WebAPI для интеграции с APACS 3000 / BIO

Дата модификации: 27.03.2019

Оглавление

Введение

Назначение

WebAPI предоставляет внешнему ПО интерфейс доступа к серверной части ПК APACS через http-протокол.

Возможности WebAPI:

- получение/ редактирования объектов конфигурации (например, уровней доступа, временных зон и т.д.),
- получение/ редактирование пользователей, карт, групп доступа (например, активация/ деактивация, включение/ исключений из уровней доступа, получение/ редактирование дат активации/ деактивации и т.д.),
- выдача команд управления (например, изменение режимов считывателей, управление реле и т.д),
- подписка и последующая рассылка online событий, уведомлений, происходящих в комплексе (например, получение событий «доступ разрешён» или «доступ разрешён», или же «тревог» от датчиков и т.д.),
- «SQL подобные» функции доступа к многочисленным объектам типа: события/ пользователи/ карты.

Детали реализации

Решение поставляется в виде отдельной службы «ApcWebApiService.exe». После запуска служба соединяется с основным сервером APACS (используя настроенную учётную запись APACS) и далее открывает порт для обмена запросами по http-протоколу.

Служба фактически перекодирует внешние Web запросы во внутренний формат комплекса, выполняет их и далее производит обратное преобразование ответов.

Развертывание

Службу WebAPI можно запустить как на компьютере с установленным Сервером APACS, так и на выделенной машине.

Примеры

В составе комплекса поставляются примеры исходных текстов приложений на языке Python. Примеры демонстрируют все основные возможности WebAPI: получение текущих сообщений с сервера APACS, запрос объектов конфигурации и построение дерева конфигурации, получение сохранённых сообщений из базы данных по фильтру. Исходные коды тестовых проектов находятся в папке: [APACS]\Samples\SDK_Web_And_DLL\.

Для получения информации о конкретных свойствах объектов, доступных командах оперативного управления, свойствах сообщений удобно использовать один из следующих методов:

1. *Просмотр информации в справке по объекту*: в приложении «Консоль» или «Консоль + Картотека» открыть нужный объект на редактирование и вызвать справку по объекту, нажав F1. В открывшейся контекстной справке, в конце статьи перейти по ссылке «Идентификаторы параметров объектов». В статье будут указаны идентификаторы объекта, свойств, их типы, ссылка на раздел «Команды управления», где перечислены все поддерживаемые команды и типы входных/ выходных аргументов объекта данного типа
2. *Просмотр информации в окне метаданных*: в приложениях «Консоль» или «Консоль + Картотека» в главном меню выбрать: «Команды -> Показать метаданные». Далее найти нужный тип объекта или события для просмотра всех доступных свойств и команд.

Совместимые версии ПК APACS

Интеграция с APACS 3000/BIO через WebAPI доступна начиная с версии APACS 7.2.2.

Настройка службы WebAPI

Для работы службы WebAPi необходимо:

- Создать оператора APACS, под которым будет работать служба.
- Задать логин пользователя и пароль, под которым авторизуемся веб-клиент. Не рекомендуется делать этот логин и пароль совпадающим с логином и паролем оператора APACS.

Настройки ApcWebApiService задаются в файле [Apacs]\Settings\ApcApplRes\ApcWebSDK\ApcWebApiServiceSettings.js и позволяют указать следующие параметры:

```
{
  "port" : 7110, // Порт веб-сервера APACS
```

```
  "APACSUserLogin" : "inst", // Логин оператора APACS, под которым работает веб-сервер
  "APACSUserPassword" : "",  // Пароль оператора APACS, под которым работает веб-сервер, закодирован base64

  "webUserLogin" : "user",   // Логин веб-оператора, под которым работает веб-клиент
  "webUserPassword" : "cGFzc3dvcmQ=", // Пароль веб-оператора, под которым работает веб-клиент, закодирован base64

  "eventBufSizeForWebHook" : 1000, //Размер кольцевого буфера на каждого подписчика вебхуков
  "batchSize" : 30, //Максимальный размер пачки событий для отправки через вебхуки в одном запросе
  "sendKeepAlive" : true, //Пинговать ли подписчика при отсутсвии сообщений от нас
  "keepAliveInterval" : 30000, //Интервал, мс, через который выполнять пинг подписчика, если не было сообещний для подписки
  "autoKillSubscriber" : true, //Удалять автоматически подписчика, если в течении killSubscriberTimeOut после первой неудачной отправки не было ни одного успеха на отправку вебхука или пинга
  "killSubscriberTimeOut" : 30, //Время, мин, через которое автоматически убивается подписчик, если в течении этого времени с первой неудачной отправки не было ни одного успеха на отправку вебхука или пинга
  "retryInterval" : 20000, //Интервал, который надо выждать после неудачи перед следующей попыткой

  "socketTimeout" : 300, //Таймаут чтения/записи в сокет, мс
  "useSSL" : true //использовать или нет SSL для Службы WebAPI (использовать или нет SSL для WebHooks отпределяется подписчиком)
}
```

Общие моменты

- Для авторизации веб-клиента используется тип авторизации base, логин и пароль передаются в каждом запросе.
- Текущая версия WebAPI: v1 (Добавляется в начало URL при конструировании запроса, например, v1/object/root
- В запросах используется кодировка utf-8.
- Внутренний идентификатор объектов и событий APACS имеет вид SA 0000.0034C5E8 при передачи такого идентификатора в строке URL необходимо заменять пробелы на %20
- В случае ошибки возвращается ответ 400 bad request, в body содержится дополнительная информация об ошибки вида

```
{
"error": 589836,
"translation" : "Объект не найден в локальном хранилище"
}
```

TLS/SSL

Соединение между службой WebAPI и web-клиентом может быть защищено с помощью протокола TLS/SSL. По умолчанию режим «использовать TLS/SSL включен». В качестве сертификатов по умолчанию используются сгенерированные разработчиками самоподписанные сертификаты, которые необходимо сменить при внедрении решения.

Включение/отключение TLS/SSL выполняется в файле [APACS]\Settings\ApcApplRes\ApcWebAPIService\ApcWebApiServiceSettings.js, настройка useSSL

Детальные настройки TLS/SSL для серверного и клиентского соединения указываются в файле: [APACS]\Settings\ApcApplRes\ApcWebAPIService\ApcWebAPISSLSettings.ini

В качестве сертификатов по умолчанию сгенерированы:

- Корневой самоподписанный сертификат (ApcWebAPIRootCert.pem)
- Серверный сертификат, подписанный корневым (ApcWebAPIServer_Cert.pem)
- Клиентский сертификат, подписанный корневым (ApcWebAPIClient_Cert.pem)

Корневой сертификат указывается для всех в качестве доверительного.

APACS Web API позволяет получать уведомления о новых событиях/изменениях конфигурации/состоянии соединения с сервером через механизм webhooks. В этом случае использовать или нет TLS/SSL для получения данных уведомлений определяет клиент в момент подписки.

Если в качестве обратного адреса для получения webhooks указан адрес вида:

- http:// - соединение не защищено
- https:// - соединение с шифрованием TLS/SSL

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха
Object		Команды работы с конфигурацией		
		Получение/редактирование/удаление объекта		
getObject	GET	object/id/{sysAddr}		200 body:json:объект
editObject	PUT	object/id/{sysAddr}	Json:объект	204 No Content
delObject	DELETE	object/id/{sysAddr}		204 No Content
		Добавление нового объекта		
getObjectForAdd	GET	object/forAdd/{sysAddr}/{childClassID}		200 body:json:объект
addObject	POST	object/id/{sysAddr}	Json:объект	201 (Created) location+ body:json:сисадрес
		Получение объекта по псевдониму		
getObjectByAlias	GET	object/alias/{alias}		200 body:json:объект
		Вычитка дерева конфигурации		
getRoot	GET	object/root		200 body:json:объект
getChildren	GET	object/children/{sysAddr}		200 body:json:объект
		Команды управления		
execCmdOnObj	PUT	object/execCmd/{sysAddr}/{cmdID}		204 No Content
		Общие настройки групп доступа		
getCommonSGSettings	GET	object/commonSG/{sysAddr}		200 Json:SGOptions
setCommonSGSettings	PUT	object/commonSG/{sysAddr}	Json:SGOptions	204 No Content
Query		Выборки по конфигурации/сообщениям		
openQuery	POST	query/	Json:sql	201 (Created) location+ body:json:queryID&eof
getQueryResult	GET	query/{queryID}/{maxRecordInResult}		200 body:json:выборка
closeQuery	DELETE	query/{queryID}		204 No Content
Event		Команды работы с сообщениями		
		Получение/удаление события		
getEvent	GET	event/id/{eventID}/{eventType}		200 body:json:событие
delEvent	DELETE	event/id/{eventID}/{eventType}		204 No Content
		Регистрация сообщений		
getEventForAdd	GET	event/forAdd/{EventTypeID}		200 body:json:сообщения

registerEvent	POST	event/	Json:сообщение	201 (Created) location+ body:json:сисадрес
		Получение событий по опросу		
getRecentEvent	GET	event/recent/		200 body:json:последнее событие или {}
getEventsAfter	GET	event/after/{sysAddrEvent}/{maxEventInResult}		200 body:json:массив сообщений
		Получение событий через WebHooks		
subscribe	PUT	callback/subscribe	Json:subscription params	204 No Content
unsubscribe	PUT	callback/unsubscribe	Json:subscription params	204 No Content

Команды работы с конфигурацией

Получение/редактирование/удаление объекта

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

getObject
 	
Получить свойства указанного объекта
	GET	object/id/{sysAddr}
		
		{sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20
				200 Ok
				
				{
				"strName": "Папка оператора 1",
				"strDesc": "",
				"IsActive": "true",
				"dtCreateTime": "20.12.2018 12:02:35:753",
				"dtLastModifyTime": "20.12.2018 12:02:35:753",
				"dtLastAccessTime": "30.12.1899 00:00:00:000",
				"strAlias": "",
				"strClassID": "TApcFolder",
				"sysAddrID": "SA 0000.00000636"
				}

editObject
 	
Отредактировать указанный объект
	PUT	object/id/{sysAddr}
		
		{sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20
			Список изменяемых свойств:
			
			{
			"strName": "Папка оператора Новое имя",
			"strDesc": "Новое описание"
			}
			
			Свойства, которые не могут быть отредактированы (даты создания, модификации, последнего обращения, сисадрес, тип объекта, адрес родительского объекта) будут проигнорированы
				204 No Content

delObject	DELETE	object/id/{sysAddr}		204 No Content

Удалить указанный объект		{sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20		

Добавление нового объекта

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

getObjectForAdd
 	
Получить список свойств по умолчанию для добавления объекта указанного типа к указанному экземпляру родителя
	GET	object/forAdd/{sysAddr}/{childClassID}
		
		{sysAddr} - сисадрес объекта, к которому будет выполнять добавление, вида SA 0000.00000345, пробелы должны быть заменены на %20
		
		{childClassID} – тип добавляемого объекта
				Дефолтный объект для добавляения
				
				Пример ответа:
				
				{
				"strName": "",
				"strDesc": "",
				"IsActive": "true",
				"dtCreateTime": "20.12.2018 12:02:35:753",
				"dtLastModifyTime": "20.12.2018 12:02:35:753",
				"dtLastAccessTime": "30.12.1899 00:00:00:000",
				"strAlias": "",
				"strClassID": "TApcFolder",
				"sysAddrID": "SA 0000.00000000"
				}
				
				Атрибуты dtCreateTime, dtLastModifyTime, dtLastAccessTime, sysAddrID заполнять не надо – они автоматически проставляются на сервере

addObject	POST	object/id/{sysAddr}	объект для добавления,	201 (Created)

Добавить новый объект к указанному объекту		{sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20	в атрибутах обязательно должен быть атрибут strClassID с правильно заполненным типом добавляемого объекта
			
			Пример:
			
			{
			"strName": "Папка оператора Name",
			"strDesc": "",
			"IsActive": "true",
			"strAlias": "",
			"strClassID": "TApcFolder",
			}
				в location адрес для get объекта
				
				в body сисадрес добавленного объекта:
				
				{
				'sysAddrID': 'SA 0000.0000063E'
				}

Получение объекта по псевдониму

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

getObjectByAlias	GET	object/alias/{alias}
		
		{alias} - псевдоним объекта, если в псевдониме используются пробелы, спецсимволы, русский язык – необходимо выполнить encoding
				200 Ok
				
				{
				"strName": "Папка оператора 1",
				"strDesc": "",
				"IsActive": "true",
				"dtCreateTime": "20.12.2018 12:02:35:753",
				"dtLastModifyTime": "20.12.2018 12:02:35:753",
				"dtLastAccessTime": "30.12.1899 00:00:00:000",
				"strAlias": "",
				"strClassID": "TApcFolder",
				"sysAddrID": "SA 0000.00000636"
				}

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

getRoot
Получить корневой объект конфигурации
	GET	object/root		200 Ok
				Корневой объект системы
				
				{
				'IsActive': True,
				'dtCreateTime': '24.03.2005 17:20:54:000',
				'dtLastAccessTime': '30.12.1899 00:00:00:000',
				'dtLastModifyTime': '24.03.2005 17:20:54:000',
				'strAlias': '',
				'strDesc': 'Корень системы',
				'strName': 'Корень системы',
				'sysAddrParentID': 'SA 0000.00000000',
				'strClassID': 'TApcRoot',
				'sysAddrID': 'SA 0000.00000001'
				}

getChildren Получить дочерние объекты	GET	object/children/{sysAddr} {sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20		200 Ok Массив указателей на дочерние объекты. Указатель на дочерний объект содержит тип объекта и сисадрес. [{'strClassID': 'TApcFolder', 'sysAddrID': 'SA 0000.000003E2'}, {'strClassID': 'TApcSecurityFolder', 'sysAddrID': 'SA 0000.000003D8'}, {'strClassID': 'TApcPC', 'sysAddrID': 'SA 0000.00000002'}]

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

execCmdOnObj
 	
Выполнить команду управления на объекте
	PUT	object/execCmd/{sysAddr}/{cmdID}
		
		{sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20
		
		{cmdID} – идентификатор команды. Список идентфикаторов доступных команд может быть просмотрен в приложение «Консоль» APACS в окне метаданных
			{
			1: 'value1',
			2: 'value2'
			}
			
			Json с входными параметрами, в качестве атрибута – номер параметра, в значении – значение параметра.
				200 Ok
				{
				}
				
				Json с выходными параметрами, если они есть.
				
				На данный момент поддерживаются только выходные/выходные параметры простых типов (str, int, bool и т.п.)

Общие настройки групп доступа

Данный набор команд позволяет изменять общие настройки группы доступа.

Они могут быть применены к объектам типа «Владелец карты» (TApcCardHolder), «Идентификатор» (TApcAccount), «Группа доступа» (TApcSecurityGroup)

Для владельца карты и идентификатора можно изменить общие настройки собственной группы доступа, для группы доступа – общие настройки данной группы.

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха
getCommonSGSettings	GET	object/commonSG/{sysAddr} {sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20		200 Json с общими настройками группы доступа, см. описание ниже
setCommonSGSettings	PUT	object/commonSG/{sysAddr} {sysAddr} - сисадрес объекта вида SA 0000.00000345, пробелы должны быть заменены на %20	Json с общими настройками группы доступа, см. описание ниже	204 No Content

Json с общими настройками группы доступа

Json с общими настройками группы доступа может содержать настройки для следующих подсистем:

- AIMPropsEditor – общие настройки для всех контроллеров Apollo AIM
- AANPropsEditor – общие настройки для всех контроллеров Apollo AAN
- APNPropsEditor– общие настройки для всех контроллеров Apollo APN
- Asp4PropsEditor– общие настройки для всех контроллеров Apollo ASP
- ReferencePropsEditor – общие логические настройки (макет карты задаваемый на группу доступа)

- SupACPropsEditor – общие настройки для всех контроллеров Suprema 1-ого поколения
- SupAC2PropsEditor – общие настройки для всех контроллеров Suprema 2-ого поколения
- VertXPropsEditor – общие настройки для всех контроллеров HID VertX

Список доступных подсистем зависит от лицензии. Ниже приведен максимально возможный вариант.

Пример json с общими настройками группы доступа


```
{'AIMPropsEditor':
  {'APBExempt': {'value': True, 'use': True},
   'UseLongTimes': {'value': True, 'use': True}},
 'AANPropsEditor':
  {'HostCheckBeforeDenied': {'value': False, 'use': False},
   'HostCheckBeforeGrant': {'value': False, 'use': False},
   'APBExempt': {'value': False, 'use': False},
   'UseLongTimes': {'value': False, 'use': False},
   'ActDate': {'value': '30.12.1899', 'use': False},
   'ActTime': {'value': '00:00:00:000', 'use': False},
   'DeactDate': {'value': '30.12.1899', 'use': False},
   'DeactTime': {'value': '00:00:00:000', 'use': False},
   'VisEscortType': {'value': '0', 'use': False}},
  'APNPropsEditor':
  {'APBExempt': {'value': False, 'use': False},
   'UseLongTimes': {'value': False, 'use': False}},
  'Asp4PropsEditor':
  {'HostCheckBeforeDenied': {'value': False, 'use': False},
   'HostCheckBeforeGrant': {'value': False, 'use': False},
   'APBExempt': {'value': False, 'use': False},
   'UseLongTimes': {'value': False, 'use': False},
   'AllowEntryIntoClosedZone': {'value': False, 'use': False},
   'APBOnlyMsg': {'value': False, 'use': False},
   'ActDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
   'DeactDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
   'EveryDayActTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
   'EveryDayDeactTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
   'ThreatLevel': {'value': '0', 'use': False},
   'VisEscortType': {'value': '0', 'use': False}},
  'ReferencePropsEditor':
  {'BadgeModel': {'value': 'SA 0000.00000000', 'use': False}},
  'SupACPropsEditor':
  {'AdminLevel': {'value': False, 'use': False},
   'SecurityLevel': {'value': '0', 'use': False},
   'AuthMode': {'value': '0', 'use': False},
```

'BypassCard': {'value': False, 'use': False},
'ActDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
'DeactDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
'AuthLimitCount': {'value': '0', 'use': False},
'TimedAntiPassback': {'value': '0', 'use': False}},
'VertXPropsEditor':
{'APBExempt': {'value': True, 'use': True},
'PINExempt': {'value': False, 'use': False},
'UseExtTimes': {'value': False, 'use': False},
'AllowPINCmds': {'value': False, 'use': False},
'ActDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
'DeactDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False}},
'SupAC2PropsEditor':
{'SecurityLevel2': {'value': '0', 'use': False},
'FingerAuthMode': {'value': '0', 'use': False},
'CardAuthMode': {'value': '0', 'use': False},
'IdAuthMode': {'value': '0', 'use': False},
'ActDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False},
'DeactDateTime': {'value': '30.12.1899 00:00:00:000', 'use': False}}}

Выборки по конфигурации и сообщениям (Query)

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

openQuery
 	
Выполнить запрос
	POST	query/	{
			"select" : "*",
			"from" : "TApcCardHolder"
			}
			
			См. раздел « Возможности языка запросов» для получения подробной информации.
				201 Created
				
				в location адрес для получения записей из выборки
				
				в body - информация об открытой выборке:
				
				{
				
				'queryID': 'TApcDBThread_ClientQuery_48',
				
				'eof': False
				
				}
				
				queryID - идентификатор выборки , который используется в дальнейшем для получения записей
				
				eof - признак был ли достигнут конец выборки

getQueryResult Получить указанное количество записей из выборки	GET	query/{queryID}/{maxRecordInResult} {queryID} – идентификатор query {maxRecordInResult} - максимальное количество записей для получения в ответе		200 Ok { "header": ["strName", "strDesc", "strFirstName"], "data": [["Буклова Я. Д.", "", "Янина"], ["Кузнецов В. О.", "", "Вячеслав"]

				], "eof": False } header - массив с идентификаторами свойств data - массив записей (порядок значений в записи соответствует порядку в header) eof - признак был ли достигнут конец выборки
closeQuery Закрыть выборку	DELETE	query/{queryID} Примечание: выборки автоматически закрываются на сервере при достижении конца выборки, поэтому вызывать команду closeQuery стоит только в том случае, если необходимо закрыть выборку до того, как был достигнут конец выборки.		204 No Content

Особенности: все выборки являются однопроходными, автоматически закрываются при достижении конца выборки.

Возможности языка запросов

На данный момент поддерживаются запросы только по одному типу объекта/сообщения. Идентификатор типа объекта/сообщения и идентификаторы доступный свойств можно узнать в приложение «Консоль» APACS в окне метаданных.

Образец оформления запроса:

```
{
  "select" : ["strName", "strDesc", "strFirstName"],
  "from" : "TApcCardHolder",
  "where" : {"=" : ["strFirstName", "Иван"]} ,
  "order by" : {"strFirstName" : "asc"}
}
```

- В атрибуте select указываем массив свойств объекта, которые хотим вычитывать или *, если надо вычитать все свойства
- В атрибуте from - идентификатор типа объекта (на данный момент можно указывать только один тип)
- В атрибуте where - условия выборки, которые представляют собой дерево операций.

  Каждая операция имеет один атрибут, который соответствует типу операции (and, or, =, <>, >=, <=, not, is null, not is null), в качестве значений этого атрибута - массив аргументов.

  Для операций and, or, not аргументами может быть только другая операция.

  Для =, <>, >=, <= хотя бы один аргумент должен быть или свойством, или операцией upper над свойством, второй аргумент может быть или константой или другим свойством

  Для is null, not is null - аргументом может быть только свойство объекта

- В атрибуте order by - условие сортировки, может быть или одно условие, или массив с условиями сортировки
  В качестве атрибута условия сортировки указывается свойство объекта, в качестве значения asc (по возрастанию) или desc (по убыванию)

Примеры разных запросов:

```
{
  "select" : ["strName", "strDesc", "strFirstName"],
  "from" : "TApcCardHolder",
  "where" : {"or" : [
                {"=" : ["strFirstName", "Иван"]} ,
                {"=" : ["strFirstName", "Петр"]}
            ]},
  "order by" : [{"strFirstName" : "asc"}, {"strDesc" : "asc"}]
}
```

Для сравнения строк по шаблону в константном выражении можно использовать символы %%

```
{
  "select" : ["strName", "strDesc", "strFirstName"],
```

```
  "from" : "TApcCardHolder",
  "where" : {"not" :
                {"or" : [
                    {"=" : ["strLastName", "%Иван%"]} ,
                    {"=" : ["strLastName", "Петров"]}
                ]}
            }
}
```

Операция upper (приведение к верхнему регистру) поддерживается только для свойств, константы надо сразу самим писать в верхнем регистре

```
{
  "select" : ["strName", "strDesc", "strFirstName"],
  "from" : "TApcCardHolder",
  "where" : {"<>" : [{"upper" : "strFirstName"}, "ИВАН"]}
}
```

События

Получение/удаление события из базы

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха
getEvent	GET	event/id/{eventID}/{eventType}		200 body:json:событие { 'msgType': 'event', 'strHolderName': 'Александров В. О.', 'SysAddrHolder': 'SA 0000.00005435', 'strEventTypeID': 'TApcCardHolderAccess_Correction', 'dtRealDateTime': '30.12.1899', 'dtRegisterTime': '28.02.2019 17:38:55', 'SysAddrInitObj': 'SA 0000.00008880', 'strInitObjName': '', 'SysAddrEventID': 'SA 0000.0061B197'}
delEvent	DELETE	event/id/{eventID}/{eventType}		204 No Content

Регистрация события

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха
getEventForAdd	GET	event/forAdd/{EventTypeID}		200 body:json:сообщения Список свойств по умолчанию для указанного типа сообщеня Пример: { 'SysAddrEventID': 'SA 0000.00000000',

				'dtRealDateTime': '30.12.1899 00:00:00:000', 'dtRegisterTime': '30.12.1899 00:00:00:000', 'strEventTypeID': 'TApcCardHolderAccess_Correction', 'SysAddrInitObj': 'SA 0000.00000000', 'strInitObjName': '', 'strHolderName': '', 'SysAddrHolder': 'SA 0000.00000000'} При регистрации нового события обязательны для заполнения поля: SysAddrInitObj – адрес объекта, от которого регистрируется событие strInitObjName – имя объекта, от которого регистрируется событие dtRealDateTime – дата возникновения события (может отличаться от текущих даты и времени, например для событий коррекции УРВ) dtRegisterTime – этот атрибут можно не проставлять, так как будет заполнен на сервере автоматически SysAddrEventID – этот атрибут можно не проставлять, так как будет присвоен новый адрес на сервере автоматически Остальные атрибуты зависят от конкретного типа события, как правило должны быть заполнены

registerEvent	POST	event/	Json:сообщение
			
			{
			'SysAddrEventID': 'SA 0000.00000000',
			'dtRealDateTime': '30.12.1899 00:00:00:000',
			'dtRegisterTime': '30.12.1899 00:00:00:000',
			'strEventTypeID': 'TApcCardHolderAccess_Correction',
			'SysAddrInitObj': 'SA 0000.00000000',
			'strInitObjName': '',
			'strHolderName': '',
			'SysAddrHolder': 'SA 0000.00000000'}
				201 (Created)
				
				в location адрес для get сообщения
				
				в body сисадрес добавленного сообщения:
				
				{
				"sysAddrID" : "SA 0000.0061AA4A"
				}

Получение событий по опросу (Events Polling)

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

getRecentEvent
 	
Получить самое свежее событие
	GET	event/recent/		200
				
				{
				'bRelayMode': '1',
				'strEventTypeID': 'TAplSCEvRelayChange',
				'dtRealDateTime': '16.01.2019 10:12:00',
				'dtRegisterTime': '16.01.2019 10:12:36',
				'SysAddrInitObj': 'SA 0000.00000023',
				'strInitObjName': 'APN-35 Доп. реле № 2',
				'SysAddrEventID': 'SA 0000.0034C5E8'
				}
				
				Или, если в буфере нет ни одного сообщения:
				
				{}
				
				Список атрибутов зависит от типа события, перечень для конкретного типа можно увидеть в приложении Консоль в окне метаданных.

getEventsAfter Получить из буфера события, пришедшие позже указанного	GET	event/after/{sysAddrEvent}/{maxEventInResult} {sysAddrEvent} - идентификатор последнего обработанного события ('SysAddrEventID' из ответа на getRecentEvent или последнего события от предыдущего запроса getEventsAfter) {maxEventInResult} – максимальный размер пачки возвращаемых событий		200 body:json:массив сообщений [{ 'bRelayMode': '1', 'strEventTypeID': 'TAplSCEvRelayChange', 'dtRealDateTime': '16.01.2019 10:12:00', 'dtRegisterTime': '16.01.2019 10:12:36', 'SysAddrInitObj': 'SA 0000.00000023', 'strInitObjName': 'APN-35 Доп. реле № 2', 'SysAddrEventID': 'SA 0000.0034C5E8'

				}, …]

Для получения событий по опросу создается буфер событий. Размер буфера конфигурируется в файле [Apacs]\Settings\ApcApplRes\ApcNativeSDK\ApcNativeSDKSettings.js в параметре eventBufferSize

Получение событий через WebHooks (Events WebHooks)

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха

subscribe
 	
Подписаться на получение webhooks
 	
При повторном вызове для того же url, выполняется редактирование параметров подписки
	PUT	callback/subscribe	{
			"url" : "http://127.0.0.1:7112/webhook",
			"event" : true,
			"notify" : true,
			"status" : true
			}
			
			url – адрес подписчика
			event – true/false подписываться ли на события
			notify - true/false подписываться ли на уведомления об изменении конфигурации
			status - true/false подписываться ли на уведомления о потери /восстановлении соединения с Сервером APACS
				204 No Content

unsubscribe
 	
Отписаться от получения WebHooks
	PUT	callback/unsubscribe	{
			"url" : "http://127.0.0.1:7112/webhook",
			}
			
			url – адрес подписчика
				204 No Content

Запросы для обработки подписчиком

Команда	Тип запроса	URL	Входной параметр в Body	Ответ в случае успеха
onMessage	POST	[url]	Массив уведомлений о новых событиях, произошедших в системе В зависимости от того, какие опции были указаны при подписке, могут приходить: • Сообщения APACS • Уведомления об изменении конфигурации • Уведомление о статусе соединения с APACS Помимо этого, если в настройка веб-службы указано выполнять периодически пинг подписчика (настройка sendKeepAlive в файле ApcWebApiServiceSettings.js), то может еще приходить событие keepAlive.	200 OK или 204 No Content Прочее интерпретируется как не доставлено и будет повторная попытка отправки спустя таймаут

			Атрибут msgType определяет тип конкретного события:
			
			"msgType" : "event" – сообщение
			"msgType" : "notify" – изменение конфигурации
			"msgType" : "status" – изменение статуса соединения с сервером APACS
			"msgType" : "keepAlive" – пинг
			
			Пример сообщения:[
			{
			"msgType" : "event",
			"dwCardNumber" : "12",
			"isOffLineEvent" : false,
			"SysAddrCard" : "SA 0000.00000580",
			"SysAddrHolder" : "SA 0000.000001BF",
			"strHolderName" : "Кротов А. Б.",
			"strEventTypeID" : "TApcCardHolderAccess_Granted",
			"dtRealDateTime" : "16.01.2019 9:38:00",
			"dtRegisterTime" : "16.01.2019 9:38:18",
			"SysAddrInitObj" : "SA 0000.00000028",
			"strInitObjName" : "Аренда Этаж № 1 - Вход",
			"SysAddrEventID" : "SA 0000.0034BA71"
			}]
			Список атрибутов зависит от типа события, перечень для конкретного типа можно увидеть в приложении Консоль в окне метаданных.
			
			Пример уведомления об изменении конфигурации:
			
			[
			{
			"msgType" : "notify",
			"msgSubType" : "edit",
			"sysAddrID" : "SA 0000.000000F3",
			"strName" : "Охрана - Вход (2)",
			"strClassID" : "TAplMCReader",
			"newValues" : {
			"bCurrStatus" : "5"
			}
			}	

			]
			msgSubType – edit/add/del - тип операции
			sysAddrID – системный адрес изменённого объекта
			strName – имя объекта
			strClassID – тип объекта
			
			Список измененных атрибутов зависит от типа объекта, перечень для конкретного типа можно увидеть в приложении Консоль в окне метаданных.
			
			Пример уведомления о потери/восстановлении соединения между сервером WebAPI и сервером APACS
			[
			{
			“msgType” : “status”,
			"connected" : true
			}
			]
			
			Пример пинга:
			
			[
			{
			“msgType” : “keepAlive”
			}
			]	

Буфер событий

На каждого подписчика Службой WebAPI создается кольцевой буфер событий для отправки. В случае, если соединение между Службой WebAPI и подписчиком провисает, этот буфер может начать расти. Максимальный размер буфера конфигурируется в файле [Apacs]\Settings\ApcApplRes\ApcWebSDK\ApcWebApiServiceSettings.js в параметре eventBufSizeForWebHook

Сценарии работы в случае неуспеха отправки webhook в течении длительного времени.

1. Автоматическое удаление подписчика в случае продолжительного неуспеха

   В этом случае подписчик автоматически удаляется, если в течении длительного времени не удается доставить webbhook или сообщение keepalive или, если подписчик не отвечает на уведомления кодами 200 или 204

   Настройки интервала, через которое происходит удаление подписчика, укзаываются в файле [Apacs]\Settings\ApcApplRes\ApcWebSDK\ApcWebApiServiceSettings.js

   "autoKillSubscriber" : true, //Удалять автоматически подписчика, если в течении killSubscriberTimeOut после первой неудачной отправки не было ни одного успеха на отправку вебхука или пинга
   "killSubscriberTimeOut" : 30, //Время, мин, через которое автоматически убивается подписчик, если в течении этого времени с первой неудачной отправки не было ни одного успеха на отправку вебхука или пинга

2. Подписчик не удаляется никогда.

   В этом случае в кольцевом буфере будут хранится последние событий, которые будут отосланы при успешной установке соединения. Для включения данного режима, необходимо выставить в файле настроек

   "autoKillSubscriber" : false

Повторная подписка

При получении запроса subscribe Сервер WebAPI определяет, существует ли уже такой подписчик с указанным url. Если подписчик найден, то произойдет только обновление параметров подписки. (Буфер событий не чистится)

Пинг подписчика

Сервер WebAPI позволяет настроить автоматический пинг подписчика, чтобы дать простой механизм определения, находится ли Сервер WebAPI на связи. Настройки задаются в файле [Apacs]\Settings\ApcApplRes\ApcWebSDK\ApcWebApiServiceSettings.js

"sendKeepAlive" : true, //Пинговать ли подписчика при отсутствии сообщений от нас

"keepAliveInterval" : 30000, //Интервал, мс, через который выполнять пинг подписчика, если не было сообщений для подписчика

Пинг выполняется только в том случае, если за указанный инетрвал времени не было других попыток отправки уведомлений подписчику.

Размер пачки событий

В случае, если в буфере накапливаются события, то они будут отправлены пачкой в порядке возникновения. В файле настроек можно указать максимальный размер пачки

"batchSize" : 10, //Максимальный размер пачки событий для отправки через вебхуки в одном запросе

Надо учитывать, что размер одного HTTP-запроса как правило ограничен, поэтому не рекомендуется делать это число очень большим.

TLS/SSL

Особенности TLS/SSL для подписчиков описаны в разделе TLS/SSL