Rule management

Rules can be managed using Websocket API or REST API.

Rule creation

A new rule can be created with `/api/rule/create` request

API	Request	Response	Response status
WS API	`SEND destination:/app/api/rule/create content-length:150 { "requestId":"41bf9d28-5dbe-45cc-9add-ff443997e1d5", "realm":"/api/rule/create", "payload": { "type":"0", "name":"rule1", "value":"test*", "profiles": ["1"] } }`	`MESSAGE destination:/user/service content-type:application/json;charset=UTF-8 subscription:sub-1 message-id:3-17 content-length:84 { "requestId":"41bf9d28-5dbe-45cc-9add-ff443997e1d5", "status":200, "reason":"SUCCESS" }`	200 OK 400 Object not found 500 Persist exception

API	Request	Response	Response status
REST API	POST: /api/rule/create "application/json; charset=utf-8" { "type":"0", "name":"rule2", "value":"test*", "profiles":["1"] }	{ "status":200, "reason":"SUCCESS" }	200 OK 400 Object not found 500 Persist exception

Where

- `type` – rule type:
  - `0` – stream name template
  - `1` - media session id template
- `name` – rule name
- `value` – rule value (template)
- `profiles` – list of identifiers of profiles, to which the rule should be applied

Rule changing

A rule can be changed with `/api/rule/update` request

API	Request	Response	Response status

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/rule/update
	content-length:163
	
	{
	"requestId":"cc2605d6-ef85-4a7e-9a8f-04733c413b3d",
	"realm":"/api/rule/update",
	"payload":
	{
	"id":"2",
	
	"type":"0",
	
	"name":"rule1",
	
	"value":"test*",
	
	"profiles":["1","2"]
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;charset=UTF-8
		subscription:sub-1
		message-id:3-20
		content-length:84
		
		{
		"requestId":"cc2605d6-ef85-4a7e-9a8f-04733c413b3d",
		"status":200,
		"reason":"SUCCESS"
		}
			200 OK 400 Object not found 500 Persist exception

API	Request	Response	Response status
REST API	POST: /api/rule/update "application/json; charset=utf-8" { "id":"3", "type":"0", "name":"rule2", "value":"test*", "profiles":["1","2"] }	{ "status":200, "reason":"SUCCESS" }	200 OK 400 Object not found 500 Persist exception

Where

- `id` – rule identifier
- `type` – rule type (`0` or `1`)
- `name` – rule name
- `value` – rule value (template)
- `profiles` – list of identifiers of profiles, to which the rule should be applied

Rule deletion

A rule can be deleted with `/api/rule/delete` request:

API	Request	Response	Response status

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/rule/delete
	content-length:100
	
	{
	"requestId":"a1887fc3-c674-4951-b5b8-ff1d69fed097",
	"realm":"/api/rule/delete",
	"payload":
	{
	"id":"2"
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;charset=UTF-8
		subscription:sub-1
		message-id:3-21
		content-length:84
		
		{
		"requestId":"a1887fc3-c674-4951-b5b8-ff1d69fed097",
		"status":200,
		"reason":"SUCCESS"
		}
			200 OK 400 Object not found 500 Persist exception

REST API	POST:
	/api/rule/delete
	"application/json;charset=utf-8"
	{
	"id":"3"
	}
		{
		"status":200,
		"reason":"SUCCESS"
		}
			200 OK 400 Object not found 500 Persist exception

Where

- `id` – rule identifier

Obtaining rule information

A rule information can be obtained with `/api/rule/list` request:

API	Request	Response	Response status

API	Request	Response	Response status

WS API	SEND
	destination:/app/api/rule/list
	content-length:111
	
	{
	"requestId":"e9a2ce8c-ad2a-49c0-9c38-6ba8e4817fdd",
	"realm":"/api/rule/list",
	"payload":
	{
	"id":"1",
	
	"profile":""
	}
	}
		MESSAGE
		destination:/user/service
		content-type:application/json;charset=UTF-8
		subscription:sub-1
		message-id:3-22
		content-length:161
		
		{
		"requestId":"e9a2ce8c-ad2a-49c0-9c38-6ba8e4817fdd",
		"status":200,
		"reason":"SUCCESS",
		"payload":
		[
		{
		"id":1,
		
		"profiles":
		[1,2],
		
		"type":0,
		
		"name":"default",
		
		"value":".*"
		}
		]
		}
			200 OK 400 Object not found 500 Persist exception

API	Request	Response	Response status
REST API	POST: /api/rule/list "application/json; charset=utf-8" { "id":"1", "profile": "" }	{ "status": 200, "reason": "SUCCESS", "payload": [{ "id": 1, "profiles": [1, 2], "type": 0, "name": "default", "value": ".*" }] }	200 OK 400 Object not found 500 Persist exception

Where

- `id` – rule identifier
- `profile` - profile identifier
- `type` – rule type (`0` or `1`)
- `name` – rule name
- `value` – rule value (template)
- `profiles` – list of identifiers of profiles, to which the rule should be applied

If rule Id is set, the response will contain only that rule information. If profile Id is set, the response will contain information only about the rules applied to that profile metrics. If none

of Ids is set, the response will contain list with all the rules on backend server.

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