Подписка на метаданные аналитики по Websocket (описание JSON)

Процессинг слушает порт 9004 по протоколу websocket. Требует авторизации по http basic с жестко зашитым логином/паролем: ***VideoIntellect:Vi-events_wss_subscr_pass.***

Есть 2 URL для подписки:

- Только описание события в формате JSON:
  ws://хост_на_котором_запущен_процессинг:9004/
- Описание события в формате JSON + PNG кадр, на котором обнаружено событие (прямоугольником обведен объект с событием):
  ws://хост_на_котором_запущен_процессинг:9004/?with_frames=1

При появлении события видеоаналитики, каждому подключившемуся клиенту по Websocket рассылается сообщение.

По первому URL передаются сообщения в текстовом формате,
по второму URL - в бинарном TLV.

Формат бинарного TLV сообщения следующий, он содержит набор данных:

1. 2 байта – тип данных (в network byte order)
2. 4 байта – длина данных (в network byte order)
3. Данные длиной из пункта 2
4. 2 байта – тип данных (в network byte order)
5. 4 байта – длина данных (в network byte order)
6. Данные длиной из пункта 5

Типы:

0x1 – JSON

0x2 - PNG

В пакете всегда будет 1 JSON и 1 PNG. Также могут быть блоки других форматов, которые нужно пропускать.

Пока бинарный блок имеет формат: 0x1 [длина_json] [JSON] 0x2 [длина_png][PNG]

Формат JSON:

{

"detector": "abandoned", // уникальное имя детектора

"url": "rtsp://192.168.1.52/axis-media/media.amp", // url видеопотока, на котором детектор

обнаружил событие

"id": "{0000014c-0000-0000-5365-637572697800}", // id камеры из конфигурации

"no": "1", // номер потока

"timestamp": 1508979636011, // время события в милисекундах в UNIX timestamp по системному таймеру компютера, на котором крутится детектор

"rects": [// координаты прямоугольника с событием на кадре (нормированы на 1 по ширине и высоте кадра)

{

"x": 0.078125,

"y": 0.277778,

"w": 0.234375,

"h": 0.277778,

"g": 1, // не используется

"duration": 5000, // не используется

"uuid": "{6da5a2af-f8da-4e9d-84fa-7c0febeed1fe}" // уникальные id события, теоретически может прийти несколько событий с одинаковым uuid, что означает, что это одно и то же событие, но продолжительное по времени. Пока всегда приходят уникальные uuid

}

],

"detector_output": // необязательные дополнительные параметры

{ // формат внутри этих скобок зависит от детектора

Для “faulter”:

"type": "имя_типа_саботажа" // для детектора “faulter” определяет тип саботажа:

“dark” // затемнение изображения

“defocusing” // расфокусировка камеры

Для “service”:

“type”: “stream_connected” или

“type”: “stream_disconnected”

Для “face”:

Массив параметров для каждого найденного лица на кадре, массив поэлементно соответствует массиву rects (в котором координаты прямоугольников с найденными лицами)

[

{"person_id": <целое число id из БД>, "score": <дробное число [0..1] степень сходства лица с БД>}

…

]

Если лицо не распознано, то "person_id" и "score" отсутствуют в параметрах (то есть элемент пустой: “{}”)

Для “doorstate”:

“state”: “opened” или

“state”: “closed”

Для “zonecounter”:

“count”: <целое число - количество человек в зоне>

// для других детекторов могут быть другие данные, пока не определены

}

}

Список имен детекторов, которые могут приходить в JSON:

1. "dummy" "Тестовый детектор с периодической генерацией событий"

2. "highload" "Тестовый детектор с ручной генерацией и нагрузкой на процессор"

3. "abandoned" "Детектор оставленных предметов"

4. "crowd" "Детектор образования толпы"

5. "direction" "Детектор движения в заданном направлении"

6. "forbidden" "Детектор движения в запрещенной зоне"

7. "violence" "Детектор агрессивного поведения"

8. "servtime" "Детектор времени обслуживания клиента"

9. "facecount" "Детектор подсчёта посетителей по лицам"

10. "shopper" "Детектор заинтересованности покупателя к товару"

11. "crossing" "Детектор пересечения линии в запрещенном направлении"

12. "activity" "Детектор активности на рабочем месте"

13. "faulter" "Детектор саботажа камеры"

14. “service” “Сервисный детектор для событий, происходящих внутри службы (например, разрыв соединения с камерой)”

15. “face” “Детектор распознавания лиц”
16. “banner” “детектор баннеров, плакатов”
17. “quorum” “Детектор скопления людей”
18. “iparking” “Парковка в неположенном месте”
19. “zonecounter” “Детектор подсчета количества людей в зоне”
20. “sheet” “Дектор обнаружения листа бумаги”
21. “banknotes” “Детектор обнаружения денежных купюр”
22. “bankcassette” “Детектор открытия банковской кассеты для купюр”
23. “codesheet” “Детектор обнаружения листа бумаги с напечатанным текстом”

Есть поддержка получения событий через onvif RealtimePullPoint интерфейс.