled": true,
  "description": "Description 1",
  "expression": "DATAPOINT(dsn1,prop1) < 60",
  "action_ids": [
    "1246881a-9526-11e5-986a-985aeb8b62f8"
  ],
  "created_at": "2017-05-19 11:42:50Z",
  "updated_at": "2017-05-19 11:42:50Z"
  }
}
```

- Add diagnostic rule

NOTE Diagnostic rules use abstract datapoint expression syntax (shown in bold text in the following example/response).

Curl JSON Example: Diagnostic Rule

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type:
application/json" https://rulesservice.aylanetworks.com/rulesservice/v1/rules -
d '{
    "rule": {
    "name": "name-foo",
 "description" : "descr-fred",
    "expression": "DP( ${oem_model=oemModel-
bar,template_version=templateVersion-quux},prop-name-baz) >= 21",
    "action_ids": [
    "dfa3b790-27e6-11e8-91e7-cd99fea1c2a5"
    ],
    "is_enabled": true
    }
}'
```

Response: Created Diagnostic Rule

```
{
  "rule" : {
"name" : "foo",
"uuid" : "6ca9c090-298a-11e8-b245-d30547e2ae18",
"description" : "descr-fred",
"expression" : "DP ( ${oem_model=oemModel-bar,template_version=templateVersion-
quux},prop-name-baz) >= 21",
"is_enabled" : true,
"is_abstract" : true,
"action_ids" : [ "dfa3b790-27e6-11e8-91e7-cd99fea1c2a5" ],
"user_uuid" : "6c121471-298a-11e8-b245-d30547e2ae18",
"created_at" : "2018-03-17T02:25:14.096Z",
"updated_at" : "2018-03-17T02:25:14.096Z",
"created_by" : "6c121471-298a-11e8-b245-d30547e2ae18",
"updated_by" : "6c121471-298a-11e8-b245-d30547e2ae18"
  }
}
```

- Add OTA rule

NOTE OTA rules use abstract VERSION expression syntax (shown in bold text in the following example/response).

Curl JSON Example: OTA (also called VERSION) Rule

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type:
application/json" https://rulesservice.aylanetworks.com/rulesservice/v1/rules -
d '{
  "rule" : {
```

```
    "name" : "name-qux",
    "description" : "description-baz",
    "expression" : "( VERSION(${oem_model <= oemModel-
foo, template_version < templateVersion-bar}) ) && (
VERSION(${oem_model <= oemModel-foo, template_version < templateVersion-bar})
)",
"action_ids" : [ "af204d00-8406-11e8-bf2d-fbf04044dd8b" ]
  }
}'
```

Response: Created OTA Rule

```
{
        "rule": {
                "rule_uuid": "af5d5600-8406-11e8-bf2d-fbf04044dd8b",
                "name": "name-qux",
                "description": "description-baz",
                "expression": "( VERSION(${oem_model <= oemModel-
foo, template_version < templateVersion-bar}) ) && (
VERSION(${oem_model <= oemModel-foo, template_version < templateVersion-bar})
)",
                "is_enabled": true,
                "is_abstract": true,
                "is_template": false,
                "action_ids": [
                        "af204d00-8406-11e8-bf2d-fbf04044dd8b"
                ],
                "user_uuid": "aeee8ea1-8406-11e8-bf2d-fbf04044dd8b",
                "created_at": "2018-07-10T06:01:28.238Z",
                "updated_at": "2018-07-10T06:01:28.238Z",
                "created_by": "aeee8ea1-8406-11e8-bf2d-fbf04044dd8b",
                "updated_by": "aeee8ea1-8406-11e8-bf2d-fbf04044dd8b"
}
}
```

- Add rule based on CONNECTION event

Curl JSON Example: CONNECTION Event Based Rule

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type:
application/json" https://rulesservice.aylanetworks.com/rulesservice/v1/rules -
d '{
        "rule": {
                "name": "name-foo",
                "expression": "( CONNECTION(DSN0000000120,online) ||
CONNECTION(DSN0000000120,offline) || CONNECTION(DSN0000000120,all) )",
                "action_ids": [
                        "4312e4d0-8472-11e8-b3c4-29146421699e"
                ]
        }
}'
```

Response: Created CONNECTION Event Based Rule


```
{
        "rule": {
                "rule_uuid": "441a90d0-8472-11e8-b3c4-29146421699e",
                "name": "name-foo",
                "expression": "( CONNECTION(DSN0000000120,online) ||
CONNECTION(DSN0000000120,offline) || CONNECTION(DSN0000000120,all) )",
                "is_enabled": true,
                "is_abstract": false,
                "is_template": false,
                "action_ids": [
                        "4312e4d0-8472-11e8-b3c4-29146421699e"
                ],
                "user_uuid": "42ce13a1-8472-11e8-b3c4-29146421699e",
                "created_at": "2018-07-10T18:51:34.002Z",
                "updated_at": "2018-07-10T18:51:34.002Z",
                "created_by": "42ce13a1-8472-11e8-b3c4-29146421699e",
                "updated_by": "42ce13a1-8472-11e8-b3c4-29146421699e"
        }
}
```


Response Status Code

201 – Success (see Responses for JSON examples)

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

422 – Failure Validation failures (test indicating list of failures)

Error message for invalid params:


```
{"errors" : [{"code": "ARE-501",
  "message": "Invalid parameter xxxx."}]}
```

4.2 POST /rulesservice/v1/rules/templates/:template_rule_uuid

This API instantiates a new rule for the user based on the rule template.

Access Control: The results are filtered depending on the user. Rules can be created by users with registered devices.

Pre-condition: The device must be mentioned in the rule using the DSN and the template rule.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
name	The name for this rule entered by the user.	string
expression	See the section on Ayla Rule Expression Syntax (ARES).	
Optional Input		
Parameter	**Description**	**Type**
description	The description of the rule as entered by the user.	string
Mandatory Input example if the rule is a template:		

<pre>{
"rule" : {
"name": "My Rule 1",
"description": "Description 1",
"expression":
"DATAPOINT(dsn1,prop1) < 60 AND
DP(dsn2, prop2) == 8"
}
}</pre>
	If true, the expression must be abstract, such as when defined on a group of devices. See the curl JSON example below for the HTTPS body.
	
	NOTE: The `action_ids` parameter is not required when rule is instantiated from the rule template. The actions for the rule instance will be inherited from the rule template being instantiated.	

JSON Example

```
$ curl -X POST -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type: application/json" -d '{"rule" : {"name": "My Concrete Rule
2",  "description": "Description 1",  "expression": "DATAPOINT(dsn1,prop1) <
60",  "action_ids": [124]}}'
```

https://rulesservice.aylanetworks.com/rulesservice/v1/rules/templates/852cd586-5ee0-11e8-9c2d-fa7ae01bbebc

Response


```
{“rule” :{"rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8",
"template_uuid": "852cd586-5ee0-11e8-9c2d-fa7ae01bbebc",  "name": "My Concrete
Rule 2",
  "is_enabled": true, "is_template": false, "description": "Description 1",
"expression": "DATAPOINT(dsn1,prop1) < 60", "action_ids": [124],
"created_at": "2017-05-19 11:42:50Z", "updated_at": "2017-05-19 11:42:50Z"}}
```


Response Status Code

201 – Success (see Response)

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

422 – Failure Validation failures (test indicating list of failures)

Error message for invalid params:


```
{"errors" : [{"code": "ARE-501",
  "message": "Invalid parameter xxxx."}]}
```


4.3 GET /rulesservice/v1/rules/templates or GET /rulesservice/v1/rules/

This API gets the list of rules (owned by the user) that pertain to templates.

Access Control: The results are filtered depending on user. OEM Admin can access this API.

Pre-condition: The DSN and user_uuid must exist for the OEM user calling this API.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Pagination: Required

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
type	Valid values are "device" or "user."	string
id	This is a valid device serial number (DSN) or user_uuid.	string

Optional Input		
Parameter	**Description**	**Type**
oemid	This is a valid OEM ID. NOTE: This parameter is mandatory if the user is Ayla Admin.	string

JSON Example

End User:

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/rules


{ "rules": [ { "rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8", "name":
"My Rule 1", "is_enabled": true, "description": "Description 1", "expression":
"DATAPOINT(dsn1,prop1) < 60", "action_ids": [ 124 ],  "created_at": "2017-05-
19 11:42:50Z", "updated_at": "2017-05-19 11:42:50Z" }, { "rule_uuid":
"1543281a-9526-11e5-986a-985aeb8b62f8",, "name": "My Rule 2", ++--------------
----"is_enabled": true, "description": "Description 2", "expression":
"DATAPOINT(dsn12,prop10) == 1)", "action_ids": [ 1567 ],  "created_at": "2017-
05-19 11:42:50Z", "updated_at": "2017-05-19 11:42:50Z" } ] }
```

OEM User:

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/rules?type=device&id=234
dw21a-9526-11e5-986a-985aeb8


{ "rules": [ { "rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8", "name":
"My Rule 1", "is_enabled": true, "description": "Description 1", "expression":
"DATAPOINT(dsn1,prop1) < 60", "action_ids": [ 124 ],  "created_at": "2017-05-
19 11:42:50Z", "updated_at": "2017-05-19 11:42:50Z" }, { "rule_uuid":
"1543281a-9526-11e5-986a-985aeb8b62f8",, "name": "My Rule 2", "is_enabled":
true, "description": "Description 2", "expression": "DATAPOINT(dsn12,prop10)
== 1)", "action_ids": [ 1567 ],  "created_at": "2017-05-19 11:42:50Z",
"updated_at": "2017-05-19 11:42:50Z" } ] }
```

Response

```
{
    "rules": [
     {
  "rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8",
  "name": "My Rule 1",
  "is_enabled": true,
  "description": "Description 1",
  "expression": "DATAPOINT(dsn1,prop1) < 60 && DATAPOINT(dsn1,prop1) <= -60
",
  "action_ids": [
    "1246881a-9526-11e5-986a-985aeb8b62f8"
  ],
   "created_at": "2017-05-19 11:42:50Z",
  "updated_at": "2017-05-19 11:42:50Z"
```

```
}
},
{
  "rule_uuid": "1543281a-9526-11e5-986a-985aeb8b62f8",
  "name": "My Rule 2",
  "is_enabled": true,
  "description": "Description 2",
  "expression": "DATAPOINT(dsn12,prop10) == 1)",
  "action_ids": [
     "0f06881a-9526-11e5-986a-985aeb8b62f8"
  ],
  "created_at": "2017-05-19 11:42:50Z",
  "updated_at": "2017-05-19 11:42:50Z"
}
  ]
}
```


Response Status Code

200 – Success (see Response)

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

4.4 GET /rulesservice/v1/oems/:str_oem_id/rules

This API gets a paginated list of rules set by the OEM.

Access Control: The results are filtered depending on user.

Pre-condition: None.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Pagination: Required

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
:str_oem_id	This requires a valid OEM ID.	string
Optional Input		
Parameter	**Description**	**Type**
abstract_only	Abstract rules are OEM level rules used mostly for a fleet of devices.	boolean
concrete_only	Concrete rules are used for a single device.	boolean

JSON Example

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/oems/oem-
fred/rules.json?abstract_only=false&paginated=true&per_page=2&page=1&order_by=
asc

{
  "total" : 3,
  "current_page_number" : 1,
  "next_page" : 2,
  "start_count_on_page" : 0,
  "end_count_on_page" : 1,
  "rules" : [ {
   "name" : "name-foo",
   "uuid" : "41821a30-2c63-11e8-bdd8-6bed45baa012",
   "expression" : "DATAPOINT(foo-dsn,bar-name) >= 10",
   "is_enabled" : true,
   "is_abstract" : false,
   "action_ids" : [ "416bfa20-2c63-11e8-bdd8-6bed45baa012", "41737430-2c63-
11e8-bdd8-6bed45baa012" ],
   "user_uuid" : "40bf1851-2c63-11e8-bdd8-6bed45baa012",
   "created_at" : "2018-03-20T17:22:24.873Z",
   "updated_at" : "2018-03-20T17:22:24.873Z",
   "created_by" : "40bf1851-2c63-11e8-bdd8-6bed45baa012",
   "updated_by" : "40bf1851-2c63-11e8-bdd8-6bed45baa012"
  }, {
   "name" : "name-bar",
   "uuid" : "41a18910-2c63-11e8-bdd8-6bed45baa012",
   "expression" : "DATAPOINT(bar-dsn,bar-name) >= 10",
   "is_enabled" : true,
   "is_abstract" : false,
   "action_ids" : [ "416bfa20-2c63-11e8-bdd8-6bed45baa012", "41737430-2c63-
11e8-bdd8-6bed45baa012" ],
   "user_uuid" : "40bf1851-2c63-11e8-bdd8-6bed45baa012",
   "created_at" : "2018-03-20T17:22:24.960Z",
   "updated_at" : "2018-03-20T17:22:24.960Z",
   "created_by" : "40bf1851-2c63-11e8-bdd8-6bed45baa012",
   "updated_by" : "40bf1851-2c63-11e8-bdd8-6bed45baa012"
  } ]
}
```

Response

```
{
  "total" : 1,
  "current_page_number" : 1,
  "next_page" : 2,
  "start_count_on_page" : 0,
  "end_count_on_page" : 0,
  "rules" : [ {
   "name" : "name-foo",
   "uuid" : "bd2dad50-2c60-11e8-962f-05d5a3a29041",
   "expression" : "DATAPOINT(TESTDSN001,bar-name) >= 10",
   "is_enabled" : true,
   "is_abstract" : false,
   "action_ids" : [ "bd1ff1b0-2c60-11e8-962f-05d5a3a29041", "bd256ff0-2c60-
```

```
11e8-962f-05d5a3a29041" ],
    "user_uuid" : "bd162db0-2c60-11e8-962f-05d5a3a29041",
    "created_at" : "2018-03-20T17:04:23.749Z",
    "updated_at" : "2018-03-20T17:04:23.749Z",
    "created_by" : "bd162db0-2c60-11e8-962f-05d5a3a29041",
    "updated_by" : "bd162db0-2c60-11e8-962f-05d5a3a29041"
  } ]
}
```


Response Status Code

200 – Success (see Response)

401 – Failure: Authentication required.

403 – Failure: Access forbidden.

4.5 GET /rulesservice/v1/rules/:rule_uuid

This API gets a single rule owned by the user.

Access Control: The results are filtered depending on user

Pre-condition: The rule must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
id	The ID of the rule.	string

JSON Example

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/rules/1611281a-9526-
11e5-986a-985aeb8b62f8
```


```
{"rule" : {"rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8", "name": "My
Rule 1", "is_enabled": true, "description": "Description 1", "expression":
"DATAPOINT(dsn1,prop1) < 60", "action_ids": [124], " "created_at": "2017-05-19
11:42:50Z", "updated_at": "2017-05-19 11:42:50Z"}}
```

Response


```
{
  "rule" : {
   "rule_uuid": "1611281a-9526-11e5-986a-985aeb8b62f8",
  "name": "My Rule 1",
  "is_enabled": false,
  "description": "Description 1",
  "expression": "DATAPOINT(dsn1,prop1) < 60",
  "action_ids": [
    "1246881a-9526-11e5-986a-985aeb8b62f8"
  ],
  "created_at": "2017-05-19 11:42:50Z",
  "updated_at": "2017-05-19 11:42:50Z"
  }
}
```


Response Status Code

200- Success (see Response)

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

404 – Failure: Not found. Rule ID does not exist.

4.6 GET /rulesservice/v1/rules/:rule_uuid/actions

This API gets all actions associated with a single rule using the ID owned by the user.

Access Control: The results are filtered depending on the user. OEM admin can access this API.

Pre-condition: The rule must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Pagination: Required

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
rule_uuid	Universal Unique Identifier (UUID) of the rule	string
Optional Input		
Parameter	**Description**	**Type**
oemid	Valid OEM ID.	string

JSON Example

End User and OEM Admin:

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/rules/1611281a-9526-
11e5-986a-985aeb8b62f8/actions


{ "actions": [ { ”action_uuid”: “0f06881a-9526-11e5-986a-985aeb8b62f8”,
"name": "Set Temperature", "type": "DATAPOINT", "parameters": {"datapoint" :
"DATAPOINT(dsn1,prop1) = 70"},  "created_at": "2017-05-19 11:42:50Z",
"updated_at": "2017-05-19 11:42:50Z" }] }
```

Response

```
{
  "actions": [
  {
     ”action_uuid”: “0f06881a-9526-11e5-986a-985aeb8b62f8”
    "name": "Set Temperature",
    "type": "DATAPOINT",
    "parameters": {
      "datapoint" : "DATAPOINT(dsn1,prop1) = 70"
    },
    "created_at": "2017-05-19 11:42:50Z",
    "updated_at": "2017-05-19 11:42:50Z"
  }
 ]
}
```

Response Status Code

200 - Success (see Response)

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

404 – Failure: Not found. Available in case the rule ID does not exist.

4.7 PUT /rulesservice/v1/rules/:rule_uuid

This API enables or disables a single rule using the UUID of the rule.

Access Control: The end user who owns the rule and the OEM admin have access to this method.

Pre-condition: The rule must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
rule_uuid	Universal Unique Identifier (UUID) of the rule	string
status	This is the flag to enable or disable the rule.	string

JSON Example

End User and OEM Admin:

```
$ curl -X PUT -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type: application/json" -d '{ "attributes" : {    "is_enabled" :
false  }}'
https://rulesservice.aylanetworks.com/rulesservice/v1/rules/1611281a-9526-
11e5-986a-985aeb8b62f8
```

```
{
  "rule_uuid" : "611281a-9526-11e5-986a-985aeb8b62f8",
  "name" : "test rule",
  "expression" : "DATAPOINT ( fred-dsn,bar-name ) >= 10",
  "is_enabled" : false,
  "is_abstract" : false,
  "action_ids" : [ "9b95ea10-432e-11e8-8b05-0daee2cbcc31", "9b9d6420-432e-
11e8-8b05-0daee2cbcc31" ],
  "user_uuid" : "9a973ec1-432e-11e8-8b05-0daee2cbcc31",
  "created_at" : "2018-04-18T17:33:29.517Z",
  "updated_at" : "2018-04-18T17:33:29.577Z",
  "created_by" : "9a973ec1-432e-11e8-8b05-0daee2cbcc31",
  "updated_by" : "9a973ec1-432e-11e8-8b05-0daee2cbcc31",
}
```

Response

```
{
  "attributes" : {
    "is_enabled" : false
  }
}
```

Response Status Code

200 - Success (see Response)

401 – Failure: Credentials are required to access this resource.

403 – Failure: User is not permitted to do the update.l

404 – Failure: Not found. Rule ID does not exist.

4.8 DELETE /rulesservice/v1/rules/:rule_uuid

This API deletes a single rule using the ID owned by the user.

Access Control: Results are filtered depending on user.

Pre-condition: Rule must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
id	The ID of the rule that is going to be deleted.	string

JSON Example

```
$ curl -X DELETE -H "Authorization: auth_token
6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/rules/1611281a-9526-
11e5-986a-985aeb8b62f8
```

Response

No response is returned on a successful deletion.

Response Status Code

204 - Success (no content returned)

401 – Failure: Credentials are required to access this resource

403 – Failure: User entitlement issues

404 – Failure: Not found. Available in case the rule ID does not exist.

4.9 GET /rulesservice/v1/rules/logs/

This API gets rule logs (paginated) for a given rule/action/user/oem/event.

Access Control: Results are filtered depending on the requesting user and filter criteria. The OEM Admin can access this API.

Pre-condition: Given resource ID should be valid.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization

Pagination: Optional

Parameters

Mandatory Input:		
Parameter	**Description**	**Type**
id	Valid ID of the type specified	string
type	Valid values are user, rule, action, oem, event.	string
Optional Input		
Parameter	**Description**	**Type**
paginated	True = paginate the results False = do not paginate results (default = false)	boolean
page	Any integer number (default = 1)	integer
perpage	Any integer number (default = 50).	integer

JSON Example

End User:

```
curl -X GET 'https://staging-
rulesservice.ayladev.com/rulesservice/v1/rules/logs?type=rule&id=e96fa4d5-
2c57-4688-be34-bdceb5b7b2e6'  -H 'authorization: auth_token
7fd0088ab8bf45f899d46557304cbc8f'
```


```
{ "next_page": 1, "start_count_on_page": 1, "total": 1, "end_count_on_page":
1, "current_page_number": 1, rule_logs": [ { "rulesStagePerformedBy":
"3aac9958-a48a-11e7-83a2-0eb9827efe40", "oem_uuid": "72f80139", "user_uuid":
"3aac9958-a48a-11e7-83a2-0eb9827efe40", "resource_uuid": "e96fa4d5-2c57-4688-
be34-bdceb5b7b2e6", "resource_name": "Test Rule with datapoint actions",
"expression": "DATAPOINT(VD72f801390000023,rule_prop) == 75", "resource_type":
"rule", "rules_stage": "create", "rules_stage_performed_at": "2017-10-
06T21:49:21.000Z", "created_at": "2017-10-06T21:49:21.000Z" } ] }
```


OEM User:

```
curl -X GET 'https://staging-
rulesservice.ayladev.com/rulesservice/v1/rules/logs?type=user&id=3aac9958-
a48a-11e7-83a2-0eb9827efe40'  -H 'authorization: auth_token
7fd0088ab8bf45f899d435547304cbc8f'
```


```
{ "next_page": 1, "start_count_on_page": 1, "total": 2, "end_count_on_page":
2, "current_page_number": 1, rule_logs": [ { "rulesStagePerformedBy":
"3aac9958-a48a-11e7-83a2-0eb9827efe40", "oem_uuid": "72f80139", "user_uuid":
"3aac9958-a48a-11e7-83a2-0eb9827efe40", "resource_uuid": "e96fa4d5-2c57-4688-
be34-bdceb5b7b2e6", "resource_name": "Test Rule with datapoint actions",
```

```
"expression": "DATAPOINT(VD72f801390000023,rule_prop) == 75", "resource_type":
"rule", "rules_stage": "create", "rules_stage_performed_at": "2017-10-
06T21:49:21.000Z", "created_at": "2017-10-06T21:49:21.000Z" }, {
"rulesStagePerformedBy": "3aac9958-a48a-11e7-83a2-0eb9827efe40", "oem_uuid":
"72f80139", "user_uuid": "3aac9958-a48a-11e7-83a2-0eb9827efe40",
"resource_uuid": "fet3435e-2c57-4688-be34-bdceb5b7b2e6", "resource_name": "New
Rule", "expression": "DATAPOINT(DSN12143543435,boolean_property) == true",
"resource_type": "rule", "rules_stage": "create", "rules_stage_performed_at":
"2017-10-06T21:49:21.000Z", "created_at": "2017-10-06T21:49:21.000Z" } ] }
```


Response


```
{
  "next_page": 1,
  "start_count_on_page": 1,
  "total": 1,
  "end_count_on_page": 1,
  "current_page_number": 1,
  "rule_logs": [
    {
      "rulesStagePerformedBy": "3aac9958-a48a-11e7-83a2-0eb9827efe40",
      "oem_uuid": "72f80139",
      "user_uuid": "3aac9958-a48a-11e7-83a2-0eb9827efe40",
      "resource_uuid": "e96fa4d5-2c57-4688-be34-bdceb5b7b2e6",
      "resource_name": "Test Rule with datapoint actions",
      "expression": "DATAPOINT(VD72f801390000023,rule_prop) == 75",
      "resource_type": "rule",
      "rules_stage": "create",
      "rules_stage_performed_at": "2017-10-06T21:49:21.000Z",
      "created_at": "2017-10-06T21:49:21.000Z"
    }
  ]
}
```


Response Status Code

200 - Success (see Response)

400 – Failure: Bad request

401 – Failure: Credentials are required to access this resource

4.10 GET /rulesservice/v1/rules/devices/

This API gets a list of devices (DSNs) active in a rule.

Access Control: The results are filtered depending on the OEM. OEM Admin can access this API.

Pre-condition: The OEM must exist.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Pagination: Optional

Parameters

Optional Input		
Parameter	**Description**	**Type**
paginated	The default value is false. When "true," the results are paginated. When "false," the results are not paginated.	boolean
page	This can be any integer number. The default value is 1.	integer
per_page	This can be any integer number. The default value is 50.	integer
oemid	This must be a valid OEM ID.	string

JSON Example

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
https://rulesservice.aylanetworks.com/rulesservice/v1/rules/devices?paginated=
true&per_page=20&page=1

{ "next_page": "2", "current_page_number": "1", "start_count_on_page": "1",
"end_count_on_page": 2, "total": 2, "devices" : [ 'dsn1', 'dsn2'] }
```

Response

```
{
   "next_page": 2,
   "current_page_number": 1,
   "start_count_on_page": 1,
   "end_count_on_page": 2,
   "total": 2,
"devices" : ['dsn1', 'dsn2']
}
```

Response Status Code

200 - Success (see Response)

400 – Failure: Bad request

401 – Failure: Credentials are required to access this resource

5 Diagnostic States

One of the features of the Ayla Rules Engine is the Diagnostic Service, which is specifically for diagnostic monitoring. The Diagnostic Service is designed to provide, on a large scale and in real time, the diagnostic state of devices (such as being in state of low Wi-Fi signal strength, high usage, or over-heated). The conditions that define diagnostic states are stored in the database of the Ayla Rules Engine as diagnostic rules. This section includes the API for diagnostic states.

5.1 GET /rulesservice/v1/diagnostic_states

This API gets the diagnostic states created either by a rule with a specific ID, by an action with a specific ID, or for a specific DSN.

Access Control: The results are filtered depending on the user and the filter criteria.

Pre-condition: None.

Post-condition: None

Related Tasks: N/A

Validation: Auth token based authentication and authorization.

Pagination: Optional

Parameters

Optional Input		
Parameter	**Description**	**Type**
rule_uuid	Universal Unique Identifier (UUID) of the rule. The default value is null.	string
action_uuid	The default value is null.	string
current_only	The default value is false.	boolean
paginated	The default value is false.	boolean
page	This can be any integer number. The default value is 1.	integer
per_page	This can be any integer number. The default value is 50.	integer

JSON Example

```
curl -X GET -H "Content-Type: application/json" -H 'authorization: auth_token
7fd0088ab8bf45f899d46557304cbc8f' 'https://staging-
rulesservice.ayladev.com/rulesservice/v1/diagnostic_states?dsn=TESTDSN_ARE030744_3537'
```

```
{
  "total" : 2,
  "current_page_number" : 1,
  "next_page" : 1,
  "start_count_on_page" : 0,
  "end_count_on_page" : 1,
  "diagnostic_states" : [ {
   "uuid" : "15816d92-2709-11e8-a7f8-2903a354ded2",
   "dsn" : "TESTDSN_ARE030744_3537",
   "user_uuid" : "16816d91-2709-11e8-a7f8-2903a354ded2",
   "rule_uuid" : "e96fa4d5-2c57-4688-be34-bdceb5b7b2e6",
   "rule_name" : "device in low signal state"",
   "oem_id" : "oemid_123",
   "oem_model" : "oem_model_123",
   "name" : "RED",
   "created_at" : "2018-03-13T21:54:22.056Z"
  }, {
   "uuid" : "16816d95-2709-11e8-a7f8-2903a354ded2",
   "dsn" : "TESTDSN_ARE030744_3537",
   "rule_uuid" : "e96fa4d5-2c57-4688-be34-bdceb5b7b2e6",
   "rule_name" : "device in low wifi signal state"",
   "oem_id" : "oemid_123",
   "oem_model" : "oem_model_123",
   "user_uuid" : "16816d94-2709-11e8-a7f8-2903a354ded2",
   "name" : "YELLOW",
   "created_at" : "2018-03-13T21:54:22.056Z"
  }, {
   "uuid" : "16816d95-2709-11e8-a7f8-2903a354ded2",
   "dsn" : "TESTDSN_ARE030744_3537",
   "rule_uuid" : "16816d91-2709-11e8-a7f8-2903a354ded2",
   "rule_name" : "device in low signal state"",
   "oem_id" : "oemid_123",
   "oem_model" : "oem_model_123",
   "user_uuid" : "16816d94-2709-11e8-a7f8-2903a354ded2",
   "name" : "RED",
   "created_at" : "2018-03-13T21:54:22.056Z"
  }  ]
}
```


Response Status Code

200 - Success (see Response)

400 – Bad request.

401 – Failure: Credentials are required to access this resource.

403 – Access is forbidden.

404 – Diagnostic states are not found.

6 Action Types

6.1 GET /rulesservice/v1/action_type

This API gets the complete list of all provisioned action types.

Access Control: Any user with a valid access token.

Pre-condition: N/A

Post-condition: None

Related Tasks: Rules Action Service: When the action is being created, the user chooses the action type. Examples of action types include:

- Datapoint
- Email
- SMS
- OTA
- Diagnostic

Action type has a special role in the lifecycle of abstract rules (also called OEM level rules), such as diagnostic rules, OTA rules, etc. The role of the action type defines one or multiple predefined operations performed at a time of the rule firing. Examples of predefined operations are:

- Updated device diagnostic state - for a diagnostic rule
- Check eligibility of a device for an OTA - for an OTA rule

Validation: N/A

Pagination: N/A

Parameters

None.

JSON Example

```
$ curl -X GET -H "Authorization: auth_token 6edf3c9bcab0485a89c82090a6c61091"
-H "Content-Type: application/json"
https://rulesservice.aylanetworks.com/rulesservice/v1/action_types.json
```

Response

```
{
  "action_types" : [ {
       "uuid" : "62a6b910-1e84-11e8-a6f3-73118b00468e",
       "name" : "DATAPOINT",
       "description" : "action to create a datapoint",
       "is_enabled" : true,
       "created_at" : "2018-03-03T01:44:17.441Z",
       "updated_at" : "2018-03-03T01:44:17.441Z"
```

```
  }, {
       "uuid" : "62c42c20-1e84-11e8-a6f3-73118b00468e",
       "name" : "EMAIL",
       "description" : "action to send an email",
       "is_enabled" : true,
       "created_at" : "2018-03-03T01:44:17.634Z",
       "updated_at" : "2018-03-05T02:59:42.511Z",
       "updated_by" : "4094e6f1-2021-11e8-a71e-0b35af576f4f"
  }, {
       "uuid" : "62c600e0-1e84-11e8-a6f3-73118b00468e",
       "name" : "SMS",
       "description" : "action to send and SMS",
       "is_enabled" : true,
       "created_at" : "2018-03-03T01:44:17.646Z",
       "updated_at" : "2018-03-03T01:44:17.646Z"
  }, {
       "uuid" : "62c7ae90-1e84-11e8-a6f3-73118b00468e",
       "name" : "DIAGNOSTIC",
       "description" : "action to set diagnostic state",
       "is_enabled" : true,
       "created_at" : "2018-03-03T01:44:17.658Z",
       "updated_at" : "2018-03-03T01:44:17.658Z"
  } ]
}
```

Response Status Code

200 - Success (see Response; array of role scopes)

401 – Failure: Authentication required.

6.2 POST /rulesservice/v1/action_types

This API creates an action type. A new action type is enabled by default.

Access Control: N/A

Pre-condition: N/A

Post-condition: None

Related Tasks: Rules Action Service

Validation: N/A

Pagination: N/A

Parameters

None.

JSON Example

```
$ curl -X POST -H "Content-Type: application/json"
https://rulesservice.aylanetworks.com/rulesservice/v1/action_types -d '{
  "action_type" : {
```

```
"name" : "OTA",
"description" : "action to start an OTA job",
"is_enabled" : true,
"created_at" : "2018-03-02T21:16:00.278Z",
"updated_at" : "2018-03-03T00:23:32.599Z",
"id" : 2,
"updated_by" : "b2210f61-1e78-11e8-b0a5-873ba902b030"
  }}
' \
-H "Content-Type: application/json" \
-H "X-Current-User:
eyJpZCI6MTgwLCJvYXV0aCI6ZmFsc2UsInV1aWQiOiJiMjIxMGY2MS0xZTc4LTExZTgtYjBhNS04Nz
NiYTkwMmIwMzAiLCJvZW0iOnsiaWQiOjMsIm1vZHVsZV9tYW51ZmFjdHVyZXIiOmZhbHNlLCJuYW1l
IjoiQXlsYSIsIm9lbV9pZCI6Im9lbS1nRkVBSyJ9LCJvZW1fYXBwcm92ZWQiOnRydWUsIm9yaWdpbl
9vZW1faWQiOjMsIm9yaWdpbl9vZW1fc3RyIjoib2VtLWdGRUFLIiwicm9sZSI6IkF5bGE6OkFkbWlu
Iiwicm9sZV90YWdzIjpbXSwiYXBwbGljYXRpb24iOnsidWlkIjoiZGV2d2Vic2VydmVyX2lkIiwiY2
xvdWRfdG9fY2xvdWQiOnRydWV9LCJzdXBlcl9hcHAiOmZhbHNlLCJsaW5rZWRfdXNlcnMiOltdLCJ1
c2VyIjp7ImZ1bGxuYW1lIjoiVGVzdCBVc2VyIiwiZW1haWwiOiJ0ZXN0QGF5bGFuZXR3b3Jrcy5jb2
0iLCJhZG1pbiI6dHJ1ZSwib2VtX2FkbWluIjp0cnVlfSwiZXhwaXJ5X3RpbWUiOiIyMDE2LTEyLTA4
VDAyOjE0OjExLjAwMFoifQ=="
```

Response (Updated Action Type)

```
{
  "action_type" : {
       "uuid" : "e7777460-1e5e-11e8-a230-93e10f213d13",
       "name" : "OTA",
       "description" : " action to start an OTA job",
       "is_enabled" : true,
       "created_at" : "2018-03-02T21:16:00.278Z",
       "updated_at" : "2018-03-03T00:23:32.599Z",
       "id" : 2,
       "updated_by" : "b2210f61-1e78-11e8-b0a5-873ba902b030"
  }}
```

Response Status Code

200 - Success (see Response; array of role scopes)

403 – Failure: Access forbidden.

404 – Failure: Action type not found.

6.3 PUT /rulesservice/v1/action_types/:action_type_name

This API activates or deactivates an action type by name. All action types are enabled by default with `is_enabled=true`. When `is_enabled` is set to false in an action type, this disables creation of new actions of that type.

Access Control: N/A

Pre-condition: N/A

Post-condition: None

Related Tasks: Rules Action Service

Validation: N/A

Pagination: N/A

Parameters

None.

JSON Example

```
$ curl -X PUT -H "Content-Type: application/json"
https://rulesservice.aylanetworks.com/rulesservice/v1/action_types/email -d
'{
    "attributes": {
    "is_enabled": true
    }
}' \
-H "Content-Type: application/json" \
-H "X-Current-User:
eyJpZCI6MTgwLCJvYXV0aCI6ZmFsc2UsInV1aWQiOiJiMjIxMGY2MS0xZTc4LTExZTgtYjBhNS04Nz
NiYTkwMmIwMzAiLCJvZW0iOnsiaWQiOjMsIm1vZHVsZV9tYW51ZmFjdHVyZXIiOmZhbHNlLCJuYW1l
IjoiQXlsYSIsIm9lbV9pZCI6Im9lbS1nRkVBSyJ9LCJvZW1fYXBwcm92ZWQiOnRydWUsIm9yaWdpbl
9vZW1faWQiOjMsIm9yaWdpbl9vZW1fc3RyIjoib2VtLWdGRUFLIiwicm9sZSI6IkF5bGE6OkFkbWlu
Iiwicm9sZV90YWdzIjpbXSwiYXBwbGljYXRpb24iOnsidWlkIjoiZGV2d2Vic2VydmVyX2lkIiwiY2
xvdWRfdG9fY2xvdWQiOnRydWV9LCJzdXBlcl9hcHAiOmZhbHNlLCJsaW5rZWRfdXNlcnMiOltdLCJ1
c2VyIjp7ImZ1bGxuYW1lIjoiVGVzdCBVc2VyIiwiZW1haWwiOiJ0ZXN0QGF5bGFuZXR3b3Jrcy5jb2
0iLCJhZG1pbiI6dHJ1ZSwib2VtX2FkbWluIjp0cnVlfSwiZXhwaXJ5X3RpbWUiOiIyMDE2LTEyLTA4
VDAyOjE0OjExLjAwMFoifQ=="
```

Response (Updated Action Type)

```
{
  "action_type" : {
       "uuid" : "e7fe9360-1e5e-11e8-a230-93e10f213d13",
       "name" : "EMAIL",
       "description" : "action to send an email",
       "is_enabled" : true,
       "created_at" : "2018-03-02T21:16:00.278Z",
       "updated_at" : "2018-03-03T00:23:32.599Z",
       "id" : 2,
       "updated_by" : "b2210f61-1e78-11e8-b0a5-873ba902b030"
  }}
```

Response Status Code

200 - Success (see Response; array of role scopes)

403 – Failure: Access forbidden.

404 – Failure: Action type not found.

7 Ayla Rule Expression Syntax (ARES)

Ayla Rule Expression Syntax (ARES) is designed to handle rule expressions of arbitrary complexity and nesting.

Following is an example of a rule expression in ARES:

```
VIRTEVENT(uuid_2) && ((DATAPOINT(dsn_1,prop_name_1) -
DATAPOINT(dsn_2, prop_name_2) > DATAPOINT(dsn_3,prop_name_3) +
DATAPOINT(dsn_4,prop_name_4) ) &&
(distance_miles(LOCATION(uuid_1),LOCATION(dsn_1))) < 112)
```

In general, rule expression is a logical statement conveyed in terms of logical relationships between any number of entities, which are called rule subjects. Rule subjects can enter a rule expression as stand-alone terms or as function arguments.

It is assumed that each entity in a rule has a definite value (of type boolean, decimal, etc.) at the time of rule evaluation. Rule evaluation is the process of replacing the rule expression with the equivalent boolean value (true or false). Evaluation is possible upon substitution of every subject in a rule with the corresponding value.

Space characters are optional in rule expressions, and whether or not space characters are included do not change the logical statement in a syntactically valid rule expression.

For example, the above statement expresses relationships between the following seven rule (event) subjects:

1. VIRTEVENT(uuid_2) -virtual event
2. DATAPOINT(dsn_1,prop_name_1) -device datapoint
3. DATAPOINT(dsn_2,prop_name_2) -device datapoint
4. DATAPOINT(dsn_3,prop_name_3) -device datapoint
5. DATAPOINT(dsn_4,prop_name_4) -device datapoint
6. LOCATION(uuid_1) -user location
7. LOCATION(dsn_1) -device location

The last two rule subjects enter the expression only as arguments of the function distance_miles():

```
distance_miles(LOCATION(uuid_1),LOCATION(dsn_1))
```

7.1 Rule Subjects

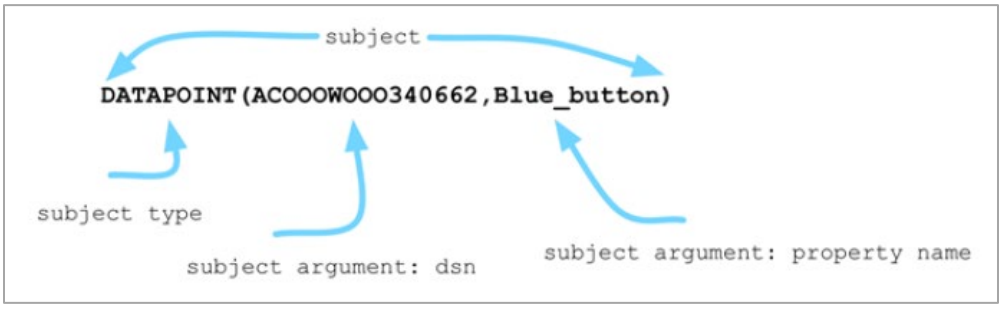
The Ayla Rules Engine's mode of operation is in response to Ayla Events. Rule subjects are the result of a direct derivation from Ayla Events. In a rule expression, the rule subject looks like a function call with a capitalized name.

The list of possible arguments of a subject are as follows:

- dsn

- property name
- user uuid

Subject arguments are used similarly to literals in programming, like without using quotes, as shown in the following example:



Following are supported rule subjects:

Rule Subject	Derived from event	Type
DATAPOINT (dsn,prop_name)	datapoint event	:: datapoint_value
DATAPOINT_ACK(dsn,prop_name)	datapoint event	:: datapoint_value
CONNECTIVITY(dsn)	connectivity event	:: status dsn online or offline
REGISTRATION(dsn)	device registration event	:: boolean dsn registered or not-registered
LOCATION(dsn)	device location change event	:: (lat,long) pair of decimals
LOCATION(uuid)	user's phone location change event	:: (lat,long) pair of decimals
VIRTEVENT (event_type, uuid)	N/A (event is also the subject)	statement of event_type about a user expressed as a boolean
VIRTEVENT (event_type, dsn)	N/A (event is also the subject)	statement of event_type about a user expressed as a boolean
VERSION($device group definition)	OTA version event	

Rule Subject	Derived from event	Type
CONNECTION(dsn connection status value)	Device coming online/offline	True or false depending on the device online/offline status matching or not matching connection status value in the subject. Examples: CONNECTION(dsn offline) is evaluated to true on event device online CONNECTION(dsn offline) is evaluated to true on event "device offline" and to "FALSE" on

7.2 Expressions

There are single subject expressions and multiple subject expressions. A single subject expression conveys the relationship between a rule subject and a constant. A rule subject can enter a single subject expression as:

- An operator to show the relationship between the subject and a constant, for example:

  ```
  Subject Operator Const
  ```

- A function call, for example:

  ```
  function(Subject), or function(Subject, Const)
  ```

- An arbitrary combination of expressions with both operators and functions, for example:

  ```
  Subject_1 > Const_1) || (function(Subject) && function(Subject, Const_2)
  > Const_3)
  ```

A multiple subject expression can include any number of rule subjects in any coherently resolvable form. The following example of a multiple subject expression shows a relationship between three subjects:

```
DATAPOINT(dsn_1,prop_name_1) > DATAPOINT(dsn_2,prop_name_2) &&
DATAPOINT(dsn_1,prop_name_2)
```

NOTE The first two subjects in the example above are decimal types, while the third subject is a boolean type.

Subjects, function calls, and constants must enter rule expressions in such way that after full evaluation (i.e. assignment of their current values to each subject and function call) the derived expression remains logically coherent. For example, if `DATAPOINT(dsn_1, prop_name_1) > 10`

takes the following form after evaluation, the expression has the logical value of FALSE and thus has perfect logical sense:

```
1 > 10
```

Because both the subject and the constant in the expression are the same type (both decimal), the expression is valid.

On the other hand, the structurally equivalent expression `DATAPOINT(dsn_1, prop_name_2) > 10` with the subject that is boolean type, after evaluation can only lead to a logically incoherent results, such as:

```
TRUE > 10 or
FALSE > 10
```

7.3 Evaluation Stages

Every ARES expression has three evaluation stages:

1. Original ARES expression;
2. Pre-evaluated expression (a boolean expression)
3. Evaluated expression (a boolean value of true or false).

As an example, consider the following ARES expression:

```
DATAPOINT(dsn_1, prop_name_2) > 10
```

In this example, if `DATAPOINT(dsn_1, prop_name_2)` evaluates to integer 10, the pre-evaluated expression is as follows:

```
1 > 10
```

The evaluated expression is as follows:

```
false
```

Some expressions in their pre-evaluated form cannot be trivially evaluated. Such expressions are called *syntactically ambiguous*. You can easily validate a rule expression by running it through the evaluation stages. The following table provides examples of pre-evaluated, syntactically ambiguous expressions.

NOTE Because of the ambiguity, using the expression syntax for the pre-evaluated expressions (in the table below) is permitted, but is not recommended.

Also, keep in mind that when numerically compared with each other, true and false are evaluated to 1 and 0 respectively. In an expression with relational operators, any number except 0 (exactly zero) is evaluated to TRUE. Number 0 (exactly zero) is evaluated to FALSE.

Pre-evaluated Expression	Evaluation
true > false	TRUE
true > = false	TRUE
true < false	FALSE
true < = false	FALSE
1 == false	FALSE
0.1 == false	FALSE
0 == true	TRUE
0 == false	FALSE
0 < true	TRUE
0 > true	TRUE
0 <= true	TRUE
0 >=true	FALSE
true == 2.3	TRUE
false == 2.3	FALSE
true > 2.3	FALSE
true < 2.3	FALSE
true <= 2.3	TRUE
true >= 2.3	TRUE
false < 2.3	TRUE
false > 2.3	FALSE
false <= 2.3	TRUE
false >= 2.3	FALSE

7.4 Operators

&&

||

>

<

<=

>=

==

+

-

/

*

7.5 Functions

Arguments that are used in function calls without quotes:

- decimals (i.e., latitude, longitude)
- dsn
- property name
- user uuid

Arguments used in function calls with single quotes:

- string value, for example:

  ```
  str_contains ( DATAPOINT(dsn,prop_name), 'substring' )
  ```

- specific date and time, for example:

  ```
  time( '2017-05-26T20:40:22.000Z' )
  ```

The following table provides the currently supported functions:

Function	Use	Type
Time and Date Functions		
time(dsn)	Converts time on the specified device to epoch seconds	:: long decimal
time(uuid)	Converts time on specified user's phone to epoch seconds	:: long decimal

Function	Use	Type
Time and Date Functions (continued)		
current_time()	Converts current time in UTC/GMT to epoch seconds	:: long decimal
time('2017-05-26T20:40:22.000Z')	Converts specified date and time in UTC/GMT to epoch seconds	:: long decimal
Distance Functions		
distance_miles(LOCATION(dsn) , LOCATION(uuid))	Calculates distance in miles between two locations	:: decimal
distance_km(LOCATION(dsn) , LOCATION(uuid))	Calculates distance in kilometers between two locations	:: decimal
lat_long(latitude,longitude)	Specific location; for use in distance calculations, e.g.: distance_m(LOCATION(uuid_1),lat_long(37.3541,-121.9552)) < 10	:: (lat,long) pair of decimals
String Functions		
str_contains (DATAPOINT(dsn,prop_name), 'substring')	Checks whether value of datapoint (of base type string) contains a substring	:: boolean
str_contains (DATAPOINT(dsn,prop_name), 'other string'	Evaluates whether the value of string typed datapoint equals another string **NOTE:** The compare-to string field 'other string' in this example is surrounded with single quotes.	booean

Following are examples of ARES expressions:

- `DATAPOINT(fred-dsn,bar-property-name) == DATAPOINT(foo-dsn,bar-property-name)`
- `current_time() >= time('2017-05-26 20:40:22')`
- `str_contains(DATAPOINT(foo-dsn, bar-property-name), 'str')`
- `str_equals(DATAPOINT(foo-dsn, bar-property-name), 'another string')`
- `distance_km(LOCATION (foo-uuid ), LOCATION (bar-dsn)) < 112`

8 Abstract Rule Syntax

An abstract rule (also called template rule) defines the business logic on a group of devices. To express applicability of a rule for a group of devices, ARES uses a group notation of `'${...}'` in a rule subject. A subject with a group notation of `'${...}'` is called an abstract subject. Following is an example of an abstract rule syntax:

```
DP(${oem_model=oem_model_value,template_version=template_version_value},Blue_button)
```

NOTE Currently only one abstract subject per rule expression is allowed.

The group of devices in an abstract rule includes all devices with OEM models and template versions equal to the values specified in the template rule subject as key-value pairs. The following table provides the applicability of abstract rules.

Abstract Rule Expression	Applicability

`DP(${oem_model=oem_m1,template_version=tv1},Blue_button) > 0`
 	
`DP(${oem_model=oem_m1,template_version=tv1, template_name=*},Blue_button) > 0`
	All devices with the OEM model `oem_m1`, template version `tv1`, and any DSN.
	NOTE: Expressions on the left side are equivalent.

`DP(${oem_model=oem_m1,template_version=*},Blue_button) > 0`
 	
`DP(${oem_model=oem_m1},Blue_button) > 0`
 	
`DP(${oem_m1,template_version=*,template_name=*},Blue_button) > 0`
	All devices with the OEM model `oem_m1`, any template version, and any DSN.
	NOTE: Expressions on the left side are equivalent.

9 Rule Action Templating Syntax

Action Template Field	Use	Replacement

`{{{dpl_event}}}`	NON-HTML escaped placeholder in URL-type parameter body of the action.
	
	NON-HTML escaped means you want just the JSON, like an API would normally consume (for example if your template is not HTML).
		At the action phase, is substituted with the entire DPL VERSION event that triggered the action. For example:
		
		Before substitution:
		`{{{dpl_event}}}`
		
		After substitution:
		`{ "dpl_event": { "metadata": { "oem_id": "4f9bec12", "oem_model": "ledevb-test", "dsn": "TESTDSN_235978_000", "resource_tags": [], "event_type": "version" }, "version_event": { "oem_model_version": "sanity-test-version", "ayla_model_version": "1.0", "ayla_model": "AY001MUS1" } } }`

Action Template Field	Use	Replacement

`{{dpl_event}}`	NON-HTML escaped placeholder in URL-type parameter body of the action.
	HTML escaped means that your quotes are escaped because you want this value inside of the HTML document (for example if your template is HTML).
		At action phase, is substituted with the entire DPL VERSION event that triggered the action. For example:
		Before substitution:
		`{{dpl_event}}`
		After substitution:
		`{"dpl_event":{"metadata":{"oem_id":"4f9bec12","oem_model":"ledevb-test","dsn":"TESTDSN_235978_000","resource_tags":[],"event_type":"version"},"version_event":{"oem_model_version":"sanity-test-version","ayla_model_version":"1.0","ayla_model":"AY001MUS1"}}}`

`{{event_timestamp_UNIX}}`	In the URL-type parameter body of an action.	At action phase, is substituted with the POSIX timestamp of the DPL event that triggered the action. For example:
		Before substitution:
		`{{event_timestamp_UNIX}}`
		After substitution
		`1508431836620`

`{{event_timestamp_ISO8601}}`	In the URL-type parameter body of an action.	At action phase, is substituted with the timestamp of the DPL event that triggered the action in ISO 8601 notation. For example:
		Before substitution:
		`{{event_timestamp_ISO8601}}`
		After substitution:
		`2018-07-31T16:07:56Z`

10 API Accessibility Rules

Type	Call	Description
Actions	POST	Callable by anybody that owns the device. Thus the post payload has a 'parameters'-> 'datapoint' key whose value includes a reference to a dsn. Code checks if caller owns the dsn
Actions	PUT/ DELETE	Callable by anybody that created the action to start with.
Actions	GET	Callable by anybody that created the action. Others may not call it – OEM Admins may not see their users
Rules	POST /rules	Callable by anybody that owns the device. Thus the post payload has a 'rule' -> 'expression' key whose value includes a reference to a dsn. Code checks if caller owns the dsn
Rules	POST /rules/:rule_uuid	Callable by anybody
Rules	DELETE /rules/:rule_uuid	Callable by anybody that owns the passed in rule_uuid
Rules	GET /rules	Callable by anybody
Rules	GET /rules/:rule_uuid	Creator of rule_uuid
Rules	GET /rules/:rule_uuid/actions	Everyone
Rules	GET /rules/logs	OEM admin
Rules	GET /devices	OEM Admin, OEM staff

4250 Burton Drive, Santa Clara, CA 95054
Phone: +1 408 830 9844
Fax: +1 408 716 2621