|  |  |
| --- | --- |
|  |  |
| **Connected Vehicle Data Privacy Protection**  **Module** |  |
|  | May 3, 2017 |

This page intentionally left blank

# Revision History

|  |  |  |
| --- | --- | --- |
| Revision | Date | Changes |
| 0.1 | April 24, 2017 | * Initial Version |
| 0.2 | May 3, 2017 | * Added basic installation information; Added parameter explanations; updated language to reflect new name PPM. |

Table of Contents

[Revision History 3](#_Toc481577801)

[Table of Figures 5](#_Toc481577802)

[The Operational Data Environment (ODE) Privacy Protection Module (PPM) Background 6](#_Toc481577803)

[PPM Purpose 6](#_Toc481577804)

[PPM Limitations 6](#_Toc481577805)

[Architecture and Interaction with the ODE 7](#_Toc481577806)

[Code and Testing Environment 7](#_Toc481577807)

[Development Methodology 7](#_Toc481577808)

[Obtaining the Code 7](#_Toc481577809)

[Code Documentation 8](#_Toc481577810)

[Details about Dependencies 8](#_Toc481577811)

[Building the PPM and the Testing Environment 8](#_Toc481577812)

[Building the PPM 8](#_Toc481577813)

[Running and Testing the PPM 8](#_Toc481577814)

[Configuring the PPM 9](#_Toc481577815)

[Configuration File and Parameters 9](#_Toc481577816)

[ODE and Kafka Parameters 9](#_Toc481577817)

[Velocity Fiter Parameters 9](#_Toc481577818)

[BSM Temporary ID Redaction Parameters 10](#_Toc481577819)

[Geofencing Parameters 10](#_Toc481577820)

[Map File Specification 11](#_Toc481577821)

[Additional PPM Operational Details 12](#_Toc481577822)

[PPM Data Inputs and Outputs (Kafka Topics) 12](#_Toc481577823)

[Important Privacy Considerations and Limitations 12](#_Toc481577824)

[Appendix A: Glossary of Terms / Acronyms 14](#_Toc481577825)

# Table of Figures

[Figure 1: Interface between ODE and De-Identification 7](file:////Users/x4j/Documents/repos/jpo-cvdp/docs/ppm_user_manual.docx#_Toc481577723)

[Figure 2: Geofence around map segment 12](#_Toc481577724)

# The Operational Data Environment (ODE) Privacy Protection Module (PPM) Background

The United States Department of Transportation Joint Program Office (JPO) Connected Vehicle Data Privacy (CVDP) Project is developing a variety of methods to enhance the privacy of individuals who generated connected vehicle data. Connected vehicle technology uses in-vehicle wireless transceivers to broadcast and receive basic safety messages (BSMs) that include accurate spatiotemporal information to enhance transportation safety. Integrated Global Positioning System (GPS) measurements are included in BSMs. Databases, some publicly available, of BSM sequences, called trajectories, are being used to develop safety and traffic management applications. **BSMs do not contain explicit identifiers that link trajectories to individuals; however, the locations they expose may be sensitive and associated with a very small subset of the population; protecting these locations from unwanted disclosure is extremely important.** Developing procedures that minimize the risk of associating trajectories with individuals is the objective of this project.

# PPM Purpose

This module enhances the privacy protection provided to data generators participating in Connected Vehicle Safety Pilots. The PPM operates on streams of raw BSMs generated by the ODE. It determines whether each individual BSM should be retained or suppressed (deleted) based only on the information in that BSM, a well-defined geographical boundary (i.e., geofence), and vehicle speed. Additionally, the procedure redacts the BSM identifier for a specified set of vehicles.

By eliminating BSMs whose speeds are below or above prescribed speed limits, messages that may be associated with accidents or violation of posted speed limits are suppressed.

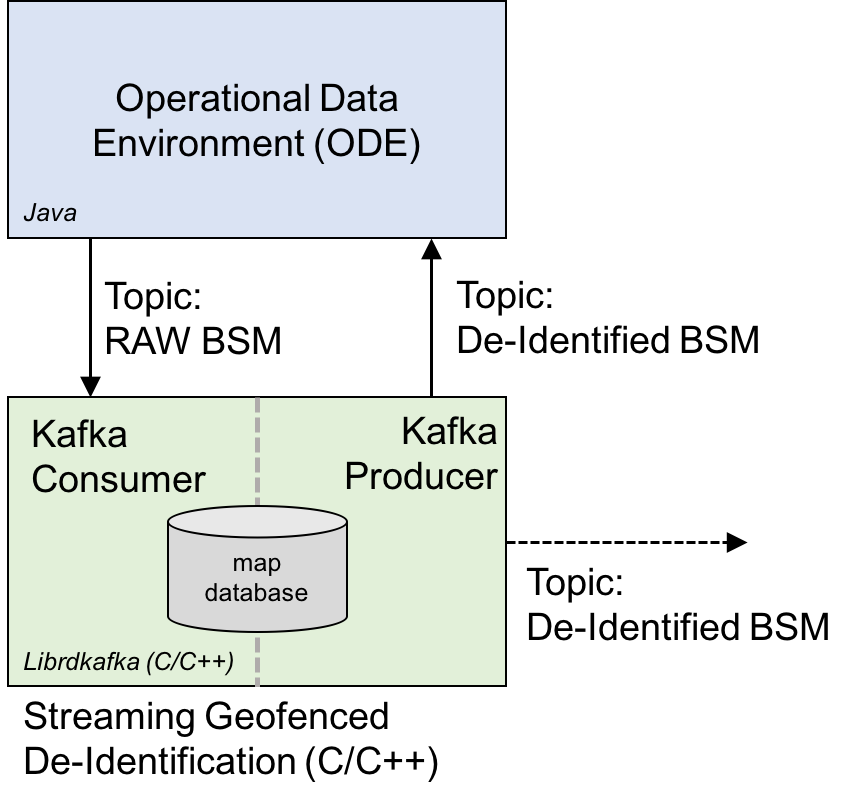
By eliminating BSMs whose locations are outside of a prescribed geofence, messages that may be used to learn information that could be used to identify the identity of an individual driver are suppressed. It is important to note this protection is predicated on the geofence describing an area where loitering or stopping at locations that can be used to infer personally identifying information, e.g., homes or businesses.

This module is specifically designed to handle data streams gathered by the Operational Data Environment (ODE).

# PPM Limitations

Protecting against inference-based privacy attacks on spatiotemporal trajectories (i.e., sequences of BSMs from a single vehicle) in general is a challenging task. An example of an inference-based privacy attack is identifying the driver that generated a sequence of BSMs using specific locations they visit during their trip, or other features discernable from the information in the BSM sequence. **This PPM treats a specific use case: a geofenced area where residences do not exist, e.g., a highway corridor, with certain speed restrictions**. **Do not assume this strategy will provide sufficient privacy protection in other uses cases.** There are alternative strategies that must be employed to handle cases where loitering locations can aid in learning the identity of the driver.

# Architecture and Interaction with the ODE



#### *Figure 1: Interface between ODE and De-Identification*

This module performs its function as a separate compute process. It interacts with the ODE through the distributed streaming data platform, Kafka. The privacy module subscribes to an ODE topic that streams raw BSMs in JSON format. The privacy module filters each BSM it receives in this stream. It then publishes the BSMs it retains, with designated fields redacted, to a new Kafka topic. This simple architecture makes the privacy module independent of the ODE. Figure 1 illustrates the basic interactions between the privacy module and the ODE.

# Code and Testing Environment

## Development Methodology

The Privacy Protection Module (PPM) is being developed under Agile Development Methodologies, using an open architecture approach, in an open source environment. All stakeholders are invited to provide input to this document and the PPM. Stakeholders should direct all input on this document to the JPO Product Owner at DOT, FHWA, JPO.

This document is a living document and will be updated throughout the life of the JPO ODE project that this module is designed to support to reflect the most recent changes in the ODE design and stakeholder feedback.

## Obtaining the Code

PPM code and additional documentation is available at: <https://github.com/usdot-jpo-ode/jpo-cvdp>

PPM code and its test environment is maintained as a Git repository. The Git source code version control system and available clients is described and can be obtained at the following website: <https://git-scm.com/downloads>.

The PPM repository includes everything you need to build and test the code. The information sources below provide much more detail on PPM dependencies.

## Code Documentation

The privacy module code is documented using [Doxygen](http://www.stack.nl/~dimitri/doxygen/) and attempts to follow [Google’s C++ Style Guide](https://google.github.io/styleguide/cppguide.html). There are instructions for building the code documentation on the jpo-cvdp Github page.

## Details about Dependencies

The privacy module is written in C and C++11. It uses the following open source libraries:

* Librdkafka (<https://github.com/edenhill/librdkafka)>: an open source (BSD License) C/C++ Kafka client. Librdkafka is an active open source project with many contributors. It has been used in commercial settings.
* RapidJSON (<https://github.com/miloyip/rapidjson)>: an open source (MIT License) C++ Javascript Object Notation (JSON) parser. The library is one of the fastest open-source JSON parsers available and it is fully compliant with the RFC7159/ECMA-404.

# Building the PPM and the Testing Environment

In this manual, very basic instructions are provided to build, run, and test the PPM. This manual explains PPM configuration parameters and operations. We highly recommend using the more comprehensive build, run, and test instructions provided on the [Github page](https://github.com/usdot-jpo-ode/jpo-cvdp) for guidance on those processes.

## Building the PPM

The PPM code includes a copy of RapidJSON; however, librdkafka (see Dependencies and links to instructions above) must be installed. Also, the PPM is built using [CMake](https://cmake.org/download/). Follow the link and install that package prior to building the PPM. Once these dependencies are satisfied, the following instructions can be used to build the PPM:

|  |
| --- |
| $ cd $BASE\_WORK\_DIR  $ git clone https://github.com/usdot-jpo-ode/jpo-cvdp.git  $ cd jpo-cvdp  $ mkdir build && cd build  $ cmake ..  $ make |

## Running and Testing the PPM

The PPM is intended to be run as a service that received data from and provides data to the ODE. It communicates using Kafka. As such, it is challenging to run and test as a stand-alone application. On the [Github page](https://github.com/usdot-jpo-ode/jpo-cvdp), there are detailed instructions outline how to standup a test environment without the ODE. Two methods are outlined: one installs all the necessary components as part of the core operating system, and the other uses Docker containers to make things easier.

## Configuring the PPM

### Configuration File and Parameters

The PPM configuration file is a text file with a specific format. Settings in this configuration file can be used to configure the Kafka service as well as PPM operations. The Kafka-specific parameters that pertain to the PPM are discussed here; more [Kafka parameters](https://kafka.apache.org/documentation/#configuration) can be found in the [Kafka documentation](https://kafka.apache.org/documentation/).

Comments are annotated in the configuration file by starting the comment with the '#' character. Configuration lines consist of two strings separated by a '=' character; lines are terminated by newlines. Configuration files names do not follow a specific convention; however, example configuration files use the file extension, ‘.properties’ In the PPM repository, example configuration files can be found in the config directory. The following is an example of a portion of a configuration file:

|  |
| --- |
| # Configuration details for privacy ID redaction.  privacy.redaction.id=ON  privacy.redaction.id.value=FFFFFFFF  privacy.redaction.id.inclusions=ON  privacy.redaction.id.included=BEA10000,BEA10001 |

The details of the settings and how they affect the function of the PPM follow:

### ODE and Kafka Parameters

* **privacy.topic.consumer**: The Kafka topic name used by the ODE (or other BSM JSON producer) consumed by the PPM. The source of the data stream to be filtered by the PPM.
* **privacy.topic.producer**: The Kafka topic name used by the PPM to write filtered BSMs.
* **group.id**: The group identifier for the PPM consumer. Consumers label themselves with a consumer group name, and each record published to a topic is delivered to one consumer instance within each subscribing consumer group. Consumer instances can be in separate processes or on separate machines.
* **privacy.kafka.partition**: The partition(s) that this PPM will consume records from. A Kafka topic can be divided, or partitioned, into several "parallel" streams. A topic may have many partitions so it can handle an arbitrary amount of data.
* **metadata.broker.list**: This is the IP address of the Kafka topic broker leader.
* **compression.type**: The type of compression to use for writing to Kafka topics. Currently, this should be set to none.

### Velocity Fiter Parameters

* **privacy.filter.velocity**: Enable or disable BSM speed filtering.
  + ON : Enables BSM filtering.
  + Any other value : Disables BSM filtering.
* **privacy.filter.velocity.min**: When velocity fitering is enabled, BSMs having velocities below this value will be suppressed. The units are in meters per second.
* **privacy.filter.velocity.max**: When velocity fitering is enabled, BSMs having velocities above this value will be suppressed. The units are in meters per second.

### BSM Temporary ID Redaction Parameters

BSM have a temporary ID field. A practitioner can use temporary ID redaction parameters to control which IDs are redacted. A user can choose not to redact identifiers, redact all identifiers, or redact specific identifiers. To achieve this two levels of control provided: The first level tells the PPM whether IDs should be redacted. The second level is only active if the first level is turned on; it allows the user to specify in more detail which IDs should be redacted.

* **privacy.redaction.id**: Enable or disable the PPM's redaction function. NOTE: currently we are redacting the BSM TemporaryID field as specified in the J2735.
  + ON : enables redaction
  + Any other value : **The PPM will NOT redact BSM identifiers**.
* **privacy.redaction.id.value**: If redaction is enabled, this value will replace the current id value in the raw BSM.
  + According to the J2735, this value is 4 hexidecimal-encoded bytes. The configured value should **NOT** be enclosed in quotes or be preceded by 0x, e.g., AAAAAAAA
* **privacy.redaction.id.inclusions**: If redaction is enabled, this parameter enables or disables selective id redaction.
  + ON : The PPM will use a redaction inclusion set. If a BSM is received having an id in this list, that id will be replaced with the privacy.redaction.id.value. The values in this set are defined in the privacy.redaction.id.included configuration parameter.
  + Any other value : **The PPM will redact the identifiers in all received BSMs**. In essence, the inclusion set defaults to the set of all possible 4 byte values.
* **privacy.redaction.id.included**: If redaction and redaction inclusions are enabled, this comma separated list specifies the BSM identifiers that will be redacted. BSMs having identifiers that are not in this set will not be modified.
  + Specify these ids as 4 hexadecimal-encoded byte values without quotes or prefixed with 0x.
  + Multiple identifiers can be specified by separated individual identifiers with a comma.

### Geofencing Parameters

The geofence parameters allow the user to turn geofencing on or off, specify a mapfile used to define the geofence, and further refine the geofence using a bounding box. **The geofence is not a simple rectangle or circle**. It is a complex shape determined by many individual road segments that define where vehicles of interest will be traveling. BSMs transmitted by vehicles that are not on these road segments are suppressed.

* privacy.filter.geofence : enables or disables geofence-based filtering.
  + ON : enables the geofence.
  + Any other value : disables geofence filtering.
* privacy.filter.geofence.mapfile : If redaction is enabled, specifies the absolute or relative path and filename of a file that contains the map information needed to define the geofence.

The geofence is stored in a geographically-defined data structured called a quadtree. The following bounding box coordinates define the quadtree's region. The data that is stored in the quadtree is limited to those segments provided in a mapfile that are contained or intersect the bounding box region. As an example of this relationship, the coordinates specified below could bound the entire state of Wyoming; however, only the segments for the I-80 corridor would be stored within a quadtree covering Wyoming since the map file is restricted to those segments. One the other hand, the coordinates below could specify a smaller section of I-80. Then, only that subset of segments in the map file would be inserted into the quadtree and used for geofencing; the others would be ignored.

* **privacy.filter.geofence.sw.lat**: The latitude of the lower-left corner of the quadtree region.
* **privacy.filter.geofence.sw.lon**: The longitude of the lower-left corner of the quadtree region.
* **privacy.filter.geofence.ne.lat**: The latitude of the upper-right corner of the quadtree region.
* **privacy.filter.geofence.ne.lon**: The longitude of the upper-right corner of the quadtree region.

### Map File Specification

A file containing geographic elements is used to define the geofence. The file can accommodate a variety of shapes; however, in this application edges are specified. Each edge in the map file defines a single straight segment of a road. Geographically, roads have complex shapes; so, many straights edges of varying distances may be used to define a single road. Each edge is used to define a rectangular region that acts as a small “component” of the larger geofence. The map file for the I-80 WYDOT corridor is located in the jpo-cvdp/data directory; it is named: I\_80.edges

The following is a small portion of the I\_80.edges file:

|  |
| --- |
| **type,id,geography,attributes**  edge,0,0;41.24789403;-111.0467118:1;41.24746145;-111.0455124,way\_type=user\_defined:way\_id=80  edge,1,1;41.24746145;-111.0455124:2;41.24733395;-111.0451337,way\_type=user\_defined:way\_id=80  edge,2,2;41.24733395;-111.0451337:3;41.24726205;-111.044904,way\_type=user\_defined:way\_id=80  edge,3,3;41.24726205;-111.044904:4;41.24713975;-111.0444827,way\_type=user\_defined:way\_id=80 |

This file has four comma-separated elements:

* **type**: edge is used for this application.
* **id**: a unique 64-bit integer shape identifier
* **geography**: A sequence of colon-split triples representing points; each point is semi-colon split as follows: <point uid>;<latitude>;<longitude>
* **attributes**: A sequence of colon-split key=value pairs.
  + The attribute way\_type determines the dimensions of the geofence surrounding each edge.

For the WYDOT use case, WYDOT provided a set of edge definitions for I-80 that were converted into the above format.

# Additional PPM Operational Details

This application will perform rule-based de-identification. In other words, it will suppress/retain individual BSM records (Part I and optionally Part II) and redact values assigned to certain fields. Suppression will be based on the following BSM Part I fields:

* Geoposition (i.e., latitude and longitude)
* Speed

The following fields will have their values redacted in an agreed upon way (replace the value with a well-defined “null” value):

* Static vehicle identifiers.

The fields selected to perform suppression and redaction could be refined.

BSM records will be geofenced using the geoposition attribute, an auxiliary map database, and explicit geofences based on map road segments. The map database will contain road segments specified by WYDOT (e.g., initially the I-80 corridor). Map information can be obtained from a variety of sources. Currently, the PPM is using data provided by WYDOT. OpenStreamMap data has been used in the past. Geofences will be parameterized and the overall geofence region can be altered by adding or removing segments.



Figure 2: Geofence around map segment

Figure 2 illustrates how a “component” of the larger geofence is defined based on a single map segment (here this is shown as an OSM edge, but any two points will work). Map file attribute elements are used to define the width and length of the rectangle surrounding the map segment. Currently, a way\_type of undefined triggers a width and extension specifically designed for the WYDOT use case. This can be easily modified to fit a broader range of use cases and segments types.

## PPM Data Inputs and Outputs (Kafka Topics)

The PPM subscribes to an ODE topic to obtain input (e.g., a Raw BSM stream). The PPM produces BSMs to another topic. The output topic will have the same format as the input records with redaction / suppression occurring as described above.

## Important Privacy Considerations and Limitations

The following points regarding data privacy and this system are worth considering:

* **As noted earlier in this document, the rule-based approach described here does not provide adequate privacy protection in general**. In this case, the geofenced roads do not directly connect to locations generally considered private (e.g., homes or businesses). Velocities outside of a well-defined dynamic range are considered abnormal within this geofence; therefore, we assume that behavior is due to traffic or drivers exceeding posted speed limits. The velocity rules are designed to protect against identifying vehicles involved in accidents and breaking the law.  
    
  If the geofence described here (i.e., I-80) is expanded to include additional roads, or BSM databases that geographically intersect this geofence are made available, additional privacy protections are needed to avoid inference-based attacks. In general, once data has been publically released it is not recallable.
* As previously discussed, the ODE will produce topics that are “sanitized” and others that do not provide any privacy protection. Security controls (authentication) will be needed to ensure arbitrary subscribers cannot gain access to “unsanitized” topics.
* The abstractions (i.e., how and where data is stored and distributed) provided by big-data platforms like Kafka make them extremely useful and easy to scale. Understanding who has access to specific topic partitions in memory and on disk should be considered especially since this data is replicated and distributed. Theses resource could be accessible without using any of the ODE’s / Sanitization interfaces.

# Appendix A: Glossary of Terms / Acronyms

|  |  |
| --- | --- |
| Term / Acronym | Description |
| BSM | Basic Safety Message |
| PPM | Privacy Protection Module |
| ODE | Operational Data Environment |
| JSON | Javascript Object Notation |
| GPS | Global Positioning System |
| CVDP | Connected Vehicle Data Privacy |
| WYDOT | Wyoming Department of Transportation |
| JPO | Joint Program Office |
|  |  |
|  |  |
|  |  |