Milesight-Troubleshooting

Integration between LPR Camera and NVR(VMS)

Version	V2.0	Update	2023.9.21

Integration between LPR Camera and NVR(VMS)

Description

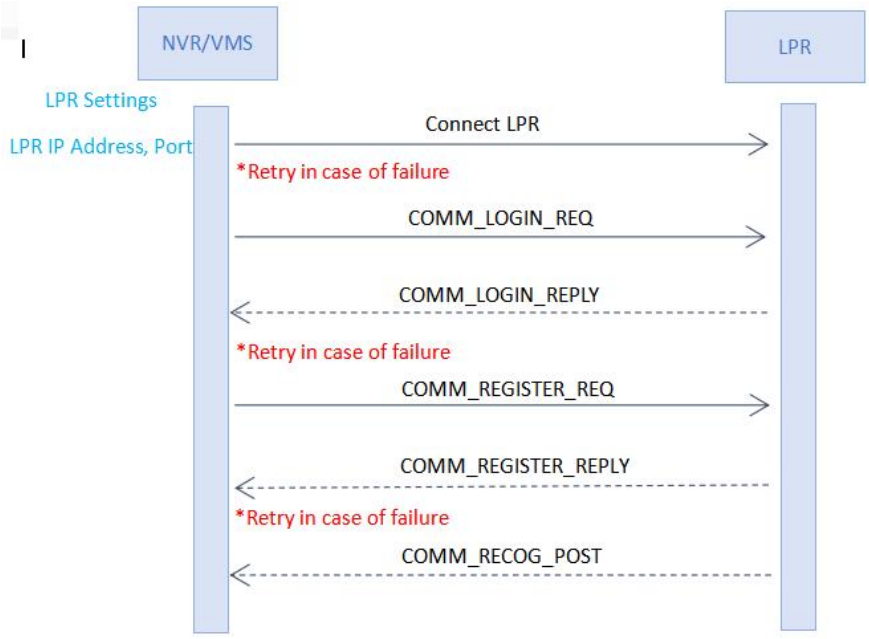
This document will introduce how to integrate between Milesight LPR camera and NVR or VMS (Including Milesight brand and the third parties brand). There are three post types: **TCP, HTTP and RTSP**.

Note:

1. For the **Evidence function,** please make sure your camera's model is TSxxxx-xxC or MS-Cxxxx-xLC and your camera's version is 45.8.0.3-LPR*-r1 or above.
2. For the **Parking Detection,** please make sure your camera's model is TSxxxx-FPC/P and your camera's version is 45.8.0.3-LPR*-r3 or above.
3. For the **Vehicle Counting,** please make sure your camera's model is TSxxxx-xxC (Except for TSxxxx-FPC/P).

1. TCP Type

- **System Structure**



Enter the LPR IP address and LPR port on the NVR/VMS to register the LPR camera. Then LPR

Camera will transmit the data of recognition result to the NVR/VMS when the license plate is recognized. The data includes the time that was recognized, the license plate, the license plate snapshot, and the full-screen snapshot.

- **List of messages**

	Command Name	Explanation
0x0001	COMM_LOGIN_REQ	Request login to LPR camera.
0x8001	COMM_LOGIN_REPLY	Response to COMM_LOGIN_REQ.
0x0002	COMM_REGISTE_REQ	Request to register features that needs to be pushed.
0x8002	COMM_REGISTE_REPLY	Response to registration status.
0x8801	COMM_RECOG_POST	The metadata and image of recognition result.
0x8802	COMM_PARKING_POST	Parking Detection data (registration required).
0x8803	COMM_COUNTING_POST	Vehicle Counting data(registration required).

- **Basic Packet Composition**

1. SIG — CC FF : Packet Start
2. SIG END — FF DD : Packet End
3. COMMAND
4. DATA_SIZE
5. DATA

Basic Packet Composition

SIG	COMMAND	DATA_SIZE	DATA	SIG END
2 byte	2 byte	4 byte	Variable	2 byte

- **COMM_LOGIN_REQ**

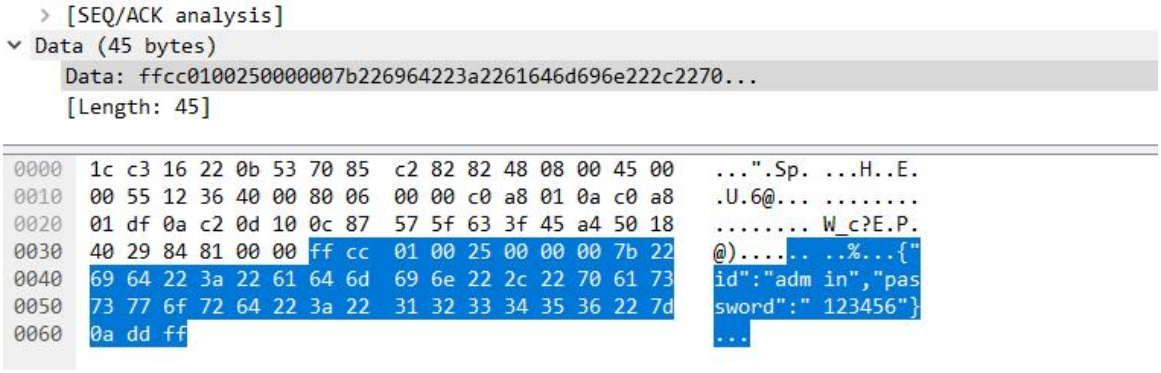
1. Data Type: JSON-charset=utf-8
2. Data Content: ID and Password
3. Example

```
{
    “id”: “admin”,
    “password”: “1234”
}
```

SIG	COMMAND	DATA_SIZE	DATA	SIG END
2 byte	2 byte	4 byte	Variable	2 byte
CC FF	0x0001	Variable	JSON DATA	FF DD

※ DATA_SIZE = DATA length + 2 bytes(SIG END)

※ Use Little-Endian



- **COMM_LOGIN_REPLY**

1. Data Type : JSON-charset=utf-8
2. Data Content : result - Required Items

HTTP Status code

a. 200 : OK

b. 401 : No Privileges

c. Etc...

3. Example

{

"result": "200"

}

SIG	COMMAND	DATA_SIZE	DATA	SIG END
2 byte	2 byte	4 byte	Variable	2 byte
CC FF	0x8001	Variable	JSON DATA	FF DD

- **COMM_REGISTER_REQ**

1. Data Type : JSON-charset=utf-8
2. Data Content: register-Required Items

a. registerAll

b. Lprparking

c. Vehiclecount

3. Example

①When you want to register all features (It currently includes Parking Detection and Vehicle Counting), you can set "registerAll" to 1. (Please note that if your camera's model or firmware version does not support this feature, even if set the "registerAll" to 1, it will not work.) Refer to the following case:

{

“registerAll”: 1,

}

②When you want to register only one feature, you can set that feature to 1 and set the “registerAll” to 0, refer to the following case:

{

“registerAll”: 0,

“vehiclecount”: 1

}

Or

{

“registerAll”: 0,

“lprparking”: 1

}

SIG	COMMAND	DATA_SIZE	DATA	SIG END
2 byte	2 byte	4 byte	Variable	2 byte
CC FF	0x0002	Variable	JSON DATA	FF DD

- **COMM_REGISTER_REPLY**

1. Data Type : JSON-charset=utf-8
2. Data Content : result - Required Items

HTTP Status code

d. 200 : OK

e. 401 : No Privileges

f. Etc...

3. Example

{

“result”: “200”

“registerAll”: 1,

}

SIG	COMMAND	DATA_SIZE	DATA	SIG END
2 byte	2 byte	4 byte	Variable	2 byte

CC FF	0x8002	Variable	JSON DATA	FF DD

- **COMM_RECOG_POST**

1. Recognition Result Message
   - ❖ The LPR camera sends the recognition results on its own initiative without requiring a request from the NVR/VMS.
   - ❖ Data Type : Binary
   - ❖ Data Content

a. **metadata**

① Device ID : 16 byte – GUID byte array : 04 f9 12 bb ce 94 65 40 89 af e8 3c d8 8f 70 be

② recognition time : 8 byte – Posix Time : 1525867890000

③ ~~Car Number : 16 byte – utf-8 string : "부산 02 가 1234"<< NULL Exclude Fixed Size~~

④ Color of the Car : 1 byte – refer to the color table（stand by）

⑤ Color of the licence plate : 1 byte – refer to the color table（stand by）

⑥ Speed : 2 byte – unsigned short integer, Km/h Unit (mph unit need to convert yourselves)

⑦ Number of resulting images : 1 byte

⑧ Direction : 1 byte – 0: Unknown 1: In 2: Out

⑨ Region: 32 byte

⑩ ROI ID : 1 byte 1~4 0:unknown

⑪ plate's length

⑫ license plate

⑬ Vehicle Type: 1 byte – refer to the vehicle type table（stand by）

⑭ Confidence: 4 byte(float)

⑮ Plate Type: 1: black 2: white 3: visitor

⑯ Distance: (int)need to enable radar

⑰ Azimuth: (float) need to enable radar

⑱ Vehicle Count: need to enable radar

⑲ Width: resolution width

⑳ Height: resolution height

㉑ coordinate_x1: The left coordinates of license plate.

㉒ coordinate_y1: The top coordinates of license plate.

㉓ coordinate_x2: The right coordinates of license plate.

㉔ coordinate_y2: The bottom coordinates of license plate.

㉕ Vehicle Brand: 2 byte – refer to the brand table（stand by）

b. **Image data** : variable size

c. Data Chunk

① Chunk ID : 4 byte

Meta : 11 ff 00 00

Image : 22 ff 00

② Chunk Size : 4 byte

Data size excluding Chunk Header 8 byte.

2. Packet Example

SIG	COMMAND	DATA_SIZE	DATA	SIG END
2 byte	2 byte	4 byte	Variable	2 byte
CC FF	0x8801	Variable	...	FF DD

Metadata Chunk	Image Chunk	Image Chunk

Chunk Header		Metadata Chunk							
Chunk ID	Chunk Size 4 byte	GUID 16 byte	Time 8 byte	~~Number~~(Obsolescent) 16 byte	V-color 1 byte	P-color 1 byte	Speed 2 byte	I-count 1 byte	Direction 1byte
11 FF 00 00	110+Number length（GUID+...+Number）	04 f9 12 ...	0x16344 D04550	~~"부산 01 가 1234"~~	0x01	0x01	100	2	0
		Region 32byte	ROI ID 1byte	Plate Len 1byte	Number Variable	Vehicle Type 1 byte	Confidence 4 byte	Plate type 1 byte	Distance 4 byte
		WOB/ZK	1	6	"AB1234"	1		3	30
		Azimuth 4 byte	Vehicle Count 4 byte	Width 2 byte	Height 2 byte	coordinate_x1 2 byte	coordinate_y1 2 byte	coordinate_x2 2 byte	coordinate_y2 2 byte
		3.5	50	1280	720				
		Brand 2 byte							

		BMW							

Plate_image:

Chunk Header		Image Chunk
Chunk ID 4 byte	Chunk Size 4 byte	JPEG image data
22 FF 00 00	Variable	FF D8 FF E0 ...

Full_image:

Chunk Header		Image Chunk
Chunk ID 4 byte	Chunk Size 4 byte	JPEG image data
22 FF 00 00	Variable	FF D8 FF E0 ...

Note: If the Evidence function is set, the following is the case with LPR images:

①No Plates: Full Image + Evidence Image(If you have it.)

②With Plates: Plate image + Full Image + Evidence Image(If you have it.)

Evidence_image0:

Chunk Header		Image Chunk
Chunk ID 4 byte	Chunk Size 4 byte	JPEG image data
22 FF 00 00	Variable	FF D8 FF E0 ...

Evidence_image1:

Chunk Header		Image Chunk
Chunk ID 4 byte	Chunk Size 4 byte	JPEG image data
22 FF 00 00	Variable	FF D8 FF E0 ...

- **Color Table**

Parameter	Note
enum LprColor{	
LPR_COLOR_UNKNOWN = 0,	
LPR_COLOR_BLACK,	"Black"
LPR_COLOR_BLUE,	"Blue"
LPR_COLOR_CYAN,	"Cyan"
LPR_COLOR_GRAY,	"Gray"
LPR_COLOR_GREEN,	"Green"
LPR_COLOR_RED,	"Red"
LPR_COLOR_WHITE,	"White"
LPR_COLOR_YELLOW,	"Yellow"
LPR_COLOR_VIOLET,	"Violet"
LPR_COLOR_ORANGE	"Orange"
};	

- **Brand Table**

Parameter	Note
typedef enum traBrandType{	
TRA_BRAND_UNKNOW = 0,	
TRA_BRAND_AUDI,	"Audi"
TRA_BRAND_ASTONMARTIN,	"Aston Martin"
TRA_BRAND_ALFAROMEO,	"Alfa Romeo"
TRA_BRAND_BUICK,	"Buick"
TRA_BRAND_MERCEDESBENZ,	"Mercedes—Benz"
TRA_BRAND_BMW,	"BMW"
TRA_BRAND_HONDA,	"Honda"
TRA_BRAND_PEUGEOT,	"Peugeot"
TRA_BRAND_PORSCHE,	"Porsche"
TRA_BRAND_BENTLEY,	"Bentley"
TRA_BRAND_BUGATTI,	"Bugatti"
TRA_BRAND_VOLKSWAGEN,	"Volkswagen"
TRA_BRAND_DODGE,	"Dodge"
TRA_BRAND_DAEWOO,	"Daewoo"
TRA_BRAND_DAIHATSU,	"Daihatsu"
TRA_BRAND_TOYOTA,	"Toyota"
TRA_BRAND_FORD,	"Ford"
TRA_BRAND_FERRARI,	"Ferrari"
TRA_BRAND_FIAT,	"Fiat"
TRA_BRAND_GMC,	"GMC"
TRA_BRAND_MITSUOKA,	"MITSUOKA"
TRA_BRAND_HAVAL,	"Haval"
TRA_BRAND_GEELY,	"Geely"
TRA_BRAND_JEEP,	"Jeep"
TRA_BRAND_JAGUAR,	"Jaguar"
TRA_BRAND_CADILLAC,	"Cadillac"
TRA_BRAND_CHRYSLER,	"Chrysler"
TRA_BRAND_LEXUS,	"Lexus"
TRA_BRAND_LANDROVER,	"Land Rover"
TRA_BRAND_LINCOLN,	"Lincoln"
TRA_BRAND_SUZUKI,	"Suzuki"
TRA_BRAND_ROLLSROYCE,	"Rolls-royce"
TRA_BRAND_LAMBORGHINI,	"Lamborghini"
TRA_BRAND_RENAULT,	"Renault"
TRA_BRAND_MAZDA,	"Mazda"
TRA_BRAND_MINI,	"MINI"
TRA_BRAND_MASERATI,	"Maserati"
TRA_BRAND_MAYBACH,	"Maybach"
TRA_BRAND_ACURA,	"Acura"

TRA_BRAND_OPEL,	"Opel"
TRA_BRAND_CHERY,	"Chery"
TRA_BRAND_KIA,	"Kia"
TRA_BRAND_NISSAN,	"Nissan"
TRA_BRAND_SKODA,	"Skoda"
TRA_BRAND_MITSUBISHI,	"Mitsubishi"
TRA_BRAND_SUBARU,	"Subaru"
TRA_BRAND_SMART,	"Smart"
TRA_BRAND_SSANGYONG,	"Ssangyong"
TRA_BRAND_TESLA,	"Tesla"
TRA_BRAND_ISUZU,	"Isuzu"
TRA_BRAND_CHEVROLET,	"Chevrolet"
TRA_BRAND_CITROEN,	"Citroen"
TRA_BRAND_HYUNDAI,	"Hyundai"
TRA_BRAND_INFINITY,	"Infinity"
TRA_BRAND_MERCURY,	"Mercury"
TRA_BRAND_SATURN,	"Saturn"
TRA_BRAND_SAAB,	"SAAB"
TRA_BRAND_LYNKCO,	"LYNK&CO"
TRA_BRAND_MORRISGARAGES,	"MorrisGarages"
TRA_BRAND_PAGANI,	"pagani"
TRA_BRAND_SPYKER,	"Spyker"
TRA_BRAND_BYD,	"BYD"
TRA_BRAND_MCLAREN,	"McLaren"
TRA_BRAND_KOENIGSEGG,	"Koenigsegg"
TRA_BRAND_VOLVO,	"Volvo"
TRA_BRAND_LANCIA,	"Lancia"
TRA_BRAND_SHELBY,	"Shelby"
TRA_BRAND_SEAT,	"Seat"
TRA_BRAND_CUPRA,	"CUPRA"
TRA_BRAND_DACIA,	"Dacia"
TRA_BRAND_DS,	"DS"
TRA_BRAND_MAX	
} TRA_BRAND_ENUM;	

- **Vehicle Type Table**

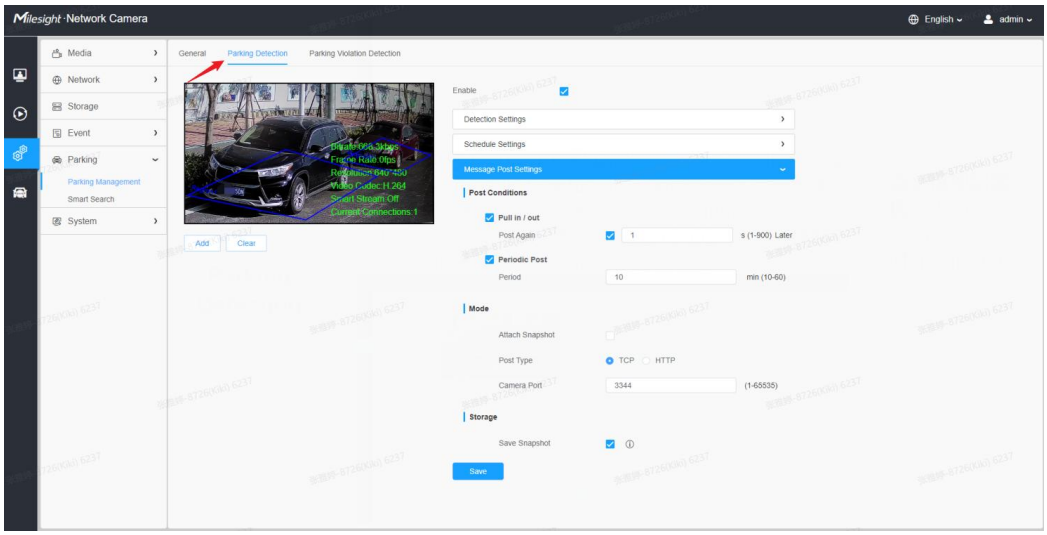
parameter	value
none	0
car	1
motor	2
bus	3
truck	4
van	5
suv	6

forklift	7
excavator	8
towtruck	9
Police-car	10
fireengine	11
ambulance	12
bicycle	13
E-bike	14
other	15

Note: All new features starting with version XX.8.0.3-LPR*-r1 push the LPR information data in json format to the VMS or NVR in real time when it is recognized. Such as Parking Detection and Vehicle Counting.

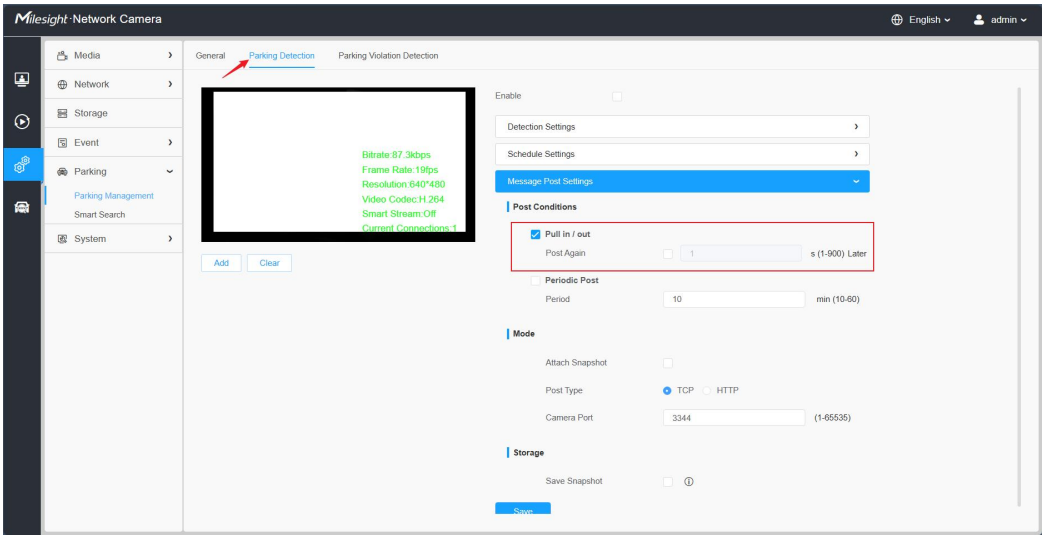
- **COMM_PARKING_POST**

This is specifically for Parking Detection, make sure your camera's model and firmware version support this function, as shown below:



There are two post modes for the Parking Detection: **Trigger mode** and **Period mode**, please refer to the following cases:

①Trigger Mode



{

"event": "Parking Detection",

"device": "Network Camera",

"time": "2023-01-02 23:46:24",

"report_type": "trigger",

"region": 1,

"region name": "ROI1",

"state": 0,

"Length of Parking": "00:00:31",

"License Plate": "BD925RA",

"Plate Color": "Blue",

"Vehicle Type": "Car",

"Vehicle Color": "Blue",

"Vehicle Brand": "Buick",

"Object tracking box_x1": 732,

"Object tracking box_y1": 885,

"Object tracking box_x2": 1193,

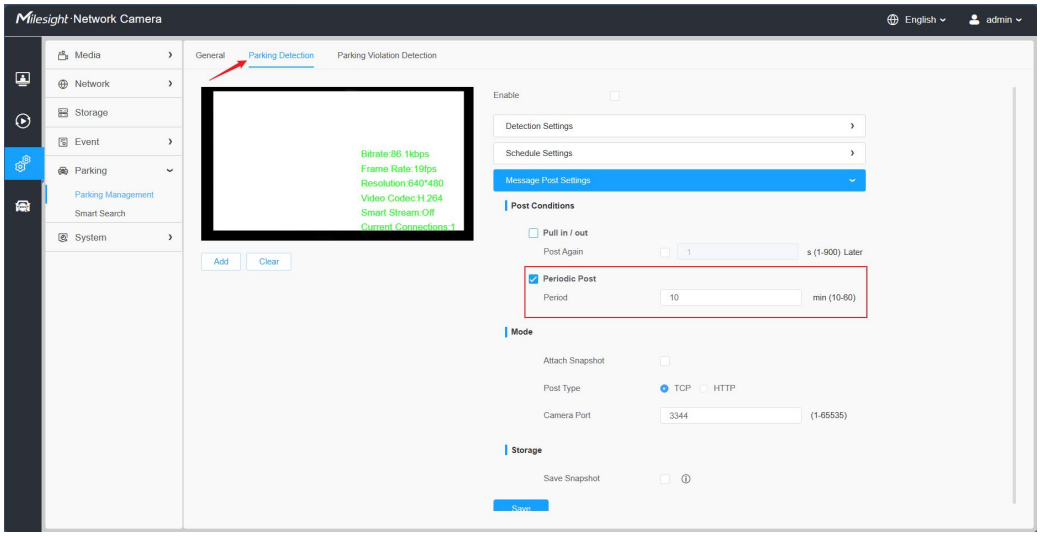
"Object tracking box_y2": 1558,

"plateSnap": "....(BASE64 code)",

"bkgSnap": "....(BASE64 code)"

}

②Period Mode

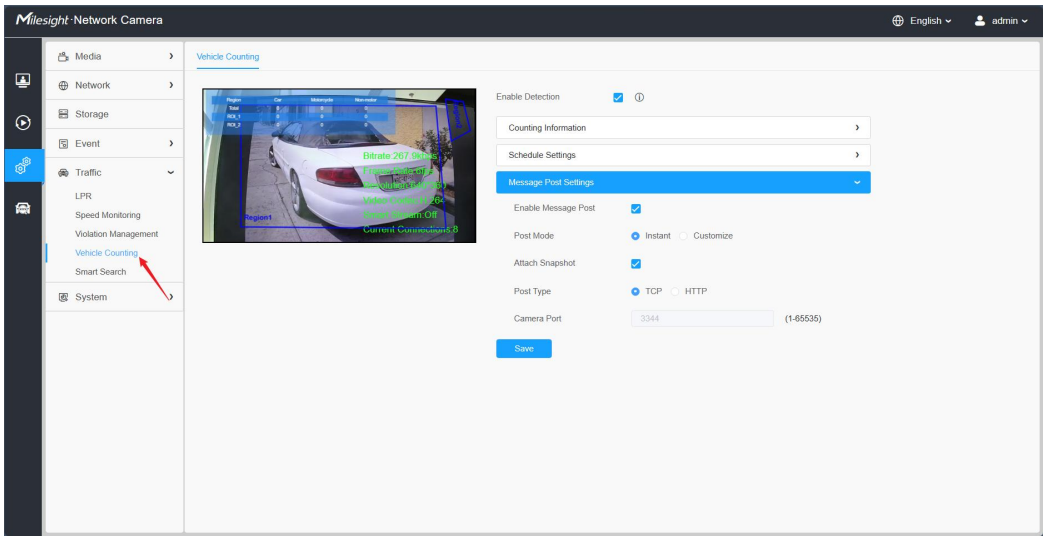



```
{
    "event":    "Parking Detection",
    "device":   "Network Camera",
    "time":     "2023-01-02 23:54:54",
    "report_type":   "interval",
    "total_occupied":       1,
    "total_available":      3,
    "region":   [1, 2, 3, 4],
    "region name":   ["ROI_1", "ROI_2", "ROI_3", "ROI_4"],
    "state":    [1, 0, 0, 0],
    "data list": [{
                "region":   1,
                "region name":   "ROI_1",
                "state":    1,
                "Length of Parking":    "00:00:17",
                "License Plate":    "DD925RA",
                "Plate Color":      "Blue",
                "Vehicle Type":   "SUV",
                "Vehicle Color":   "Blue",
                "Vehicle Brand":  "Buick",
                "Object tracking box_x1":    1092,
                "Object tracking box_y1":    870,
                "Object tracking box_x2":    1509,
                "Object tracking box_y2":    1541,
            "plateSnap":   "....(BASE64 code)"
        },
        {
                "region":   2,
                "region name":   "ROI_2",
                "state":    1,
                "Length of Parking":    "00:01:17",
```

```
                "License Plate":  "DD92AAA",
                "Plate Color":    "Blue",
                "Vehicle Type":   "SUV",
                "Vehicle Color":  "Blue",
                "Vehicle Brand":  "Buick",
                "Object tracking box_x1":    1092,
                "Object tracking box_y1":    870,
                "Object tracking box_x2":    1509,
                "Object tracking box_y2":    1541,
            "plateSnap":  "....(BASE64 code)"
        }
    ],
     "bkgSnap": "....(BASE64 code)"
}
```

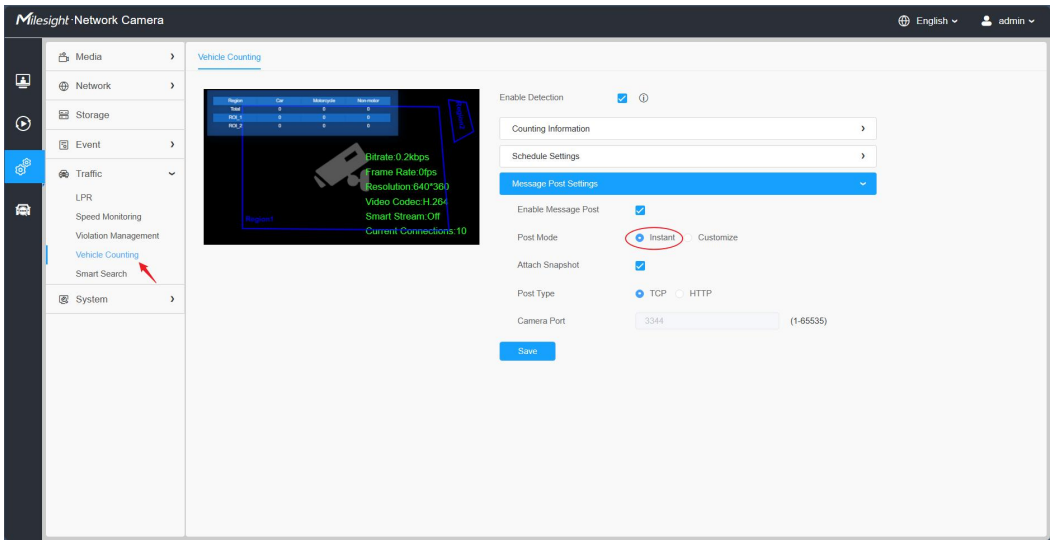

- **COMM_COUNTING_POST**

This is specifically for the Vehicle Counting, make sure your camera's model and firmware version support this function, as shown below:



There are two post modes for the Vehicle Counting: **Instant mode** and **Customize mode**, please refer to the following cases:

①Instant Mode



{

"event": "Vehicle Counting",

"device": "Network Camera",

"time": "2023-01-03 00:28:07",

"region": 1,

"region name": "ROI_1",

"All-Car": 1,

"All-Motorcycle":0,

"All-Non-motor":0,

"Car": 1,

"Motorbike": 0,

"Bus": 0,

"Truck": 0,

"Van": 0,

"SUV": 0,

"Fire engine": 0,

"Ambulance": 0,

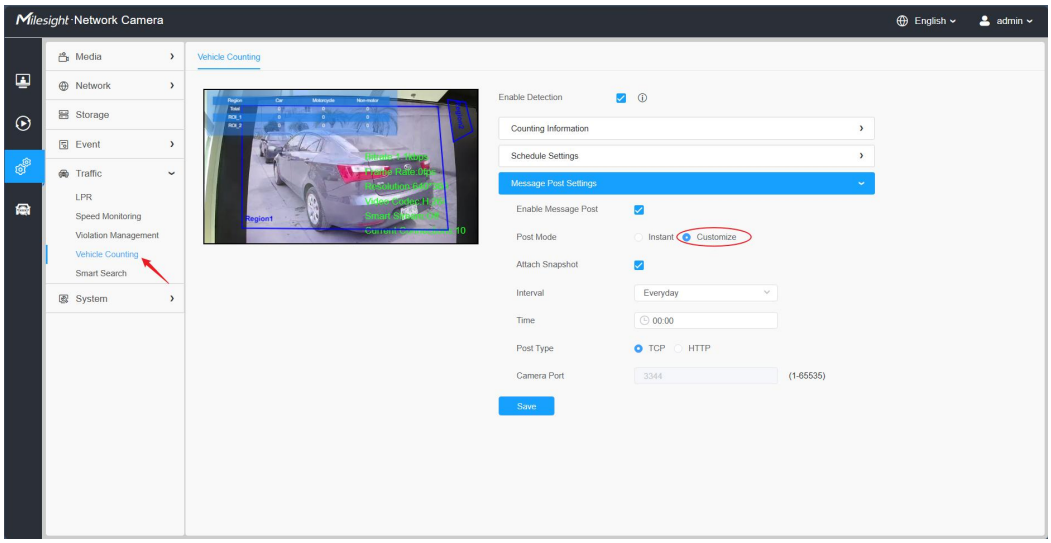
"Bicycle": 0,

"Other": 0,

"snapshot": "....(BASE64 code)"

}

②Customize Mode



{

"event": "Vehicle Counting",

"device": "Network Camera",

"time": "2023-01-03 01:00:00",

"regionList": [{

"region": 1,

"region name": "ROI_1",

"All-Car": 23,

"All-Motorcycle":0,

"All-Non-motor":0,

"Car": 19,

"Motorbike": 0,

"Bus": 0,

"Truck": 1,

"Van": 0,

"SUV": 3,

"Fire engine": 0,

"Ambulance": 0,

"Bicycle": 0,

"Other": 0

}, {

"region": 2,

"region name": "ROI_2",

"All-Car": 40,

"All-Motorcycle":1,

"All-Non-motor":0,

"Car": 35,

"Motorbike": 1,

"Bus": 2,

"Truck": 2,

```
                "Van":      0,
                "SUV":      1,
                "Fire engine":      0,
                "Ambulance":      0,
                "Bicycle":   0,
                "Other":    0
            }, {
                "region name":   "Total",
                "All-Car":   63,
                "All-Motorcycle":1,
                "All-Non-motor":0,
                "Car":      54,
                "Motorbike":      1,
                "Bus":      2,
                "Truck":    3,
                "Van":      0,
                "SUV":      4,
                "Fire engine":      0,
                "Ambulance":      0,
                "Bicycle":   0,
                "Other":    0
            }],
        "snapshot":       "....(BASE64 code)"
}
```


2. HTTP Type

● Integrate Method

For the HTTP Type, currently our LPR camera supports HTTP Post and Get request method.

VMS or NVR needs to develop matched API to receive the LPR information from the camera. The matched API URL may be like below:

URL of Post Method: http://IP:Port/xxxx

URL of Get Method: http://IP:Port/xxxx?

After VMS or NVR has completed the API, our LPR camera could use the API URL to send LPR information to the VMS or NVR when the license plate is recognized.

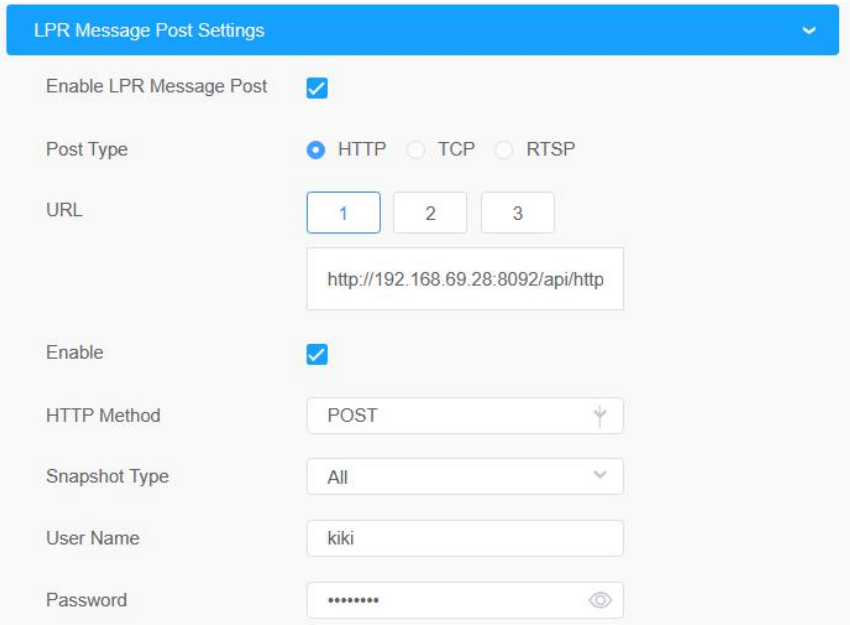
● LPR Information transfer

✓ Post Method

Take an example, the API URL from a VMS is like：

“http://192.168.69.28:8092/api/httpEvent”

Fill in the specified URL in camera’s web UI (if the VMS requires the authentication, please also fill in) :



Camera will post the LPR information data in json format to the VMS or NVR in real time when it is recognized.

The content that will be sent is as follows:

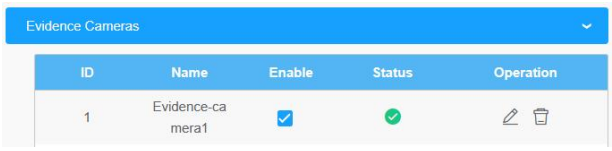
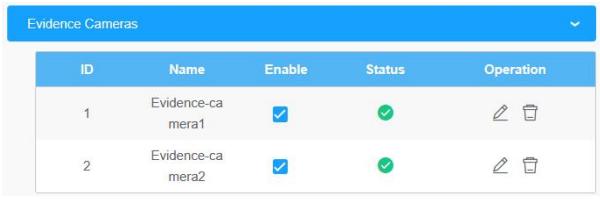
```
POST /api/httpEvent HTTP/1.1
Host: 192.168.69.28:8092
Authorization: Digest username="kiki", realm="Milesight Public RTSP", nonce="736aldef736aldef736aldef736aldef", uri="/api/httpEvent",
response="15e8072001620d7cac86b03e8b6af976"
Accept: */*
Content-Type: application/json
Content-Length: 320166

{
      "device":     "......demo.............",
      "time":       "2023-09-15 06:19:57.267",
      "time_msec":  "2023-09-15 06:19:57.267",
      "plate":      "NSC5870",
      "type":       "Visitor",
      "speed":      "-",
      "direction":  "-",
      "detection_region": "2",
      "region":     "NLD",
      "resolution_width": "1920",
      "resolution_height":      "1080",
      "coordinate_x1":     "539",
      "coordinate_y1":     "736",
      "coordinate_x2":     "699",
      "coordinate_y2":     "874",
      "confidence": "-",
      "plate_color":       "White",
      "vehicle_type":      "Car",
      "vehicle_color":     "Red",
      "Vehicle Brand":     "Volkswagen",
      "plate_image":       "/9j/4AAQSkZJRgABAQAAAQABAAD/
2wBDAAUDBAQEAwUEBAQFBQUGBwwIBwcHBw8LCwkMEQ8SEhEPERETFhwXExQaFRERGCEYGh0dHx8fExciJCIeJBweHx7/2wBDAQUFBQcGBw4ICA4eFBEUHh4eHh4eHh4eHh4eHh4eHh4eHh4eHh4
eHh4eHh4eHh4eHh4eHh4eHh7/wAARCACKAKADASIAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAECAwQFBgcICQoL/
8QAtRAAAgEDAwIEAwUFBAQAAAF9AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZnaGlqc3R1dnd4eXqDhIWGh4iJ
ipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8vP09fb3+Pn6/8QAHwEAAwEBAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL/
```

"full_image": "/9j/4AAQSkZJRgABAQAAAQABAAD/

"evidence_image0": "/9j/4AAQSkZJRgABAQAAAQABAAD/

"evidence_image1": "/9j/4AAQSkZJRgABAQAAAQABAAD/

Key	Sample of Value	Description
device	.....demo.....	The Device Name which can be configured on the System Info of camera. The default is Network Camera.
time	2023-09-15 06:19:57.267	The time when license plate is recognized.
time_msec	2023-09-15 06:19:57.267	The time when license plate is recognized.(Accurate to the millisecond level.)
plate	NSC5870	The recognized license plate number.
type	Visitor	The plate list type of recognized license plate, Black or White or Visitor.
speed	-	The running speed of detected vehicle.
direction	-	The driving direction of detected vehicle, Approach or Away.
detection_region	2	The ID of detection region where the vehicle is being tested, 1 or 2 or 3 or 4.
region	NLD	The registration country/region of the recognized license plate.
resolution_width	1920	The width of LPR processing resolution.
resolution_height	1080	The height of LPR processing resolution.
coordinate_x1, coordinate_y1	539, 736	The top left coordinates of license plate.
coordinate_x2, coordinate_y2	699, 874	The bottom right coordinates of license plate.
confidence	-	The confidence value of recognized license plate.
plate_color	White	The color of recognized license plate.
vehicle_type	Car	The type of recognized vehicle.
vehicle_color	Red	The color of recognized vehicle.
Vehicle Brand	Volkswagen	The brand of recognized vehicle.
plate_image		The snapshot of license Plate, depends on whether it is configured to send together. As shown below, it will be sent together if select License Plate or All. 
full_image		The full snapshot, depends on whether it is configured to send together. As shown below, it will be sent together if select Full Snapshot or All. 
evidence_image0		The snapshot of license Plate from evidence camera 1, depends on whether a linkage evidence camera is added to your LPR camera.

		
evidence_image1		The snapshot of license Plate from evidence camera 2, depends on whether two linkage evidence cameras are added to your LPR camera. 

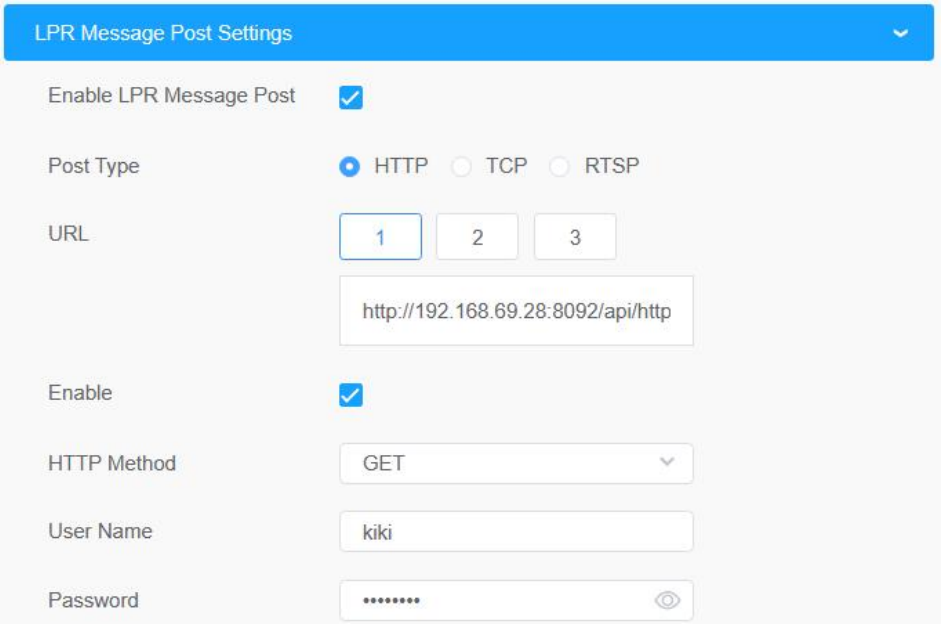
Note:

1. The common format for field names for functions prior to version 45.8.0.3-LPR*-r1 is: abc_def. Such as: plate_color and vehicle_type.
2. The common format for field names for functions in 45.8.0.3-LPR*-r1 and above version is: Abc Def. Such as: Vehicle Brand and Vehicle Type.
3. It is recommended to capture the packet to confirm the field names of the current camera version and functions first before compatibility.

✓ Get Method

Take an example, the API URL from a VMS is like "http://192.168.69.28:8092/api/http"

Fill in the specified URL in camera's web UI (if the VMS requires the authentication, please also fill in) :



For sending the license plate information, the LPR camera will automatically add the license

plate parameters to the URL.

For example, the license plate is “MS12345”. Once it’s detected, the LPR camera will send below URL to VMS:

http://192.168.69.28:8092/api/httpEventsource=LPR&description=MS12345

If the license plate information is to be displayed in VMS, the VMS side needs to extract it from the URL.

3. RTSP Type

● Prerequisites

This part is implemented in onvif metadata. There are three streams in rtsp: video stream, audio stream, and alarm stream. Metadata alarm is performed through the onvif alarm stream in the rtsp. So if the VMS or NVR supports and can receive the onvif alarm stream in the rtsp, it can work with rtsp.

● Integrate Method

We have defined the format of the XML. Knowing the XML format, VMS or NVR can be developed to be integrated, and LPR information can be displayed in VMS or NVR.

ex.) The contents of the xml include the date, time, license plate, and license plate snapshot paths as shown below.

```
<tt:MetaDataStream>
<tt:Event>
<wsnt:NotificationMessage>
<wsnt:Topic
Dialect="http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet">tns1:RuleEngine/LicensePlateDetector/LicensePlate</wsnt:Topic>
<wsnt:Message>
<tt:Message UtcTime="2018-05-15T06:19:34Z" PropertyOperation="Changed">
<tt:Source>
<tt:SimpleItem Name="VideoSourceConfigurationToken" Value="VideoSourceToken"/>
<tt:SimpleItem Name="VideoAnalyticsConfigurationToken" Value="VideoAnalyticsToken"/>
<tt:SimpleItem Name="Rule" Value="MyLicensePlateDetectorRule"/>
```

```
</tt:Source>
<tt:Key>
<tt:SimpleItem Name="LicensePlateResult" Value="43 거 6510"/>
</tt:Key>
<tt:Data>
<tt:SimpleItem Name="LicensePlatePicturePath" Value="/LPR/2018051506193401.jpg"/>
<tt:SimpleItem Name="LicenseCarSpeed" Value="25km/h"/>
<tt:SimpleItem Name="LicenseCarDirection" Value="1"/>
</tt:Data>
</tt:Message>
</wsnt:Message>
</wsnt:NotificationMessage>
</tt:Event>
</tt:MetaDataStream>
```

Description: You can receive the recognized license plate image through the snapshot path and the command named get (ex. http://IP:PORT/LPR/2018051506193401.jpg). Only 10 latest images are available for download. (If you can't see image in the NVR or VMS, type url (ex.http://IP:PORT/LPR/2018051506193401.jpg) in the web browser address box to see if the image is visible.)

- **LPR information transfer**

When the integration is complete, the LPR camera sends an xml containing LPR information to the VMS or NVR in real time when it is recognized.

—END—