Alarm management

It is possible to obtain metric value changes in realtime, when metric value has exceeded or dropped below specified threshold. That can be done using alarms, which can be obtained via STOMP over Websocket by /alarms event subscription.

Alarms can be managed usingWebsocket API orREST API

Alarm creation

A new alarm can be created with /api/alarm/create request

<table>
<tr><th>API</th><th>Request</th><th>Response</th><th>Response status</th></tr>
<tr><td>WS API</td><td><pre>
SEND
destination:/app/api/alarm/create
content-length:174

{
 "requestId":"c71ec29d-b292-46c0-9138-
a8ff434f2c3e",
 "realm":"/api/alarm/create",
 "payload":
 {
  "type":"0",
  "name":"alarm1",
  "value":"100000",
  "metric":"3",
  "node":"3",
  "time":"1000"
 }
}
</pre></td><td><pre>
MESSAGE
destination:/user/service
content-type:application/json;
charset=UTF-8
subscription:sub-1
message-id:3-51
content-length:84

{
 "requestId":"c71ec29d-b292-46c0-9138-
a8ff434f2c3e",
 "status":200,
 "reason":"SUCCESS"
}
</pre></td><td>200 OK
400 Object not found
500 Persist exception</td></tr>
<tr><td>REST API</td><td><pre>
POST: /api/alarm/create "application/json;
charset=utf-8"
{
 "type":"0",
 "name":"alarm2",
 "value":"100000",
 "metric":"3",
 "node":"3",
 "time":"1000"
}
</pre></td><td><pre>
{
 "status":200,
 "reason":"SUCCESS"
}
</pre></td><td>200 OK
400 Object not found
500 Persist exception</td></tr>
</table>

Where

- type – alarm type:
  - 0 – value has dropped below the threshold
  - 1 – value has exceeded the threshold
  - 2 - value is equal to threshold
  - 4 - monotonically increasing value has decreased
  - 5 - monotonically decreasing value has increased
- name – alarm name
- value – threshold value
- metric – metric identifier (for example, video bitrate)
- node – node identifier
- time –time in milliseconds during which the metric value must be above or below the threshold

In the example above, an alarm, which is triggered if the video bitrate of the stream drops below 100 kbps for more than 1 second, was created.

If node Id is not set, the alarm applies to all the nodes on backend server.

Alarm changing

An alarm parameters can be changed with /api/alarm/update request

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/alarm/update
	content-length:183
	
	{
	 "requestId":"a60920eb-257a-451f-937f-1226a3856610",
	 "realm":"/api/alarm/update",
	 "payload":
	 {
	  "id":"6",
	  "type":"0",
	  "name":"alarm1",
	  "value":"100000",
	  "metric":"3",
	  "node":"3",
	  "time":"1000"
	 }
	}
		MESSAGE
		destination:/user/service
		content-type:application/json; charset=UTF-8
		subscription:sub-1
		message-id:3-56
		content-length:84
		
		{
		 "requestId":"a60920eb-257a-451f-937f-1226a3856610",
		 "status":200,
		 "reason":"SUCCESS"
		}
			200 OK
			400 Object not found
			500 Persist exception

REST API	POST: /api/alarm/update "application/json; charset=utf-8"
	{
	 "id":"7",
	 "type":"0",
	 "name":"alarm2",
	 "value":"10000",
	 "metric":"3",
	 "node":"3",
	 "time":"1000"
	}
		{
		 "status":200,
		 "reason":"SUCCESS"
		}
			200 OK
			400 Object not found
			500 Persist exception

Where

- id – alarm identifier
- type – alarm type:
  - 0 – value has dropped below the threshold
  - 1 – value has exceeded the threshold
  - 2 - value is equal to threshold
  - 4 - monotonically increasing value has decreased
  - 5 - monotonically decreasing value has increased
- name – alarm name
- value – threshold value
- metric – metric identifier (for example, video bitrate)
- node – node identifier
- time –time in milliseconds during which the metric value must be above or below the threshold

Alarm deletion

Alarm can be deleted with /api/alarm/delete request

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/alarm/delete
	content-length:101
	
	{
	 "requestId":"c108dbf9-35c0-42e5-814c-0eec57c4de8e",
	 "realm":"/api/alarm/delete",
	 "payload":
	 {
	  "id":"6"
	 }
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;
		charset=UTF-8
		subscription:sub-1
		message-id:3-57
		content-length:84
		
		{
		 "requestId":"c108dbf9-35c0-42e5-814c-0eec57c4de8e",
		 "status":200,
		 "reason":"SUCCESS"
		}
			200 OK
			400 Object not found
			500 Persist exception

REST API	POST: /api/alarm/delete "application/json; charset=utf-8"
	{
	 "id":"7"
	}
		{
		 "status":200,
		 "reason":"SUCCESS"
		}
			200 OK
			400 Object not found
			500 Persist exception

Where

- id – alarm identifier

STOMP messages about the alarm triggering stop when it is deleted.

Obtaining alarm information

An alarm information can be obtained with /api/alarm/list request

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/alarm/list
	content-length:98
	
	{
	 "requestId":"d8e79851-85eb-4df1-bd3a-9f13090e8be5",
	 "realm":"/api/alarm/list",
	 "payload":
	 {
	  "id":""
	 }
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;charset=UTF-8
		subscription:sub-1
		message-id:3-60
		content-length:177
		
		{
		 "requestId":"d8e79851-85eb-4df1-bd3a-9f13090e8be5",
		 "status":200,
		 "reason":"SUCCESS",
		 "payload":[
		  {
		   "id":8,
		   "name":"alarm1",
		   "type":0,
		   "value":100000,
		   "time":1000,
		   "metric":3,
		   "node":3
		  }
		]
		}
			200 OK
			400 Object not found
			500 Persist exception

REST API	POST: /api/alarm/list "application/json;
	charset=utf-8"
	{
	 "id":""
	}
		{
		    "status": 200,
		    "reason": "SUCCESS",
		    "payload": [
		        {
		            "id": 8,
		            "name": "alarm1",
		            "type": 0,
		            "value": 100000,
		            "time": 1000,
		            "metric": 3,
		            "node": 3
		        }
		    ]
		}
			200 OK
			400 Object not found
			500 Persist exception

Where

- id – alarm identifier

If alarm Id is set, the response will contain only that alarm information. If alarm Id is not set,the response will contain list with all the alarms on backend server.

For every alarm, the response contains the same fields as/api/alarm/update request.

Alert message receiving

Alert messages are received if client is subscribed to /alarms queue. Alert message looks as follows:

```
MESSAGE
destination:/alarms
content-type:application/json;charset=UTF-8
subscription:sub-0
message-id:4-187
content-length:242

{
 "timestamp":1561101716609,
 "status":"RAISED",
 "alarmType":"LESS",
 "alarmValue":700000,
 "alarmName":"alarm1",
 "mediaId":"617691c0-93f2-11e9-8808-938c74814152",
 "metricEnumName":"VIDEO_RATE",
 "metricValue":400232,
 "nodeHostName":"test.flashphoner.com"
}
```

Where:

- timestamp - time of alarm raised or cleared
- status - alarm state:
  - RAISED - alarm is raised
  - CLEARED - alarm is cleared
- alarmType - alarm type:
  - LESS– value has dropped below the threshold
  - MORE– value has exceeded the threshold
  - EQUAL- value is equal to threshold
  - MONOTONIC_UP- monotonically increasing value has decreased
  - MONOTONIC_DOWN- monotonically decreasing value has increased
- alarmValue - alarm threshold value
- alarmName - alarm name
- mediaId - media session identifier for the stream which event is occured
- metricEnumName - metric name
- metricValue - metric value by which alarm was raised or cleared
- nodeHostName - server hostname on which the stream is published or played