



HTTP API ПК Интеллект

1. Общие сведения о HTTP API	3
2. Версия продукта	3
3. Карта	4
3.1 Получение списка карт	4
3.2 Информация об одной карте	4
3.3 Список слоёв для выбранной карты	5
3.4 Информация о конкретном слое	6
3.5 Фоновый рисунок слоя	6
3.6 Список точек на слое	6
3.7 Информация об отдельной точке на слое	9
4. Классы объектов	9
4.1 Список классов объектов, которые существуют на сервере	9
4.2 Отдельный класс объектов	10
4.3 Список состояний для определённого класса объектов	10
4.4 Информация о конкретном состоянии	10
4.5 Получение иконки для определённого состояния	11
4.6 Список сообщений для определённого класса объектов	11
4.7 Информация об отдельном объекте	13
4.8 Состояние отдельного объекта	13
4.9 Список доступных действий с объектом, находящимся в определённом состоянии	13
5. Получение событий	14
5.1 Получение событий видеоподсистемы блоками	15
6. Отсылка команд на сервер	17
7. Макрокоманды	17
8. Видео	18
8.1 Запрос конфигурации	19
8.2 Запрос видео	20
8.3 Формат основного потока	21
8.3.1 Управление записью	22
8.3.2 Постановка и снятие с охраны камеры	23
8.3.3 Управление телеметрией	23
8.3.4 Работа с архивом	24
9. Нотификация	29
10. Звук	31
10.1 Получение живого звука	31
10.2 Проигрывание звука из архива	33
10.3 Отправка живого звука	33

Общие сведения о HTTP API

Программно HTTP API предоставляется модулем web2 (*Веб-сервер 2.0*).

Примечание.

См. Руководство Администратора, раздел Настройка Сервера для подключения Клиентов с помощью модуля Веб-сервер 2.0.

Port – порт.

/somecontext – опциональный веб-контекст, в котором работает приложение. Это контекст веб-приложения.

Таким образом можно на одном домене иметь несколько систем:

www.example.com/sistema1/

www.example.com/videosistema23/

www.example.com/a/

Причем этот контекст может быть более сложным:

www.example.com/redirects/toitvwebserver/firstsystem/

www.example.com/redirects/toitvwebserver/secondsystem/

www.example.com/redirects/toitvwebserver/sauna/

Далее описание будет опускаться там, где действие запроса понятно из контекста.

Внимание!

URL, id объектов и расширения файлов чувствительны к регистру.

Примечание.

Дата и время везде используется в формате RFC3339, подробнее см. <http://www.ietf.org/rfc/rfc3339.txt>

Версия продукта

Для идентификации сервера можно использовать URL

`http://example.com:[port][/somecontext]/product/version`

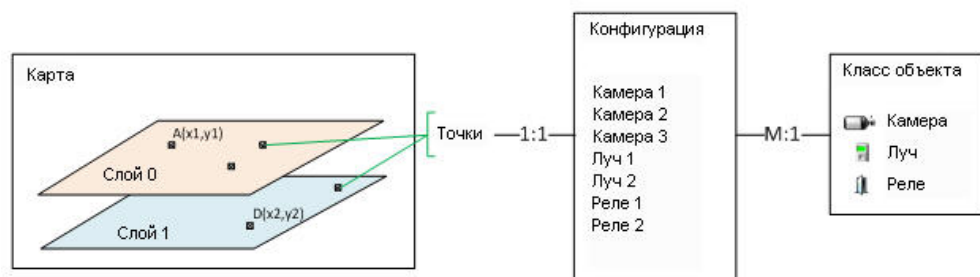
Если в ответ приходит text/plain строка типа

Intellect 4.8.4

Это означает, что сервер поддерживает протокол, описанный в данном документе. Строка может меняться в зависимости от версии продукта. Это сделано для того, чтобы

различать два схожих по функциональным возможностям, но разных по протоколу веб-сервера в разных продуктах.

Карта



На Сервере может быть создано несколько карт. Каждая карта может содержать один и более слоёв. На каждом слое расположены точки. Каждая точка связана с одним из объектов конфигурации.

Конфигурация – это объекты ПК *Интеллект*. Каждый объект является объектом определённого класса. Каждый объект имеет одно состояние и список действий, которые можно с ним производить.

Класс объекта описывает его вид (значки), возможные состояния и возможные действия с объектом в каждом из состояний .

Получение списка карт

Карт может быть 0 и более.

[http://example.com:\[port\]\[somecontext\]/secure/kartas/](http://example.com:[port][somecontext]/secure/kartas/)

Пример ответа:

```
<kartas>
<karta>
<id>plan</id>
<name>This is plan of a building</name>
</karta>
<karta>
<id>site</id>
<name>This is site around the building</name>
</karta>
</kartas>
```

Информация об одной карте

plan – id карты.

http://example.com:[port][/somecontext]/secure/kartas/**plan**/

Пример ответа:

```
<karta>
<id>plan</id>
<name>This is plan of a building</name>
</karta>
```

Список слоёв для выбранной карты

plan – id карты.

Слоёв может быть 1 и более.

http://example.com:[port][/somecontext]/secure/kartas/**plan**/layers/

Пример ответа:

```
<layers>
<layer>
<height>1000</height>
<id>base</id>
<mapId>plan</mapId>
<name>Base layer for plan</name>
<width>1000</width>
<zoomDef>1.0</zoomDef>
<zoomMax>4.0</zoomMax>
<zoomMin>0.25</zoomMin>
<zoomStep>0.25</zoomStep>
</layer>
</layers>
```

Описание параметров:

Height – высота картинки слоя в пикселях;

Width – ширина картинки слоя в пикселях;

zoomMin – минимальный масштаб картинки;

zoomMax – максимальный масштаб картинки;

zoomStep – шаг увеличения масштаба при zoom in и zoom out;

zoomDef – масштаб по-умолчанию.

Пример. Пусть ширина картинки равна 100 пикселям. Тогда ширина для масштаба 0,25 будет $100 * 0,25 = 25$ пикселей.

Информация о конкретном слое

Описание параметров см. в разделе [Список слоёв для выбранной карты](#).

base – id слоя.

[http://example.com:\[port\]\[somecontext\]/secure/kartas/plan/layers/base/](http://example.com:[port][somecontext]/secure/kartas/plan/layers/base/)

Пример ответа:

```
<layer>
  <height>1000</height>
  <id>base</id>
  <mapId>plan</mapId>
  <name>Base layer for plan</name>
  <width>1000</width>
  <zoomDef>1.0</zoomDef>
  <zoomMax>4.0</zoomMax>
  <zoomMin>0.25</zoomMin>
  <zoomStep>0.25</zoomStep>
</layer>
```

Фоновый рисунок слоя

[http://localhost:8080/server-1.0/secure/kartas/plan/layers/base/image.\[png|jpg\]](http://localhost:8080/server-1.0/secure/kartas/plan/layers/base/image.[png|jpg])

В ответ приходит изображение в формате png или jpg.

На запрос JPG, Jpg, JPEG, PNG будет возвращаться ошибка 404

Список точек на слое

[http://example.com:\[port\]/\[somecontext\]/secure/kartas/plan/layers/base/points/](http://example.com:[port]/[somecontext]/secure/kartas/plan/layers/base/points/)

id – совпадает с id объекта из конфигурации. Id не обязательно всегда равен CAM:1. Следует воспринимать id как строку.

Координатная сетка привязана к слою следующим образом:



Т.е. x и y не могут быть отрицательными, но могут быть дробными.

Пример ответа:

```
<points>
  <point>
    <id>CAM:1</id>
    <layerId>base</layerId>
    <mapId>plan</mapId>
    <x>100.0</x>
    <y>100.0</y>
  </point>
  <point>
    <id>CAM:2</id>
    <layerId>base</layerId>
    <mapId>plan</mapId>
    <x>200.0</x>
```

```
<y>200.0</y>
</point>
<point>
  <id>GRAY:1</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>300.0</x>
  <y>300.0</y>
</point>
<point>
  <id>GRAY:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>400.0</x>
  <y>400.0</y>
</point>
<point>
  <id>GRELE:1</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>500.0</x>
  <y>500.0</y>
</point>
<point>
  <id>GRELE:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>600.0</x>
  <y>600.0</y>
```



```
</point>
</points>
```

Информация об отдельной точке на слое

[http://example.com:\[port\]/somecontext/secure/kartas/plan/layers/base/points/CAM:2](http://example.com:[port]/somecontext/secure/kartas/plan/layers/base/points/CAM:2) – запрос информации о точке, соответствующей камере с идентификатором 2.

Пример ответа:

```
<point>
  <id>CAM:2</id>
  <layerId>base</layerId>
  <mapId>plan</mapId>
  <x>200.0</x>
  <y>200.0</y>
</point>
```

Классы объектов

Список классов объектов, которые существуют на сервере

[http://example.com:\[port\]/somecontext/secure/objectClasses](http://example.com:[port]/somecontext/secure/objectClasses)

Пример ответа:

```
<objectClasses>
  <objectClass>
    <id>GRELE</id>
  </objectClass>
  <objectClass>
    <id>USERS</id>
  </objectClass>
  <objectClass>
    <id>CAM</id>
  </objectClass>
  <objectClass>
```

```
    <id>RIGHTS</id>
  </objectClass>
  <objectClass>
    <id>GRAY</id>
  </objectClass>
</objectClasses>
```

Отдельный класс объектов

[http://example.com:\[port\]/\[somecontext\]/secure/objectClasses/GRELE/](http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/)

Пример ответа:

```
<objectClass>
  <id>GRELE</id>
</objectClass>
```

Список состояний для определённого класса объектов

[http://example.com:\[port\]/\[somecontext\]/secure/objectClasses/GRELE/states/](http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/states/) - получить список состояний для класса объектов **Реле**.

Пример ответа:

```
<states>
  <state>
    <id>off</id>
  </state>
  <state>
    <id>on</id>
  </state>
  <state>
    <id>disabled</id>
  </state>
</states>
```

Информация о конкретном состоянии

`http://example.com:[port]/[somecontext]/secure/objectClasses/[ObjectClass]/states/[State]/`

Пример:

`http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/states/off/` - получение информации о состоянии OFF класса объектов **Реле**.

Пример ответа:

```
<state>
<id>off</id>
</state>
```

Получение иконки для определённого состояния

`http://example.com:[port]/[somecontext]/secure/objectClasses/[ObjectClass]/states/[State]/image.png`

Пример:

`http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/states/off/image.png` - получение иконки для состояния OFF класса объектов **Реле**.

В ответ приходит изображение в формате png:



Список сообщений для определённого класса объектов

GET

`http://example.com:[port]/[somecontext]/secure/objectClasses/GRELE/events/` – получение списка сообщений для класса объектов **Реле**.

Пример ответа:

XML

```
<events>
  <event>
    <id>23</id>
    <sid>grele.disable</sid>
    <description>Disable rele</description>
  </event>
  <event>
    <id>24</id>
    <sid>grele.enable</sid>
```

```
    <description>Enable rele</description>
  </event>
<baseObject>
  <CAM>
    <id>CAM:1</id>
    <name>--</name>
    <state>
      <id>disconnected</id>
    </state>
  </CAM>
  <GRELE>
    <id>GRELE:2</id>
    <name>[GRELE] 2</name>
    <state>
      <id>off</id>
    </state>
  </GRELE>
  <GRELE>
    <id>GRELE:1</id>
    <name>Relay 1</name>
    <state>
      <id>disabled</id>
    </state>
  </GRELE>
  <GRAY>
    <id>GRAY:2</id>
    <name> 2</name>
    <state>
      <id>alarmed</id>
    </state>
  </GRAY>
</baseObject>
```

```
</state>
</GRAY>
</baseObjects>
```

Информация об отдельном объекте

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/) – получение информации об объекте **Луч** с идентификатором 2.

Пример ответа:

```
<GRAY>
  <id>GRAY:2</id>
  <name> 2</name>
  <state>
    <id>alarmed</id>
  </state>
</GRAY>
```

Состояние отдельного объекта

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/state/](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/state/) – получение состояния объекта **Луч** с идентификатором 2.

Пример ответа:

```
<state>
  <id>alarmed</id>
</state>
```

Список доступных действий с объектом, находящимся в определённом состоянии

Список действий запрашивается не по классу объекта, а берётся из контекста конкретного объекта, т.к. возможны различные права пользователя на объекты одного и того же класса.

Работа с полученным списком описана в разделе [Отсылка команд на сервер](#).

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/state/actions/](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/state/actions/) – получение списка доступных действия для объекта **Луч** с идентификатором 2.

Пример ответа:

```
<actions>
  <action>
```

```
    <description>Disarm ray</description>
    <id>ray.disarm</id>
</action>
<action>
    <description>Confirm alarm</description>
    <id>ray.confirm</id>
</action>
</actions>
```

Если состояние объекта не предусматривает никаких действий, то xml будет таким:

```
<actions/>
```

Получение событий

Соединение не разрывается и события приходят бесконечно.

action – тип события. Возможные значения: create, delete, update.

Все поля ниже опциональны:

objectId – id объекта, от которого приходит событие (обязательно приходит с update, delete, create).

state – id нового состояния объекта (обязательно приходит в create. Если состояние не изменилось, то в событии update состояния не будет).

x, y – новые координаты, если изменились.

Запрос:

```
http://example.com:[port][[/somecontext]/secure/feed/
```

Примеры ответа:

```
<message>
    <action>update</action>
    <objectId>CAM:1</objectId>
    <state>disconnected</state>
</message>
```

```
<message>
```

```
<action>state</action>
<objectId>CAM:1</objectId>
<x>10.0</x>
<y>123.9</y>
</message>
```

```
<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <state>connected</state>
  <x>300.8</x>
  <y>670</y>
</message>
```

```
<message>
  <action>state</action>
  <objectId>CAM:1</objectId>
  <x>100</x>
  <y>100</y>
</message>
```

```
<message>
  <action>ping</action>
</message>
```

Получение событий видеоподсистемы блоками

[http://example.com:\[port\]\[somecontext\]/secure/events/](http://example.com:[port][somecontext]/secure/events/)

Параметры:

from – Самая старая дата промежутка поиска сообщений. (2012-12-27T15%3A19%3A16.000%2B04%3A00)

to – Самая последняя дата промежутка поиска сообщений. (2012-12-27T15%3A19%3A16.000%2B04%3A00)

count – максимальное количество сообщений в ответе [1, 200]. По-умолчанию 20. Сервер может вернуть чуть больше, если сообщений в базе данных осталось мало.

objectId – id объекта (CAM:1, GRAY:5 и т.д). Если параметр не задан, то возвращаются события всех объектов.

Коды возврата:

200 - ОК

400 - неверный параметр (формат даты, например)

500 - ошибка

503 - ошибка соединения с ядром

504 - таймаут (ядро не вернуло данные в течение 2000 миллисекунд)

Примеры ответа

XML:

```
<events>
  <event>
    <description> </description>
    <id>{E56B09A0-1A50-E211-840E-005056C00008}</id>
    <objectId>CAM:1</objectId>
    <ts>2012-12-27T15:43:27+04:00</ts>
  </event>
  <event>
    <description> </description>
    <id>{4482F63F-1A50-E211-840E-005056C00008}</id>
    <objectId>CAM:1</objectId>
    <ts>2012-12-27T15:40:50+04:00</ts>
  </event>
  <event>
    <description> </description>
    <id>{35D4BE3E-1750-E211-840E-005056C00008}</id>
    <objectId>CAM:1</objectId>
    <ts>2012-12-27T15:19:16+04:00</ts>
  </event>
</events>
```


JSON:

```
[ {
  "id" : "{E56B09A0-1A50-E211-840E-005056C00008}",
  "description" : " ",
  "ts" : "2012-12-27T15:43:27.000+04:00",
  "objectId" : "CAM:1"
}, {
  "id" : "{4482F63F-1A50-E211-840E-005056C00008}",
  "description" : " ",
  "ts" : "2012-12-27T15:40:50.000+04:00",
  "objectId" : "CAM:1"
}, {
  "id" : "{35D4BE3E-1750-E211-840E-005056C00008}",
  "description" : " ",
  "ts" : "2012-12-27T15:19:16.000+04:00",
  "objectId" : "CAM:1"
} ]
```

Отсылка команд на сервер

PUT

[http://example.com:\[port\]/\[somecontext\]/secure/configuration/GRAY:2/state/actions/disarm/execute](http://example.com:[port]/[somecontext]/secure/configuration/GRAY:2/state/actions/disarm/execute) - пример отсылки на сервер команды снятия с охраны Луча с идентификатором 2.

Макрокоманды

В разделе:

- [Получение списка макрокоманд](#)
- [Получение параметров макрокоманд](#)
- [Запрос на выполнение макрокоманды на сервере](#)

Макрокоманды (макросы) – это некоторая предопределённая последовательность реакций на определённые события. Макрокоманды создаются на сервере и имеют ID и

название. Они похожи на действия с объектами, но не привязаны к объекту.

Получение списка макрокоманд

GET

`http://example.com:[port]/[somecontext]/secure/actions/`

Пример ответа:

```
<actions>
  <action>
    <description>Start recording by all cameras</description>
    <id>macro2</id>
  </action>
  <action>
    <description>Disarm all zones</description>
    <id>1</id>
  </action>
</actions>
```

Получение параметров макрокоманд

Каких-либо дополнительных параметров у объекта нет. Можно ограничиться получением списка макросов.

GET

`http://example.com:[port]/[somecontext]/secure/actions/macro2/` - получение параметров макрокоманды с идентификатором macro2.

Пример ответа:

```
<action>
<description>Start recording by all cameras</description>
<id>macro2</id>
</action>
```

Запрос на выполнение макрокоманды на сервере

PUT

`http://example.com:[port]/[somecontext]/secure/actions/macro2/execute` – запрос на выполнение на сервере макрокоманды с идентификатором macro2.

Видео

Запрос конфигурации

GET

`http://example.com:[port]/[somecontext]/secure/video/config.properties?version=4.7.8.0&login=XXX&password=YYY`

Параметры:

- `version` – обязательное поле. Версия клиента (на случай смены протокола). Сейчас нужно посылать значение "4.7.8.0".
- `login` – необязательное поле. Логин.
- `password` – необязательное поле. Используется, если установлен доступ по паролю.

Особенности использования

В начале работы неизвестно, установлены ли пароль, логин и т.п. Поэтому в первый раз необходимо послать следующий запрос:

GET

`http://www.examplehost.com/[somecontext]/secure/video/config.properties?version=4.7.8.0`

В ответ сервер отправит текстовый файл `config.properties` следующего формата:

```
password.enabled=true  
login.enabled=true  
password.invalid=true#
```



Примечание.

Символ # является признаком конца конфигурационного файла.

После получения файла такого вида можно понять, что пароль установлен и пароль неправильный. Неправильный он потому, что в данном случае был послан пустой пароль и пустой логин.

Необходимо запросить у пользователя логин и пароль и снова отослать серверу запрос на конфигурацию:

GET

`http://www.examplehost.com/[somecontext]/secure/video/config.properties?version=4.7.8.0&login=XXX&password=YYY`

Если пароль правильный или доступ разрешен без пароля, то сервер в ответ вышлет конфигурацию в следующем виде:

```
password.enabled=true  
login.enabled=true  
password.invalid=false  
cam.0.id=2
```

```
cam.0.name=Face
cam.0.rights=11
cam.1.id=3
cam.1.name=Camera 3
cam.1.rights=11
cam.2.id=5
cam.2.name=Camera 5
cam.2.rights=11
cam.2.telemetry_id=1.1
cam.count=3#
```

password.invalid=false означает, что введён верный пароль.



Примечание.

Если разрешен доступ без пароля, то password.enabled=false, и вся нужная конфигурация будет получена с первого раза.

cam.count=3 – общее количество камер в присланной конфигурации (id начинается с нуля).

Для каждой из трёх камер необходимо получить данные из конфигурации.

cam.N.id – id камеры.

cam.N.name – название камеры.

cam.N.rights – права.

cam.N.telemetry_id – id телеметрии (может отсутствовать, если телеметрии нет, тогда необходимо скрывать элементы управления телеметрией).

cam.N.rights – определяет права (они проверяются на сервере, но чтобы не показывать пользователю лишних опций, доступны и на клиенте). Параметр представляет собой флаги. Если флаг проставлен, то элемент интерфейса следует показывать, если нет, то скрывать.

```
static final int RIGHT_VIEW = 0x1; // доступен просмотр живого видео (этот всегда проставлен в 1)
```

```
static final int RIGHT_CONTROL = 0x2; // управление (телеметрия, постановка и снятие с охраны)
```

```
static final int RIGHT_CONFIG = 0x4; // reserved
```

```
static final int RIGHT_HISTORY = 0x8; // доступ к архиву
```

Запрос видео

[http://example.com:\[port\]/\[somecontext\]/secure/video/action.do?version=4.7.8.0&sessionId=FC126734&cam.id=5&login=XXX&password=YYY](http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionId=FC126734&cam.id=5&login=XXX&password=YYY) - запрос видео для камеры с идентификатором 5.

cam.id – идентификатор камеры.

sessionId – любое значение.

Формат основного потока

Поток состоит из фреймов и сообщений.

Все даты посылаются в 24-часовом формате: dd.MM.yy kk:mm:ss. Например, 15.05.10 10:51:44 – 15 мая 2010 года 10 часов 51 минута 44 секунды

В ответе сразу после http заголовка присылаются три строчки:

```
sessionId=29101F1\nerrmsg=Error text message\nerrcode=100\n
```

Здесь:

- sessionId – id сессии. Это зарезервированный параметр, не используется.
- errmsg – сообщение об ошибке в текстовом виде;
- errcode принимает следующие значения:
 - 100 – ошибок нет
 - 101 – слишком много подключенных пользователей
 - 102 – неверный пароль (теоретически пароль может быть изменен в любой момент работы)
 - 103 – видеосервер не доступен
 - 104 – старая версия клиента. Обновите версию.

После этих трёх строчек (если код errcode=100) будет получен видеофрейм или сообщение.

Формат видеофрейма:

```
Frame\nsize=23978\ndelay=5243\nwidth=320\nheight=240\nfile.name=C:\VIDEO\13-05-10 19\1._01\ncolor=1\nfrm.total=500\nfrm.id=100\nfrm.time=15.05.10 10:51:44\n
```

```
format=1\n
```

```
byte[size-4] jpegdata
```

```
byte[4] jpegdataformat
```

Здесь:

- В последних 4-х байтах должно содержаться название: либо JPEG, либо JPG1. Другие форматы не поддерживаются.
- size – размер бинарного буфера с изображением
- delay – задержка между кадрами в миллисекундах.
- width – ширина в пикселях.
- height – высота в пикселях.
- file.name – имя файла архива. Используется при навигации по архиву.
- color – 0-черно-белый, 1-цвет
- frm.total – всего фреймов в файле архива (для живого видео этот параметр не важен)
- frm.id – порядковый номер в файле архива (для живого видео этот параметр не важен)
- frm.time – время возникновения фрейма
- format – всегда 1;
- jpegdata – буфер с фреймом JPEG;
- jpegdataformat – 4 байта с JPEG или JPG1.

Формат сообщения:

```
Msg=start\n
```

```
type=CAM\n
```

```
id=1\n
```

```
action=DISABLED\n
```

```
paramname=paramvalue\n
```

```
Msg=end\n
```

Здесь:

- Msg=start и Msg=end отмечают начало и конец сообщения соответственно.
- type – тип объекта, от которого пришло сообщение.
- id – идентификатор объекта.
- action – действие.
- paramname=paramvalue – набор параметров и их значений. Параметров может быть несколько и с разными названиями.

Управление записью

Начало записи

```
GET
```

```
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionId=29101F1&cam.id=1&target=CAM&targetid=1&command=REC&login=XXX&password=YYY
```

Окончание записи

GET

`http://example.com:[port]/somecontext/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=CAM&targetid=1&command=REC_STOP&login=XXX&password=YYY`

Здесь `targetid=cam.id`.

Постановка и снятие с охраны камеры

Постановка на охрану

GET

`http://example.com:[port]/somecontext/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=1&target=CAM&targetid=1&command=ARM&login=XXX&password=YYY`

Снятие с охраны

GET

`http://example.com:[port]/somecontext/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=CAM&targetid=1&command=DISARM&login=XXX&password=YYY`

Здесь `targetid=cam.id`.

Управление телеметрией

GET

`http://example.com:[port]/somecontext/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&target=PTZ&targetid=1.1&command=RIGHT&login=XXX&password=YYY&speed=2`

Параметры:

command – обязательный параметр. Может принимать следующие значения:

- RIGHT
- UP
- LEFT
- DOWN
- ZOOM_IN
- ZOOM_OUT
- GO_PRESET – перейти в определенный пресет.
- POINTMOVE – зуммирование выделенной точки на изображении (x, y).
- AREA_ZOOM – зуммирование выделенной области изображения (x,y,w,h).

speed – обязательный параметр. Скорость отработки команды (от 0 до 10). При управлении по сети из-за задержек лучше использовать низкие значения.

cam.id – обязательный параметр. Идентификатор камеры.

target – обязательный параметр. Всегда равно PTZ.

targetid – обязательный параметр. Id телеметрии, связанной с камерой (присылается в конфигурации SETUP).

preset – номер пресета (число). Обязательный параметр только для command=GO_PRESET. В остальных случаях его значение игнорируется.

x – координата x относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=POINTMOVE или command=AREAZOOM. В остальных случаях его значение игнорируется.

y – координата y относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=POINTMOVE или command=AREAZOOM. В остальных случаях его значение игнорируется.

w – ширина области зуммирования y относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=AREAZOOM. В остальных случаях его значение игнорируется.

h – высота области зуммирования относительно размера изображения. Может принимать значения от 0.0 до 1.0. Обязательный параметр только для command=AREAZOOM. В остальных случаях его значение игнорируется.

Работа с архивом

В разделе:

- Вход в архив - arc.enter
- Проигрывание одной записи архива - "arc.play"
- Непрерывное проигрывание записей архива - "arc.playnonstop"
- Переход на один фрейм или на одну запись назад - "arc.prev"
- Переход на один фрейм или на одну запись вперёд - "arc.next"
- Остановить проигрывание - "arc.stop"
- Получение списка записей

Поток из видеоархива присылается в таком же формате, как и живое видео.

Важные поля видеофрейма при работе с архивом:

file.name – значение этого поля нужно хранить для навигации внутри архива (покадровый просмотр, просмотр по записям и т.п.)

frm.total – сколько всего фреймов в записи

frm.id – id текущего фрейма в записи (начинается с нуля)

Информация о записях в архиве передаётся посредством Msg одним из следующих способов:

```
1. Msg=start\n
   type=CAM\n
   id=1\n
   action=SET_INTERVALSREC\n
   intervals=14-05-08 14:06:25 14-05-08 14:06:26;14-05-08 14:06:30 14-05-08 14:06:31;\n
   Msg=end\n
```



```
2. Msg=start\n
type=CAM\n
id=1\n
action=SET_INTERVALSREC\n
intervals=14:06:25 14:06:26;14:06:30 14:06:31;\n
date=14-05-08
Msg=end\n
```

В обоих случаях в сообщениях находится одна и та же информация, только по-разному отформатированная.

Информация о начале и конце записи разделены пробелом, информация о записях разделена ";"

день-месяц-год часы:минуты:секунды

14-05-08 14:06:25 14-05-08 14:06:26

Если значение параметра intervals пустое, значит за заданный день нет записей.

Имея в наличии список записей в архиве, file.name, file.id и набор команд ниже, можно организовать подобие плеера на клиенте (пауза, стоп, проигрывание вперёд-назад и т.п.).

Вход в архив - arc.enter

GET

```
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&command=arc.enter&intervals=true&date=15.05.10
09:51:07&login=XXX&password=YYY
```

command=arc.enter – команда входа в архив.

date – дата, архив за которую требуется получить.

intervals – присылать информацию о записях в архиве (начало и конец записи).

После входа в архив клиенту присылается сообщение с интервалами и 1 (один кадр).

Проигрывание одной записи архива - "arc.play"

GET

```
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&command=arc.play&file.name=C%3A%5CVIDEO%5C20-05-10%20
16%5C0._01&frame.id=0&login=XXX&password=YYY
```

file.name берётся из пришедшего видеофрайма, frame.id - фрейм, с которого требуется проигрывать архив.

Присылается последовательность фреймов из данного отрезка записи, и на этом проигрывание обрывается (отследить это можно по frame.id).

Непрерывное проигрывание записей архива - "arc.playnonstop"

GET

`http://example.com:[port][/somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&command=arc.playnonstop&file.name=C%3A%5CVIDEO%5C20-05-10%2016%5C0._01&frame.id=0&login=XXX&password=YYY`

Команда аналогична "arc.play", но проигрыватель не останавливается на одной записи, а продолжает проигрывать записи архива дальше. Не следует забывать обновлять `file.name` и интегралы, которые будут периодически меняться.

Проигрывание прекращается, когда все записи вплоть до конца архива проиграны.


Переход на один фрейм или на одну запись назад - "arc.prev"

GET

`http://example.com:[port][/somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&command=arc.prev&file.name=C%3A%5CVIDEO%5C20-05-10%2016%5C0._01&frame.id=0&login=XXX&password=YYY`

Если указывается параметр `frame.id`, то система ищет предыдущий фрейм в архиве и присылает его. Если этот параметр не указан, то присылается первый фрейм предыдущей записи.

В ответ присылается только один фрейм.

 **Примечание.**
Также могут приходить Msg.

Если достигнуто начало архива и больше нет записей, на которые можно было бы переходить, то присылается сообщение:

```
Msg=start\n
type=CAM\n
id=1\n
action=ARCH_ERROR\n
Msg=end\n
```


Переход на один фрейм или на одну запись вперёд - "arc.next"

GET

`http://example.com:[port][/somecontext]/secure/video/action.do?version=4.7.8.0&sessionid=29101F1&cam.id=5&command=arc.next&file.name=C%3A%5CVIDEO%5C20-05-10%2016%5C0._01&frame.id=0&login=XXX&password=YYY`

Если указывается параметр `frame.id`, то система ищет следующий фрейм в архиве и присылает его. Если этот параметр не указан, то присылается первый фрейм следующей записи.

В ответ присылается один фрейм.

 **Примечание.**
Также могут приходить Msg.

Если достигнут конец архива и нет больше записей, на которые можно было бы переходить, то присылается сообщение:

```
Msg=start\n
type=CAM\n
id=1\n
action=ARCH_ERROR\n
Msg=end\n
```

Остановить проигрывание - "arc.stop"

GET

[http://example.com:\[port\]/somecontext/secure/video/action.do?version=4.7.8.0&sessionId=29101F1&cam.id=5&command=arc.stop&login=XXX&password=YYY](http://example.com:[port]/somecontext/secure/video/action.do?version=4.7.8.0&sessionId=29101F1&cam.id=5&command=arc.stop&login=XXX&password=YYY)

Получение списка записей

GET

[http://example.com:\[port\]/somecontext/secure/archive/CAM:2/\[2011-12-30|2011-12\]\[?splitTreshold=50\]](http://example.com:[port]/somecontext/secure/archive/CAM:2/[2011-12-30|2011-12][?splitTreshold=50])

splitTreshold – если разница между окончанием предыдущей записи и началом следующей меньше этого числа (в миллисекундах), то записи объединяются в одну. Чтобы никакие записи не объединялись, необходимо указать splitTreshold=0. Значение данного параметра по умолчанию равно 50.

Всё время интерпретируется как локальное для сервера.

Пример. Записи за день (30 декабря 2011 года):

[http://example.com:\[port\]/somecontext/secure/archive/CAM:2/2011-12-30/](http://example.com:[port]/somecontext/secure/archive/CAM:2/2011-12-30/)

[http://example.com:\[port\]/somecontext/secure/archive/CAM:2/2011-12-01/](http://example.com:[port]/somecontext/secure/archive/CAM:2/2011-12-01/)

Пример ответа:

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<days>
  <day>
    <id>2011-09-01T00:00:00-05:00</id>
    <records>
      <from>2011-09-01T00:00:00-05:00</from>
      <to>2011-09-01T00:00:35-05:00</to>
    </records>
  </records>
</days>
```

```
        <from>2011-09-01T00:00:35-05:00</from>
        <to>2011-09-01T00:01:10-05:00</to>
    </records>
</day>
</days>
```

JSON:

```
[ {
  "id" : "2011-09-01T00:00:00-0500",
  "records" : [ {
    "from" : "2011-09-01T00:00:00-0500",
    "to" : "2011-09-01T00:00:35-0500"
  }, {
    "from" : "2011-09-01T00:00:35-0500",
    "to" : "2011-09-01T00:01:10-0500"
  }, {
    "from" : "2011-09-01T01:26:24-0500",
    "to" : "2011-09-01T01:26:59-0500"
  } ]
} ]
```

Пример. Записи за месяц (показывает, в какие дни сентября есть записи):

[http://example.com:\[port\]/\[somecontext\]/secure/archive/CAM:2/2011-12/](http://example.com:[port]/[somecontext]/secure/archive/CAM:2/2011-12/)

XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<days>
  <day>
    <id>2011-09-02T00:00:00-05:00</id>
  </day>
  <day>
    <id>2011-09-03T00:00:00-05:00</id>
```

```
</day>
<day>
  <id>2011-09-05T00:00:00-05:00</id>
</day>
</days>
```

JSON:

```
[ {
  "id" : "2011-09-01T00:00:00-0500",
  "records" : [ ]
}, {
  "id" : "2011-09-03T00:00:00-0500",
  "records" : [ ]
}, {
  "id" : "2011-09-01T00:00:00-0500",
  "records" : [ ]
} ]
```

Если записей нет, то присылается

XML:

```
<days/>
```

JSON:

```
[ ]
```

Нотификация

В разделе:

- [Подписка на сообщения](#)
- [Аннулирование подписки](#)
- [Формат сообщения APN](#)

Используются системы нотификации APNS(iOS), C2DN (Android) и т.д.
deviceid – device token (APNs), registration id (в случае C2DN) и т.д.;

username – логин пользователя. Может быть пустой.

Подписка на сообщения

Приложение при соединении с сервисом может осуществить подписку на сообщения APNS. В этом случае при выходе из программы на устройство будут приходить уведомления о тех или иных событиях.

POST

`http://example.com:[port]/[somecontext]/secure/subscription/`

Ответ с кодом "201 Created" означает, что подписка прошла успешно.

Код 400 означает, что параметры заданы не верно (deviceId не должно быть пустым, должно быть длиной от 5 до 150 символов и содержать только цифры и буквы английского алфавита).

Тело POST должно содержать информацию о создаваемой подписке. Принимается только формат JSON. Требуется корректно проставлять заголовок Content-Type.

Пример ответа:

JSON

Content-Type : application/json

```
{
  "username" : "johndoe",
  "deviceId" : "somedeviceid"
}
```

Аннулирование подписки

Аннулирование подписки происходит в следующих случаях:

- Пользователь подписался на события с другого устройства;
- Сменился device token или registration id;
- Другой пользователь подписался на события с данного устройства;
- Произошла ручная отписка от сообщений.

DELETE

`http://example.com:[port]/[somecontext]/secure/subscription/[deviceId]`

Ответ с кодом "204 No Content" означает, что подписка прошла успешно.

Формат сообщения APN

```
{
```

```
"aps" : {
    "alert" : "Motion Detected",
    "badge" : 2 // .
},
"e" : {
    "srv" : "XXX", //id . iOS
    "stt" : 88, //id (. )
    "obj" : "6", //id
    "ts" : "2010-08-02T23:30:00Z" //
}
}
```

Звук

Получение живого звука

GET

[http://example.com:\[port\]/somecontext/secure/video/action.do?version=4.9.0.0&sessionId=FC126734&command=audio.play&audio_in=MIC:5&format=L16&login=XXX&password=YYY](http://example.com:[port]/somecontext/secure/video/action.do?version=4.9.0.0&sessionId=FC126734&command=audio.play&audio_in=MIC:5&format=L16&login=XXX&password=YYY)

sessionId – идентификатор сессии (тут пока не используется).

audio_in – идентификатор аудиопотока.

format – формат аудиоданных (пока только L16).

В ответ будут получены аудиопакеты в следующем виде:

HTTP/1.0 200 OK

Connection: close

Server: ITV-Intellect-Webserver/4.9.0.0

Cache-Control: no-store,no-cache,must-revalidate,max-age=0

Pragma: no-cache

Date: Mon, 13 Jan 2013 10:44:27 GMT

Content-Type: multipart/mixed;boundary=audioframe

--audioframe

Content-Type: text/xml

Content-Length: 138

<audio_in>

<sessionid>FC126734</sessionid>

<audio_in>MIC:5</audio_in>

<newstate>started</newstate>

<errcode>100</errcode>

</audio_in>

--audioframe

Content-Type: audio/L16;rate=8000;channels=1

Content-Length: 1024

X-Time: 2013-03-22T13:16:31.371+04:00

<audio packet PCM16>

--audioframe

Content-Type: audio/L16;rate=8000;channels=1

Content-Length: 1278

X-Time: 2013-03-22T13:16:31.873+04:00

<audio packet PCM16>

Для остановки потока без разрыва соединения необходимо в этом же соединении отправить команду

GET
[http://example.com:\[port\]/\[somecontext\]/secure/video/action.do?version=4.9.0.0&sessionid=29101F1&command=audio.stop&audio_in=MIC:5&login=XXX&password=YYY](http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionid=29101F1&command=audio.stop&audio_in=MIC:5&login=XXX&password=YYY)

В этом случае в потоке придёт завершающий xml пакет:


```
--audioframe
Content-Type: text/xml
Content-Length: 106
<audio_in>
  <sessionid>FC126734</sessionid>
  <audio_in>MIC:5</audio_in>
  <newstate>closed</newstate>
  <errcode>100</errcode>
</audio_in>
```

Проигрывание звука из архива

GET

```
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionid=29101F1&command=arc.play&audio_in=MIC:5&format=L16&time_from=2013-03-22T13:16:31.873+04:00&time_to=2013-03-22T13:04:52.312+04:00&login=XXX&password=YYY
```

audio_in – идентификатор аудиопотока;

format – запрашиваемый формат (сейчас пока только L16);

time_from - время начала проигрывания архива;

time_to - время завершения проигрывания архива.

Поток приходит в том же виде, что и при живом звуке.

При завершении данных приходит завершающий xml пакет (как при получении живого звука – см. [Получение живого звука](#)).

Отправка живого звука

Отправка звука идёт последовательной передачей пакетов командами:

POST

```
http://example.com:[port]/[somecontext]/secure/video/action.do?version=4.9.0.0&sessionid=FC126734&command=audio.receive&audio_out=SPEAKER:3&login=XXX&password=YYY
```

```
Content-type: audio/L16;rate=8000;channels=1
Connection: keep-alive
```

Далее происходит передача аудиопакета.

Формат звука – только L16.

rate – любое разумное значение.

channels – от 1 до 6.