Metrici LPR
~ Third party integration ~

Metrici LPR consists of one or several applications for detecting and recognising license plates (LPR engines) and a data display interface. LPR engines are independent applications that analyse real time video streams from the IP cameras and send, by HTTP POSTS, the information regarding the detected and recognised plates to the data display system. This system is formed of a MySQL relational data base and a collection of php scripts.

Integration with these systems can be made by one of the following three methods:

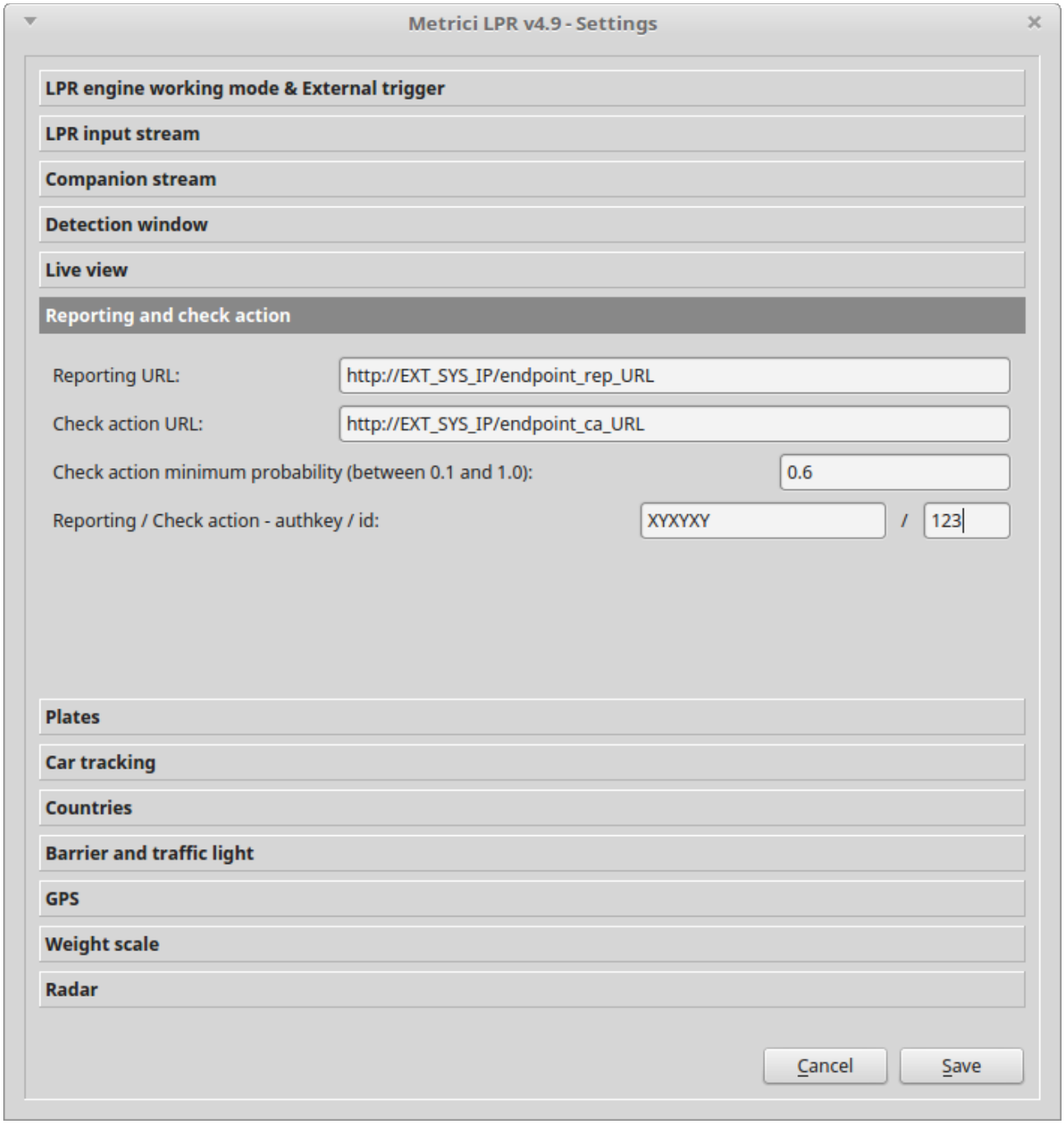
1. By PUSH directly from the LPR engines to the external system;
2. By PUSH from the reporting interface to the external system;
3. By GET from the reporting interface, using the methods from API.

1. PUSH Connection from LPR engines to the external system.

This working mode is useful when one wishes to receive events in real time and also to answer with an action (for ex: barrier opening, alert sending, etc.), following the decision taken in the external system. The drawback is that the data do not reach the web interface directly, being left to the programmer to create a retransmission mechanism for it, if necessary.

For each vehicle / license plate, the LPR engine generates two events: action checking, and reporting. The first event occurs when the license plate enters in the camera view, and the second when the vehicle leaves and is no longer visible in the image.

For each of these two events, the LPR engine sends the data by PUSH over HTTP, in a multipart/form-data format defined in RFC2388 (http://tools.ietf.org/html/rfc2388), to the two distinct URLs, which can be defined in the application interface.



The data sent are the following:

id (int, camera id, it can be from your data base, ex: 1 = ENTRANCE, 2 = EXIT)

number (string, recognised licence plate, no spaces)

country_code (string, RO, BG, I, F, D, etc.)

first_seen (string, yyyy-mm-dd_hh:mm:ss)

last_seen (string, yyyy-mm-dd_hh:mm:ss)
probability (float, correct recognition probability, between 0 and 1)
direction (int, 1 means that the vehicle is coming, 2 means that the vehicle is leaving, 3 = the direction cannot be determined)
plate_image (jpeg base64 encoded, licence plate photo)
car_image (jpeg base64 encoded, vehicle photo)
transactionkey (string, sole key for each licence plate, it can be used to logically link the two events)
gps_latitutde, gps_longitude (string , ex: 40.76) – is sent only if a GPS receiver is connected
have_companion (int, 1 if exists, 0 if there isn't defined any companion camera)
companion_image (jpeg base64 encoded, companion photo)
weight (float, weight received from industrial weighing machine)
speed (float, speed received from radar system)
auth (string, md5 generated using a private key, it can be used for authentication)

auth = md5(id + number + country_code + first_seen + last_seen + probability + transactionkey + direction + gps_latitude + gps_longitude + have_companion + weight + speed + reporting_check_action_authkey)

If the request is sent correctly, the external system should prompt with an approval message. For the reporting event, if there is no answer, the LPR engine will try to resend the data subsequently. The approval message must contain the string **fbd782b5b1f90875a9773ef20bcc16aa** for the action checking event and **bb1e8f805814a0b8e465601346872377** for the reporting event. În addition to this message, the answer for the reporting event may contain a list of actions to be done.
This is a string type list and can contain the values **open_barrier** and/or **open_barrier2** separated by spaces.

Each application has a log file where all actions and statuses are submitted. All log files are located into the metrici folder and each has a name composed of the id of the application and .log extension (ex: 0.log, 1.log and so on). Below it's an example of a transaction related to an external system located at the address: http://10.70.1.202/verif.php?loc=BU3&poarta=intrare_principala

2017-01-12 16:16:10 [Metrici LPR v4.9] Check action event for plate number B111LSI was sent to http://10.70.1.202/verif.php?loc=BU3&poarta=intrare_principala
2017-01-12 16:16:11 [Metrici LPR v4.9] Check action reply received: fbd782b5b1f90875a9773ef20bcc16aa open_barrier

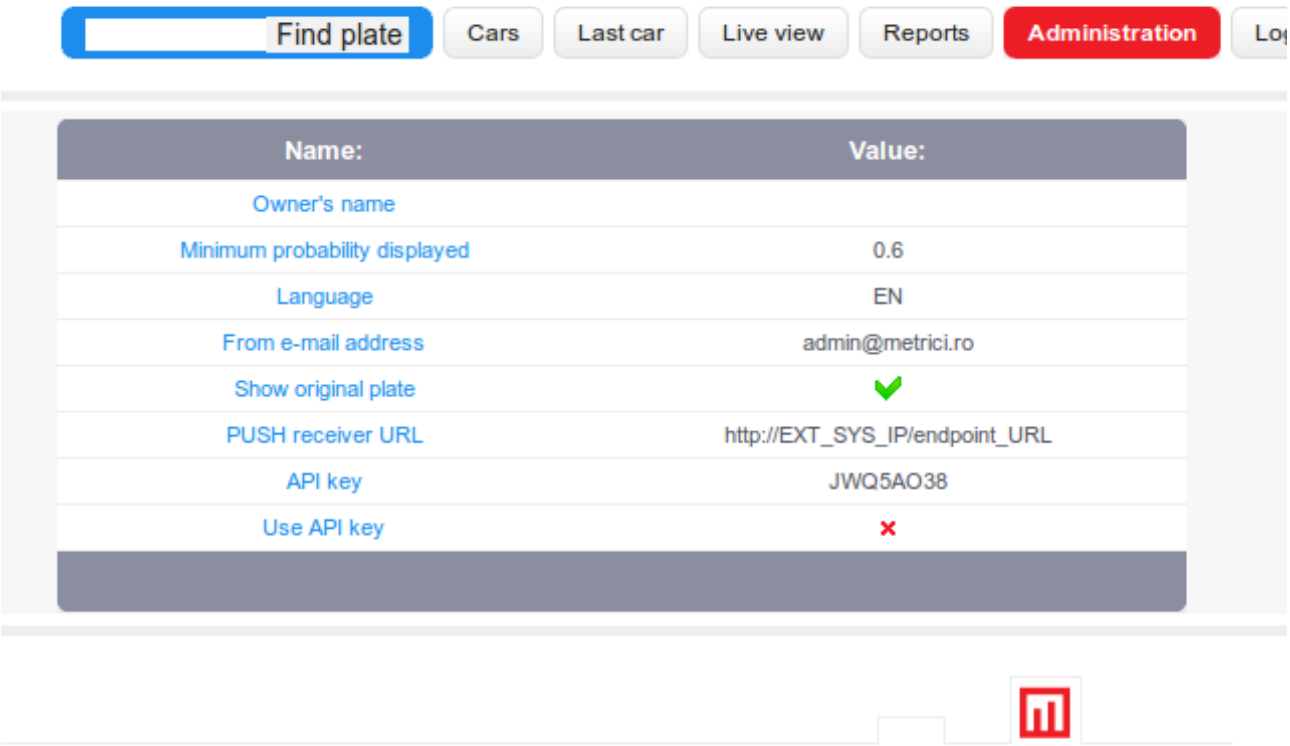
2017-01-12 16:16:11 [Metrici LPR v4.9] Open barrier command sent to http://10.10.16.233/rc.cgi?o=4,1
2017-01-12 16:16:11 [Metrici LPR v4.9] Open barrier reply received
2017-01-12 16:16:14 [Metrici LPR v4.9] Plate number B111LSI | RO | 2017-01-13_16:16:09 | 2017-01-13_16:16:10 | 1 was inserted into local buffer
2017-01-12 16:16:14 [Metrici LPR v4.9] Reporting event for plate number B71CWN was sent to http://localhost/io/new_plate_event.php

2.PUSH Connection from the reporting interface to the external system

This working mode is useful when one wishes to receive the events in both the reporting interface and the external system. The main drawbacks are the fact that the events reach the external system with a 1 ~ 2 sec. delay from the time of occurrence and that the external system cannot answer with an action, in case of recognising a certain licence plate.

The same as with the sending directly from the LPR engine, two events are generated for each vehicle / licence plate: IN and OUT. The first event occurs when the licence plate enters in the camera view, and the second when the vehicle leaves and is no longer visible in the image.

For each of these two events, the LPR engine sends the data by PUSH over HTTP, in a multipart/form-data format defined in RFC2388 (http://tools.ietf.org/html/rfc2388), to an URL, which can be defined in the application interface.



The data sent are the following:

plate_number (string, contains the licence plate, without spaces)

country_code (string, contains the country code, ex: RO, BG)

plate_probability (float, correct recognition probability, between 0 and 1)

time_stamp (string, type YYYY-MM-DD HH:MM:SS)

location_name (string, ex: Gate3)

camera-name (string, ex: Entrance2)

event-type (string, it can have the values IN (first seen) or OUT (out of the frame))

plate_image (jpeg base63 encoded, licence plate photo)

car_image (jpeg base64 encoded, vehicle photo)

3.Connection from the external system, by the API of the reporting interface

This working mode is useful for analyses and complex reports, subsequent to the occurrence of events, without being necessary to keep them in data base located on the external system.

Using GET type requests, over HTTP, to http://IP_OF_THE_REPORTING_INTERFACE/api, you can get data in the following formats: jSON (output=json), XML (output = xml) or DUMP (output = dump).
Example:

GET http://reports.metrici.ro/api/search.php?apikey=12345&plate=B72DAV&start=01-11-2016&end=02-11-2016&output=json

JSON RESPONSE:


```
{
    "error":false,
    "duration":"0.0704",
    "response":{
        "header":{
            "keyword":"B72DAV",
            "count":10,
        },
        "body":[
            {
                "id":1222830,
                "probability":"1",
                "plate_snapshot":"/1.jpg",
                "car_snapshot":"/car_image.php?id=1222830",
                "country":"RO",
                "direction":"COMING",
                "seen":"01-11-2016 09:02",
                "camera_id":"1",
                "camera_description":"Giurgiu",
                “camera_geo_location”:” 44.472991,26.137505”
                [...]
            }
            [...]
        ]
    }
```

Methods:

/search/ - used when one wished to search for a licence plate
entry parameters:

- plate (string, searched licence plate, no spaces, ex: B72DAV)
- start (string, first day of the search interval in DD-MM-YYY format, ex: 01-11-2014, if missing, it takes the default value: now () – 1 week interval)
- end (string, last day of the search interval in DD-MM-YYYY format, ex: 02-11-2014, if missing, it takes the default value, today)

Example:
http://reports.metrici.ro/api/search.php?apikey=12345&plate=B72DAV&start=01-11-2016&end=02-11-2016&output=xml

/list/ - used when you wish to obtain a list of events from a certain period
entry parameters:

- camera_id (string, list of camera ids separated by comma, if missing, the results from all cameras are obtained)
- start (string, first day of the searching period in DD-MM-YYYY format, ex: 01-10-2014, if missing, it takes the default value: now () / 7 days period)
- end (string, the last day of the search interval in DD-MM-YYYY format, ex: 02-11-2014, if missing, it takes the default value, today)

Example:
http://reports.metrici.ro/api/list.php?apikey=12345&start=01-10-2016&end=02-11-2016&camera_id=1,2&output=json

/stats/ - used when you wish to obtain the number of events from a certain period
entry parameters:

- camera_id (string, list with camera id’s separated by comma, if missing, the results from all cameras are obtained)
- start (string, first day of the searching period in DD-MM-YYYY format, ex: 01-10-2014, if missing, it takes the default value: now () / 7 days period)
- end (string, the last day of the search interval in DD-MM-YYYY format, ex: 02-11-2014, if

missing, it takes the default value, today)

Example:

http://reports.metrici.ro/api/stats.php?apikey=12345&start=01-10-2016&end=02-11-2016&camera_id=1,2&output=dump

/last_car/ - used when you wish to obtain the last recognized plate number

entry parameters:

- camera_id (string, list with camera id's separated by comma, if missing, the results from all cameras are obtained)

Example:

http://reports.metrici.ro/api/last_car.php?apikey=12345&camera_id=1,2&output=dump

/insert_action/ - used when you wish to insert an action into the Actions list

entry parameters:

- plate_number (is the plate number that must be inserted into Actions list)
- action (open_barrier or open_barrier2)
- camera_id (the id of the camera. If this parameter is missing or has a null value, the plate number will be added for all cameras connected into the system)
- expire (timestamp which defines the date and hour until the action is active. If this parameter is missing there will be no expiration)

Example:

http://reports.metrici.ro/api/insert_action.php?plate_number=VN100XFL&action=open_barrier&camera_id=&expire=

/delete_action/ - used when you wish to delete an action from the Actions list

entry parameters:

- plate_number (is the plate number that must be deleted from Actions list)

Example:

http://reports.metrici.ro/api/delete_action.php?plate_number=VN100XFL

/delete_expired/ - used when you wish to delete all expired number plates from Actions list (it's recommended to access it from time to time)
Example:
http://reports.metrici.ro/api/delete_expired.php

All API actions are logged into the web interface on behalf of api@metrici.ro user.

ANNEX 1 – Exemple of code, PUSH from the LPR engine, endpoint check action event

```
<?php

// v1.0
// this is given only as an example to see how one can parse and use POST
parameters reveived from Metrici LPR engine, at check action time

$ok_response= "fbd782b5b1f90875a9773ef20bcc16aa";
$msg= "";   //response message

//-------------------------------------
// retrieve parameters from POST

if (isset($_POST['id']))
{
      $id= (int)strip_tags($_POST['id']);
}
else
{
      $id= 0;
}

if (isset($_POST['number']))
{
      $number= strip_tags($_POST['number']);
}
else
{
      $number= "";
}

if (isset($_POST['country_code']))
{
      $country_code= strip_tags($_POST['country_code']);
}
else
{
      $country_code= "??";
}

if (isset($_POST['first_seen']))
{
      $first_seen= strip_tags($_POST['first_seen']);
}
else
{
      $first_seen= "";
}

if (isset($_POST['last_seen']))
{
      $last_seen= strip_tags($_POST['last_seen']);
}
else
```

```
{
	$last_seen= "";
}

if (isset($_POST['probability']))
{
	$probability= (float)strip_tags($_POST['probability']);
}
else
{
	$probability= 0;
}

if (isset($_POST['transactionkey']))
{
	$transactionkey= strip_tags($_POST['transactionkey']);
}
else
{
	$transactionkey= "";
}

if (isset($_POST['direction']))
{
	$direction= (int)strip_tags($_POST['direction']);
}
else
{
	$direction= 3;	//UNKNOWN
}

if (isset($_POST['auth']))
{
	$auth= strip_tags($_POST['auth']);
}
else
{
	$auth= "";
}

if ($id!= 0 && $number!= "" && $country_code!= "" && $first_seen!= "" &&
$last_seen!= "" && $probability!=0 && $auth!= "")	//not junk
{
	//... code  ...//
	//... retrieve authkey for this id, from your database
	//... code  ...//
	$authkey='XXXXXX';

	if($auth!= md5($id.$number.$country_code.$first_seen.$last_seen.
$probability.$transactionkey.$direction.$authkey))
	{
		$msg= "Error: unauthenticated request!";
	}
	else  //all good, authenticated request
	{
		$first_seen= str_replace("_", " ", $first_seen);
```

```
            $last_seen= str_replace("_", " ", $last_seen);

            $plate_image= file_get_contents($_FILES["plate_image"]["tmp_name"]);
            //$plate_image = mysql_real_escape_string($plate_image);   //use it
only when you want to insert the image into database

            $car_image= file_get_contents($_FILES["car_image"]["tmp_name"]);
            //$car_image = mysql_real_escape_string($car_image); //use it only
when you want to insert the image into database

            //... code  ...//
            //... insert new data into your database
            //... code ...//

            $msg= $ok_response;

            //... code  ...//
            //... verify if there is an action for this plate number, ex:
open_barrier
            //... code ...//

            $msg.= ' open_barrier'; //... if so, append this action to the
ok_response, leaving a space between
      }
}

echo $msg;  //send feedback to LPR engine

?>
```

ANNEX 2 – Exemple of code, PUSH from the LPR engine, endpoint reporting event

```
<?php

// v1.0
// this is given only as an example to see how one can parse and use POST
parameters reveived from Metrici LPR engine

$ok_response= "bb1e8f805814a0b8e465601346872377";
$msg= "";   //response message

//-------------------------------------
// retrieve parameters from POST

if (isset($_POST['id']))
{
      $id= (int)strip_tags($_POST['id']);
}
else
{
      $id= 0;
}

if (isset($_POST['number']))
{
      $number= strip_tags($_POST['number']);
}
else
{
      $number= "";
}

if (isset($_POST['country_code']))
{
      $country_code= strip_tags($_POST['country_code']);
}
else
{
      $country_code= "??";
}

if (isset($_POST['first_seen']))
{
      $first_seen= strip_tags($_POST['first_seen']);
}
else
{
      $first_seen= "";
}

if (isset($_POST['last_seen']))
{
      $last_seen= strip_tags($_POST['last_seen']);
}
else
```

```
{
	$last_seen= "";
}

if (isset($_POST['probability']))
{
	$probability= (float)strip_tags($_POST['probability']);
}
else
{
	$probability= 0;
}

if (isset($_POST['transactionkey']))
{
	$transactionkey= strip_tags($_POST['transactionkey']);
}
else
{
	$transactionkey= "";
}

if (isset($_POST['direction']))
{
	$direction= (int)strip_tags($_POST['direction']);
}
else
{
	$direction= 3;	//UNKNOWN
}

if (isset($_POST['auth']))
{
	$auth= strip_tags($_POST['auth']);
}
else
{
	$auth= "";
}

if ($id!= 0 && $number!= "" && $country_code!= "" && $first_seen!= "" &&
$last_seen!= "" && $probability!=0 && $auth!= "")	//not junk
{
	//... code  ...//
	//... retrieve authkey for this id, from your database
	//... code  ...//
	$authkey='XXXXXX';

	if($auth!= md5($id.$number.$country_code.$first_seen.$last_seen.
$probability.$transactionkey.$direction.$authkey))
	{
		$msg= "Error: unauthenticated request!";
	}
	else  //all good, authenticated request
	{
		$first_seen= str_replace("_", " ", $first_seen);
```

```
            $last_seen= str_replace("_", " ", $last_seen);

            $plate_image= file_get_contents($_FILES["plate_image"]["tmp_name"]);
            //$plate_image = mysql_real_escape_string($plate_image);   //use it
only when you want to insert the image into database

            $car_image= file_get_contents($_FILES["car_image"]["tmp_name"]);

            //$car_image = mysql_real_escape_string($car_image); //use it only
when you want to insert the image into database

            //... code  ...//
            //... verify if there is another row into database containing the
same transactionkey
            //... if so, update that row with new values
            //... if not, insert new data
            //... code ...//

            $msg= $ok_response;
      }
}

echo $msg;  //send feedback to LPR engine

?>
```

ANNEX 3 – Exemple of code, PUSH from the LPR engine, send event

```
<?php

// v1.0
// this is given only as an example to see how one can send POST events to
Metrici LPR web interface

$ok_response= "bb1e8f805814a0b8e465601346872377";
$boundary= 'someboundary';
$push_receiver_url= 'http://Metrici_web_interface_IP/io/new_plate_event.php';

//-------------------------------------------------------
// retrieve first unsent event from your database

//calculate auth
$auth= md5($id.$number.$country_code.$first_seen.$last_seen.$probability.
$transactionkey.$direction.$authkey);

//build post data

$post_data= '';

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="id"';
$post_data.= "\r\n\r\n";
$post_data.= $id;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="number"';
$post_data.= "\r\n\r\n";
$post_data.= $number;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="country_code"';
$post_data.= "\r\n\r\n";
$post_data.= $country_code;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="first_seen"';
$post_data.= "\r\n\r\n";
$post_data.= $first_seen;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="last_seen"';
$post_data.= "\r\n\r\n";
$post_data.= $last_seen;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="probability"';
```

```
$post_data.= "\r\n\r\n";
$post_data.= $probability;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="transactionkey"';
$post_data.= "\r\n\r\n";
$post_data.= $transactionkey;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="direction"';
$post_data.= "\r\n\r\n";
$post_data.= $direction;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="auth"';
$post_data.= "\r\n\r\n";
$post_data.= $auth;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="plate_image";
filename="plate_image"';
$post_data.= "\r\n";
$post_data.= "Content-Type: image/jpeg";
$post_data.= "\r\n\r\n";
$post_data.= $plate_image;
$post_data.= "\r\n";

$post_data.= "--".$boundary."\r\n";
$post_data.= 'Content-Disposition: form-data; name="car_image";
filename="car_image"';
$post_data.= "\r\n";
$post_data.= "Content-Type: image/jpeg";
$post_data.= "\r\n\r\n";
$post_data.= $car_image;
$post_data.= "\r\n";

$post_data.= "--".$boundary."--\r\n";

$ch = curl_init($push_receiver_url);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER , array('Content-Type: multipart/form-data;
boundary='.$boundary, 'Content-Length: '.strlen($post_data)));
curl_setopt($ch, CURLOPT_POSTFIELDS, $post_data);

$result= curl_exec($ch);

if ($result!== false)
{
      if ($result== $ok_response)   //data from POST was accepted
      {
            //... code  ...//
```

```
            //... update your database to change the status of this event from
unsent to sent
            //... code ...//
      }
}

?>
```