Управление правилами

- Создание правила
- Изменение правила
- Удаление правила
- Получение информации о правиле

Управление правилами осуществляется при помощиWebsocket API или REST API

Создание правила

Новое правило создается при помощи запроса /api/rule/create

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

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

Здесь:

- type – тип правила:
  - 0 – шаблон имени потока, опубликованного на WCS
  - 1 - шаблон идентификатора медиасессии
- name – имя правила
- value – значение правила (шаблон)
- profiles – список идентификаторов профилей, к которым применяется правило

Изменение правила

Правило может быть изменено при помощи запроса /api/rule/update

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

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

Здесь:

- id – идентификатор правила
- type – тип правила (0 или 1)
- name – имя правила
- value – значение правила (шаблон)
- profiles – список идентификаторов профилей, к которым применяется правило

Удаление правила

Правило может быть удалено при помощи запроса /api/rule/delete

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

Здесь:

- id – идентификатор правила

Получение информации о правиле

Информацию о правиле можно получить при помощи запроса /api/rule/list

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

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

Здесь:

- id – идентификатор правила
- profile - идентификатор профиля

Если указан идентификатор правила, то ответ будет содержать информацию только об этом правиле. Если указан идентификатор профиля, то ответ будет содержать информацию только о правилах, применяемых к метрикам данного профиля. Если ни тот, ни другой идентификатор не заданы, ответ будет содержать список всех правил на бэкенд-сервере.

Состав полей ответа аналогичен составу полей запроса/api/rule/update.