Приложение **Integration Client** предназначено для интеграции ПО FireSec c программными продуктами стороннего производителя. По средствам программной интеграции возможно управление сторонним оборудованием на ПО FireSec.

Компонент интеграции Integration Client может отправлять и принимать информацию по протоколам HTTP, OPC и WSDL.

# Описание работы HTTP

Настроим обмен данными в **Integration Client** по протоколу HTTP. В этом случае Integration Client будет получать данные о событиях, параметрах, состояниях и т.д. из FireSec и отсылать клиенту в формате xml.

Для этого необходимо выполнить следующие действия:

1. Запустить приложение Администратор FireSec3 (См. [2](#запуск)) и открыть нужную конфигурацию;
2. Воспользовавшись кнопкой , сохранить конфигурацию на сервере (Рис. 1);



Рис. 1 Окно приложения **Администратор**

1. Запустить **FS\_IntegrationClient**. Путь: *C:\ProgramFiles(x86)\FireSec3\ FS\_IntegrationClient;*
2. В открывшемся окне **Ввод имени пользователя и пароля** нужно ввести adm в графе *Пользователь* и нажать **Ок** (по умолчанию пароля нет);
3. В результате откроется окно настройки **Сервисы интеграции FireSec** (Рис. 2), в котором уже загружена наша конфигурация:



Рис. 2 Окно IntegrationClient

1. Чтобы пополнить список клиентов необходимо в правой части окна в поле *Клиенты* нажать кнопку **Добавить** и в открывшемся окне **Свойства HTTP-клиента** ввести IP- адрес и номер порта компьютера, который будет принимать информацию, а затем нажать **Ok** (Рис. 3) (следует выбрать незанятый порт и добавить его в исключения в Firewall и Антивирус);



Рис. 3 Окно Свойства HTTP-клиента

1. В поле *Фильтрация* доступен фильтр по устройствам (закладка *Устройства*), зонам (закладка *Зоны*), классам состояний (закладка *Классы*), подсистемам (закладка *Прочее*) доступным для просмотра клиенту. Выбор осуществляется простановкой галочки напротив соответствующего элемента.

Для проверки установим все возможные галочки фильтров (Рис. 4) и закроем приложение.











Рис. 4 Фильтрация по объектам системы

1. Затем на компьютере – клиенте запустим проверочную утилиту **IntegrationClient\_Test**  (чтобы проверить обмен данными запрос-ответ) По умолчанию она находится в папке с установленным FireSec: C:\ProgramFiles(x86)\FireSec3\Utils\IntegrationClient\_Test
2. В результате откроется окно следующего вида:



Рис. 5 Проверочная утилита IntegrationClient Test

1. В поле **Server** в графе **Port** следует ввести номер порта, который указан в **FS\_IntegrationClient** и нажать кнопку **Run**. После чего снова запустить Integration Client;
2. Если все сделано верно, то в окне **IntegrationClient\_Test** отобразится информация из Integration Client следующего вида (Рис. 6):



Рис. 6 Окно IntegrationClient Test, отображена информация из Integration Client

## Описание полученной информации

1. **DeviceRegistry** – Мета. Возможности сервера. Если выделить первую строку, то в нижней части окна отобразится подробная информация о мете:

<DeviceRegistry FS\_Version="1.0.0.0" clsid="Computer">

 <class clsid="CompositeDevice">

 <parent clsid="DevicePanel"/>

 <param name="HideInTree" type="Bool" value="1"/>

 <param name="Icon" type="String" value="Device\_Device"/>

 <param name="DeviceClassName" type="String" value="Сложное конечное устройство"/>

 </class>

…

<drv name="Компьютер" id="F8340ECE-C950-498D-88CD-DCBABBC604F3" child\_count="0" clsid="Computer" options="Placeable" maxZoneCardinality="0" minZoneCardinality="0" addrGroup="-1" parentInAddr="0" shortName="Компьютер" acr\_enabled="0" acr\_from="0" acr\_to="0" ar\_enabled="0" ar\_from="0" ar\_to="0" ar\_no\_addr="1" cat="0" caseCnt="0" baseType="-1">

 <propInfo name="QueryInterval" type="Int" default="500" editType="pkText" caption="Пауза между опросами,мс" helpIndex="0" hint="Пауза между опросом устройств в миллисекундах" paramID="0" min="100" max="1000" tslen="1" hidden="0" showOnlyInState="0" delay="0" imageIndex="0" RW="0" off="0" rawType="0" shiftInMemory="0" multiplier="0"/>

 <state id="-1" name="Мониторинг устройства отключен" code="Disabled" class="5" type="Common" CanResetOnPanel="0" AffectedParent="0" AllowIFBlock="0" SuperState="0" affectChildren="1" manualReset="0" primaryState="0"/>

 <state id="-2" name="Не удалось включить устройство" code="InitFailed" class="2" type="Common" CanResetOnPanel="0" AffectedParent="0" AllowIFBlock="0" SuperState="0" affectChildren="1" manualReset="0" primaryState="0"/>

 <state id="-3" name="Устройство инициализируется" code="Init" class="5" type="Common" CanResetOnPanel="0" AffectedParent="0" AllowIFBlock="0" SuperState="0" affectChildren="1" manualReset="0" primaryState="0"/>

 <state id="-4" name="Устройство в списке обхода" code="Mute" class="4" type="Common" CanResetOnPanel="0" AffectedParent="0" AllowIFBlock="0" SuperState="0" affectChildren="1" manualReset="0" primaryState="0"/>

 <state id="-5" name="Ключ защиты отсутствует" code="StopByHASP" class="5" type="Common" CanResetOnPanel="0" AffectedParent="0" AllowIFBlock="0" SuperState="0" affectChildren="0" manualReset="0" primaryState="0"/>

 </drv>

…

</DeviceRegistry>

**Краткое описание**:

<class> - класс (группа устройств) Описание класса устройств. Реализует правила подключения устройств друг к другу;

<drv> - описание драйвера устройства;

**name** – имя,

**shortName** – краткое имя,

**id** – GUID драйвера.

<propInfo> статическое свойство, доступное устройству, принадлежащему этому драйверу;

<paramInfo > динамические параметры, доступные устройству, принадлежащему этому драйверу;

**name** – строковый идентификатор,

**type** – тип,

**caption** – название,

**hint** – описание,

**default** – значение по умолчанию,

**min** – минимальное значение,

**max** – максимальное значение.

<state> состояние, доступное устройству этого драйвера;

**id** – идентификатор,

**name** – название,

**code** – строковый идентификатор,

**class** – класс состояние (число) (тревога/пожар – 0, внимание – 1, …),

**type** – тип (подсистема) (пожарное, охранное, общее, …).

1. **Config** – Конфигурация. Настройка оборудования.

<config FS\_Version="1.0.0.0">

 <drv idx="0" name="Компьютер" id="F8340ECE-C950-498D-88CD-DCBABBC604F3"/>

 <drv idx="1" name="COM порт (V1)" id="{0695ADC6-4D28-44D4-8E24-7F13D91F62ED}"/>

…

 <zone idx="0" no="10006" name="1ЭТ.КОРИДОР ЛВ.КРЫЛО" idZones="7556">

 <shape id="-2093943748"/>

 <param name="DB$IDZones" type="Int" value="7556"/>

 <param name="DB$IDParentZones" type="Int" value="0"/>

 <param name="AutoSet" type="Int" value="0"/>

 <param name="FireDeviceCount" type="Int" value="1"/>

 <param name="Skipped" type="Int" value="0"/>

 <param name="DB$IDSecObj" type="Int" value="750"/>

 <param name="ExitTime" type="SmallInt" value="0"/>

 <param name="ZoneType" type="Int" value="0"/>

 <param name="Delay" type="Int" value="0"/>

 <param name="GuardZoneType" type="Int" value="0"/>

 </zone>

…

 <part no="0" id="1" gid="16" name="Пользователь 1" type="guarduser">

 <PinZ pidz="10175"/>

 <param name="UnSetZone" type="Bool" value="1"/>

 <param name="DeviceUID" type="String" value="2ff457ac-b386-431d-9e1b-0550da4f04a4"/>

 <param name="Password" type="String" value="1"/>

 <param name="SetZone" type="Bool" value="1"/>

 <param name="KeyTM" type="String" value="000000000000"/>

 </part>

…

 <dev drv="0" addr="0">

 <prop name="INT$DEV\_GUID" type="String" value="b9293f8d-f816-4865-9d3c-008b0b77feab"/>

 <dev drv="74" addr="0">

 <prop name="BaudRate" value="4"/>

 <prop name="INT$DEV\_GUID" type="String" value="6a5ab2b8-1bd6-468c-8020-5ac39e8465ed"/>

 <dev drv="76" addr="1">

 <prop name="Address" value="34"/>

 <prop name="INT$DEV\_GUID" type="String" value="868f75c1-3ef7-4eca-9d2c-3473fe52b543"/>

 <dev drv="13" addr="1" disabled="1">

 <prop name="INT$ShapeIDList" value="-520588356;-1654555700"/>

 <prop name="INT$DEV\_GUID" type="String" value="a2530c14-1eca-4f6a-b029-34555d5c3f6e"/>

 <shape id="-520588356"/>

 <shape id="-1654555700"/>

 <dev drv="42" addr="257">

 <prop name="INT$ShapeIDList" value="1096905448;197156432;-1331213152;-641430275;173744928;-63240740;1307626085"/>

 <prop name="INT$DEV\_GUID" type="String" value="41c66ce4-a22a-45fa-a22f-a6b2a20f7fc0"/>

 <inZ idz="92"/>

…

</config>

**Краткое описание**:

<drv> - драйвер устройства, доступный в системе;

**idx** - идентификатор внутри XML (если у нас появляются новые устройства/приборы он будет меняться) используется в описании устройства (аттрибут drv),

**id** - GUID-драйвера, он уникальный, можете к нему привязываться,

**name** - полное имя драйвера устройства.

<zone> - зона;

**idx** - идентификатор внутри XML (используется для привязки устройства к зоне тег InZ),

**no** - номер зоны,

**name** - описание зоны.

<param> - разные параметры зоны: тип зоны, задержки, ....;

<dev> - иерархические теги описания дерева устройств;

**drv** - идентификатор драйвера (idx),

**addr** - адрес устройства (если это конечное устройство то значение двухбайтное, старший байт - номер шлейфа).

<prop name="INT$DEV\_GUID" type="String" value="\_\_\_\_\_"/> - GUID устройства - используется во всей системе, в частности в событиях;

<inZ idz="\_\_\_"/> - ID зоны (idx). У датчиков одна зона, у исполнительных устройств может быть много.

1. **States** – изменения состояний устройств.

<?xml version="1.0" encoding="windows-1251"?>

<States>

 <state deviceUID="a2530c14-1eca-4f6a-b029-34555d5c3f6e" devicePath="0\0\0" deviceText="Рубеж-2AM 1" add="1" stateName="Мониторинг устройства отключен" stateID="-1" stateCode="Disabled" stateClass="5" subSystem="0"/>

…

</States>

Содержит несколько <state> в которых:

deviceUID – GUID устройства;

devicePath – путь к устройству в дереве устройств;

deviceText – название устройства;

stateName – название состояния;

stateID – идентификатор состояния;

stateCode – строковый идентификатор состояния;

stateClass – класс состояния;

subsystem – подсистема (0 – общая, 1 – пожарная, 2 - охранная);

1. **Params – изменения параметров устройств.** Например, дым, пыль**.**

<?xml version="1.0" encoding="windows-1251"?>

<Params Generation="1" IsDelta="0">

 <Device UID="6a5ab2b8-1bd6-468c-8020-5ac39e8465ed" Path="0">

 <param name="Config$LastParamReaded" type="Int" value="-1"/>

 …

 </Device>

…

</Params>

Содержит несколько <Device> в которых описаны все параметры, принадлежащие устройству:

UID – GUID устройства;

Path - путь к устройству в дереве устройств;

<param> - параметр:

**Name** – строковый идертификатор параметра,

**type** – тип параметра,

**value** – значение.

1. **Events** – события. Любые события, произошедшие в системе.

<?xml version="1.0" encoding="windows-1251"?>

<Events>

 <event IDEvents="2092" IDSubSystem="2" DT="41649,5530787037" SysDT="41649,5530872917" IDTypeEvents="6" EventDesc="Корпус закрыт" IDDevices="" IDDevicesSource="2ff457ac-b386-431d-9e1b-0550da4f04a4" Device="" DeviceSource=" Рубеж-2ОП 2" IDZones="" Zone="" UserInfo="" HasEventAttr="1" EventAttr=""/>

…

</Events>

Содержит несколько <event>, которые включают в себя:

IDEvents – идентификатор события;

IDSubSystem – подсистема (0 – общая, 1 – пожарная, 2 - охранная);

DT – дата/время в приборе;

SysDT – системная дата/время (время получения);

IDTypeEvents – класс события (тревога, внимание, неисправность, …);

EventDesc – описание события;

IDDevices – GUID устройства;

IDDevicesSource – GUID прибора;

Device – описание устройcтва;

DeviceSource – описание прибора;

IDZones – идентификатор зоны;

Zone – описание зоны;

UserInfo – информация о пользователе.

## Запросы обрабатываемые сервером HTTP

Встроенный HTTP сервер принимает следующие запросы (Таблица 1) В окне HTTP Server Test их можно выбрать из раскрывающегося списка или набрать самостоятельно, а затем нажать кнопку **Send** (Рис. 7):



Рис. 7 Окно IntegrationClient Test, запросы к серверу

Таблица 1

|  |  |
| --- | --- |
| **Шаблоны запросов** | **описание** |
| Ping | пинг |
| Config | текущая конфигурация |
| DeviceRegistry | мета |
| States | актуальные состояния устройств |
| DeviceStates:FindBy:Param | Состояния устройства |
| FindBy – тип параметра устройства (“GUID”/”Path”)Param – параметр для поиска устройства (в зависимости от FindBy) |
| LastEventID | идентификатор последнего события |
| Events:ID:0 | получение событий начиная с идентификатора ID |
| ID – начальный идентификатор события |
| Params:N | параметры устройств начиная с генерации N |
| N – номер генерации |
| Zone:Guard:Set:N | постановка зоны с номером N на охрану |
| N – номер зоны |
| Zone:Guard:Unset:N | снятие зоны с номером N с охраны |
| N – номер зоны |
| SCUDConfig | Получение полного списка пользователей строкой в формате JSON |
| ResetFire | сброс пожаров |
| ResetAlarm | сброс тревог (в охранных зонах) |
| Device:FindBy:Param:Command | управление устройствами |
| FindBy – тип параметра устройства (“GUID”/”Path”)Param – параметр для поиска устройства (в зависимости от FindBy)Command – команда к устройству “Control$Run”, “Control$Stop”, … – управление устройством* Все доступные команды управления доступны в мете в описании параметров драйвера, начинающиеся с префикса “Control$”

 “Disable” – отключить устройство “Enable” – задействовать устройство |
| Scenario:N:Command | управление сценариями |
| N – номер сценарияCommand – команда (“Run”, “Stop”, “Unblock”, “Block”) |

# Интеграция СКУД через REST API

В ПО «Охранно-пожарная сигнализация Firesec**»** существует возможность обмена данными по персоналу с ПО «1С». Для этого необходимо запустить приложение для интеграции FS\_IntegrationClient, в котором реализованы API методы для добавления, удаления, изменения и просмотра сведений о сотрудниках.

**Запрос GET /api/personnel/employee.**

 Для того что бы начать работу необходимо по адресу http://127.0.0.1:8081(для локального подключения) отправить запрос GET.

Path: /api/personnel/employee.

Header: Accept = application/json; accept-charset = utf-8.

Параметры: Action, Name, Code, Folder, Sex, BirthDay, ApiKey, TokenCode, TokenFormat, StrJSON, ID, PhotoLink где:
**Action** – команда управления: SCUDConfig – получение конфигурации скуд в формате JSON, Add - добавление, Delete – удаление, View - просмотр, Change – изменение, AddToken – добавление идентификатора, DelToken – удаление идентификатора, ViewPhoto- получение фотографии пользователя, AddJSON – добавление сотрудника строкой JSON, DelJSON – удаление сотдруника JSON, AddTokenJSON – добавление идентификатора строкой JSON, DelTokenJSON – удаление идентификатора.
В зависимости от команды управления необходим разный набор параметров(описано ниже).

Name – ФИО сотрудника, должно задаваться в формате - Фамилия\_Имя\_Отчество.
Code – табельный номер сотрудника, должен быть уникальным.
Folder – группа в которой создается сотрудник, подгруппы разделяются через знак «/».Пример: КБ1/ОТД/.
Sex – пол сотрудника.
BirthDay – дата рождения сотрудника.
TokenCode – код идентификатора.
TokenFormat – формат идентификатора.
StrJSON – строка в формате JSON содержащая информацию о сотруднике или идентификаторе.
PhotoLink – ссылка на фотографию пользователя в базе данных.
ID – внутренний id объекта.

ApiKey – ключ формируемый в FS\_IntegrationClient для проверки безопасности. 



Для того что бы иметь доступ ко всем группам (папкам), поле папка оставляем пустым.

**Параметры для команд:**

**SCUDConfig** – вызывается без параметров.

Для команды **Add** – обязательны три параметра Name, Code, APIKey. Если Folder будет пустой то пользователь будет добавлен в корневую папку, если параметр BirthDay будет пустой, то дата рождения будет установлена по умолчанию, если параметр Sex будет пустой, то он будет установлен по умолчанию мужской.

Для команды **Delete** – необходимы три параметра Name, Code, APIKey.

Для команды **View** – необходимы параметры Name, Code, APIKey. Поиск ведется по параметрам Name, Code. Должен быть заполнен хотя бы 1 из этих параметров. При нахождении будут показываться все данные о пользователе. При заполнении поля Name фамилию имя или отчество можно заменить на символ “\*” и тогда при поиске будут учитываться только заполненные поля. Для получения всего списка пользователей поля Name и Code надо оставить пустыми.

Для команды **ViewPhoto** – необходим параметр PhotoLink, значение которого можно получить либо из полного запроса базы SCUDConfig, либо из конкретного запроса пользователя ViewPhoto и параметр APIKey.

Для команды **Change** – необходимы параметры Name, Code, APIKey. Ищется пользователь по табельному номеру и ему присваиваются новые ФИО заданные в параметре Name.

Для команды **AddToken** – необходимы параметры Name, Code, TokenCode, TokenFormat, APIKey.

Для команды **DelToken** – необходимы параметры TokenCode, TokenFormat, APIKey.

Для команды **AddJSON** – необходимы параметры StrJSON, Folder, APIKey.

Для команды **DelJSON** – необходимы параметры ID, APIKey.

Для команды **AddTokenJSON** – необходимы параметры StrJSON, ID, APIKey.

Для команды **DelTokenJSON** – необходимы параметры ID, APIKey.

Для команды **AddFolderJSON** – необходимы параметры StrJSON, Folder, APIKey.

Для команды **DelFolderJSON** – необходимы параметры ID, APIKey.

**Примеры:**

* Получение конфигурации SCUDConfig http://127.0.0.1:8081/api/personnel/employee?Action=SCUDConfig&apiKey=7u2QkaX1QZiKZvcOo94UAN
* Добавление сотрудника *http://127.0.0.1:8081/api/personnel/employee?Action=Add&Name=Ivanov\_Ivan\_Ivanovich&Code=555&Folder=Сенсор/КБ1&apiKey=8rYs13A5TJVyV9ZYu9VVAN&Sex=Мужской&BirthDay=11.02.1995*
* Удаление сотрудника *http://127.0.0.1:8081/api/personnel/employee?Action=Delete&Name=Ivanov\_Ivan\_Ivanovich&Code=555&Folder=Сенсор/КБ1&apiKey=1HZxtvPjSatb4OSBZJ7VAJ*
* Просмотр сотрудника
*http://127.0.0.1:8081/api/personnel/employee?Action=View&Name=Ivanov\_\*\_\*&apiKey=8rYs13A5TJVyV9ZYu9VVAN.* Результатом выполнения будет строка в формате JSON содержащая информацию о всех пользователях с фамилией Ivanov.
* Изменения сотрудника *http://127.0.0.1:8081/api/personnel/employee?Action=Change&Name=Ivanov\_Ivan\_Olegovich&apiKey=8rYs13A5TJVyV9ZYu9VVAN&Code=555.* В результате сотруднику с табельным номером 555 будут присвоены ФИО Ivanov\_Ivan\_Olegovich.

**Запрос /api/personnel/reports/**

В стандартных конфигурациях **1 С: 8.x**, **1 C Зарплата и Управление Персоналом** (ЗУП) и **Комплексная конфигурация** есть документ «Табель учета рабочего времени», который имеет печатную форму Т-13. Для формирования этого документа данные «сотрудник – день часы» запрашиваются ПО 1С из FireSec.

Для того что бы начать работу необходимо по адресу http://127.0.0.1:8081(для локального подключения) отправить запрос GET.

Path: /api/personnel/reports.

Header: Accept = application/json; accept-charset = utf-8.

Параметры: Journal, Name,Code,Folder, From, To, apiKey, где:
Journal – вид запрашиваемого отчета, **simplePassLog** – журнал проходов, **currentPositions** – местонахождение, **hoursWorkedDetails** – отработанное время, **hoursWorked** – отчеты Т12/Т13.
Name – ФИО сотрудника, должно задаваться в формате - Фамилия\_Имя\_Отчество.
Code – табельный номер сотрудника, должен быть уникальным.
Folder – группа в которой создается сотрудник, подгруппы разделяются через знак «/».Пример: КБ1/ОТД/.
From – начало периода.
To – конец периода.
apiKey – ключ формируемый в FS\_IntegrationClient для проверки безопасности.

# Описание работы OPC-сервера.

Для запуска OPC сервера необходимо первоначально regOPCserver.bat который находится *C:\ProgramFiles(x86)\FireSec3\* *regOPCserver.bat*

При запуске FS\_IntegrationClient на основе конфигурации создает дерево тегов состоящее из устройств, зон и сценариев.

Если значение тега равно 0(False) это состояние отключено, если значение равно 1(True) это состояние включено. **Состояния сценариев не отображаются через теги!**

**Устройства**

У каждого устройства в дереве тегов добавлены все его состояния.



Для управления устройствами можно использовать только следующие теги:

При помощи тега «HardwareIgnore» устройство можно добавлять или удалять из обхода. При ручном вводе значения 1(True) устройство добавляется в обход, при вводе 0(False) убирается из обхода.

Через тег «Run» или тег заканчивающийся на «On» (RMOn, MPT\_On) запускаются или останавливаются исполнительные устройства. При вводе 1(True) - выполняется команда включения, при вводе 0(False) выполняется команда выключения.

Через тег «Auto\_Off» управляется автоматика. При вводе 1(True) – включается ручной режим, при вводе 0(False) включается автоматический режим.

Через тег «ClapanOn1e» управляется МДУ-1. При вводе 1(True) – открыто, при вводе 0(False) закрыто.

Через теги «SHUV\_Start» и «SHU\_Start» управляются ШУВ и ШУ соответственно. При вводе 1(True) – выполняется команда включения, при вводе 0(False) – выполняется команда выключения.

Изменение других тегов ни к чему не приведет, они служат для отображения состояния устройства.

**Зоны**

У зон добавлены все классы состояний:



Для управления зонами можно использовать только следующие теги:

По тегу «Mute» зона добавляется и удаляется из обхода. 1(True) – добавление зоны в обход, 0 (False) – удаление из списка обхода.

По тегу «OnGuard» зона ставится и снимается с охраны. 1(True) – постановка на охрану, 0 (False) – снятие с охраны.

Изменение других тегов ни к чему не приведет, они служат только для отображения состояния зоны.