# Connected and Automated Vehicle education (CAVe)-Lite

## System Design Document

**June 3, 2022**

U.S. Department of Transportation

Produced under Saxton Transportation Operations Laboratory Task Order 19-419 (Development of Workforce Education Materials and Equipment), DTFH6116D00030L - 693JJ319F000419

U.S. Department of Transportation
Office of the Assistant Secretary for Research and Technology
Federal Highway Administration

# TECHNICAL REPORT DOCUMENTATION PAGE

| | | |
|---|---|---|
| **1. Report No.**<br>FHWA-JPO-22-xxx | **2. Government Accession No.** | **3. Recipient's Catalog No.** |
| **4. Title and Subtitle**<br>Connected and Automated Vehicle education (CAVe)-Lite System Design Document | | **5. Report Date**<br>May 24, 2022<br>**6. Performing Organization Code** |
| **7. Author(s)**<br>Will Martin, Thomas Bayhi, Steven Vu, Anjan Rayamajhi, Safak Ercisli, Animesh Balse | | **8. Performing Organization Report No.** |
| **9. Performing Organization Name and Address**<br>Leidos<br>11251 Roger Bacon Drive<br>Reston, VA 20190 | | **10. Work Unit No. (TRAIS)**<br>**11. Contract or Grant No.** |
| **12. Sponsoring Agency Name and Address**<br>U.S. Department of Transportation<br>Federal Highway Administration<br>Saxton Transportation Operations Laboratory<br>6300 Georgetown Pike<br>McLean, VA 22101 | | **13. Type of Report and Period Covered**<br>System Design Document<br>May, 2022<br>**14. Sponsoring Agency Code**<br>HRDO |
| **15. Supplementary Notes**<br>Deb Curtis, GTM | | |

**16. Abstract**

Connected and automated vehicle education Lite (CAVe-Lite) is a tool that enables researchers at universities and other educational institutions to test and validate connected vehicle (CV) messages. CAVe-Lite is a tool that removes the traffic signal controller (TSC) and other larger components from the design to create an ultra-portable CV testing platform that focuses on application development. The platform has been developed on a raspberry pi using a series of docker containers and initialization scripts for a quick and cost-effective setup. The primary mode of data transmission is through Wi-Fi, as it is the most easily accessible wireless network in indoor environments. The data from all other crucial intelligent transportation systems (ITS) components like TSC and CV radio are provided through various scripts running on the compute node. This CAVe-Lite system design document details the various software and hardware components, the data flows, and the configuration options required to test the available use cases. The purpose for this document is to enable researchers to build their own CAVe-Lite kits and include CV application development and testing to their transportation curriculums.

| | |
|---|---|
| **17. Key Words**<br>Intelligent Transportation Systems, Curriculum Development, Connected Vehicles, Edge Computing | **18. Distribution Statement**<br>No restrictions. |

| **19. Security Classif. (of this report)** | **20. Security Classif. (of this page)** | **21. No. of Pages** | **22. Price** |
|---|---|---|---|
| Unclassified | Unclassified | 31 | N/A |

TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

## LIST OF ABBREVIATIONS

| | |
|---|---|
| AC | alternating current |
| API | application programming interface |
| BSM | basic safety message |
| C-V2X | cellular vehicle-to-everything |
| CAN | Controller Area Network |
| CAV | connected and automated vehicle |
| CAVe | Connected and Automated Vehicle education |
| CAVERS | Connected and Automated Vehicle Education Research Software |
| CV | connected vehicle |
| DC | direct current |
| DHCP | Dynamic Host Configuration Protocol |
| DMS | dynamic message sign |
| DSRC | dedicated short-range communications |
| FHWA | Federal Highway Administration |
| GPS | Global Positioning System |
| GUI | graphical user interface |
| HDMI | High-Definition Multimedia Interface |
| HIL | hardware-in-the-loop |
| HSM | hardware security module |
| IP | Internet Protocol |
| ITS | intelligent transportation system |
| ITS JPO | Intelligent Transportation Systems-Joint Program Office |

| | |
|---|---|
| JSON | Java Script Object Notation |
| LiDAR | Light Detection and Ranging |
| LTE | Long-Term Evolution |
| NTCIP | National Transportation Communications for ITS Protocol |
| OBD | Onboard Diagnostics |
| OBU | onboard unit |
| OID | object identifier |
| PC | personal computer |
| PCAP | packet capture |
| PoE | Power over Ethernet |
| PSID | Provider Service Identifier |
| PSM | personal safety message |
| RSU | roadside unit |
| SAE | Society of Automotive Engineers |
| SDX | Situation Data Exchange |
| SNMP | Simple Network Management Protocol |
| SPaT | Signal Phase and Timing |
| SRM | Signal Request Message |
| SSH | Secure Shell |
| STOL | Saxton Transportation Operations Laboratory |
| TFHRC | Turner-Fairbank Highway Research Center |
| TIM | traveler information message |
| TMC | traffic management center |

| | |
|---|---|
| TSC | traffic signal controller |
| UDP | User Datagram Protocol |
| UPER | unaligned packet encoding rules |
| USB | Universal Serial Bus |
| USDOT | United States Department of Transportation |
| VAC | volts alternating current |
| V2I | vehicle-to-infrastructure |
| V2X | vehicle-to-everything |
| WAVE | Wireless Access in Vehicular Environments |
| WSA | WAVE Service Advertisement |
| WZDx | Work Zone Data Exchange |
| XML | Extensible Markup Language |

# CHAPTER 1. INTRODUCTION

This system design document serves as a companion to the initial Connected and Automated Vehicle education (CAVe)-in-a-box document[1]. CAVe-Lite is a scaled-down version of the original kit for easier access to existing educational material for intelligent transportation systems (ITS). Development of these materials have been tasked by the Intelligent Transportation Systems Joint Program Office (ITS JPO) and Federal Highway Administration (FHWA) for development by the Saxton Transportation Operations Laboratory (STOL).

CAVe-Lite is the result of feedback from alpha testers of CAVe-in-a-Box. This document aims to address those additionally identified gaps in knowledge expressed by users. Similar to CAVe-in-a-Box, the purpose of CAVe-Lite is to allow learning of vehicle-to-everything (V2X) topics and applications. This documentation focuses on use of CAVe-Lite with signal phase and timing (SPaT) messages, and a vehicular application that uses this message type. Users who may benefit from this material include future ITS technicians studying at community colleges and trade schools, beginners working with ITS vendors, bench testers at certification labs, and university students and researchers studying transportation.

This material varies from CAVe-in-a-Box, as it will be most useful as an introduction to the high-level functions of ITS equipment, while not covering the details of the equipment itself. It also addresses an identified knowledge gap in Linux and computer networking, which will assist with using the greater CAVe-in-a-box system.

## BACKGROUND

As connected and automated vehicle (CAV) technologies transition from research and pilot applications to large-scale deployments, installation, setup, and maintenance of devices and systems will need to be performed by technicians and engineers on a large scale. The training used for deploying CAV technologies must be presented with manifold resources to a diverse audience of learners. CAVe-Lite is a training resource to help the emerging workforce learn the basic skillsets needed to understand and interact with CAV technologies and their applications.

While later material covers information on V2X wireless communication technologies, such as dedicated short-range communications (DSRC) and cellular vehicle-to-everything (C-V2X), this documentation focuses on the data flow through an ITS network. In CAVe-Lite, this flow of information is facilitated using a Wi-Fi network. Wi-Fi is one of the most commonly used methods of wireless communication. This provides an opportunity for readers of this documentation to more easily grasp application layer concepts, as Wi-Fi shares the same compatibility of physical and link layer standards for computer networking as the other V2X wireless communication technologies. This document does not recommend using any specific wireless communication technology, but rather leverages the technology agnostic nature of the application stack to enable continued research using available resources.

[1] URL: https://www.pcb.its.dot.gov/documents/CAVe_Box_SysDes_April2022.pdf

## OBJECTIVES

The objectives of this document are as follows:

1. Introduce CAVe-Lite as a technical tool for educational purposes.
2. Provide detailed documentation of all aspects related to CAVe-Lite.
3. Provide information on software and network requirements, including testing and debugging.
4. Provide a use-case related to CAVe-Lite.

The future workforce in the CAV space should be familiar with the required communication protocols, embedded system concepts, and standards that make up ITS. CAVe-Lite is focused on providing the tools and materials to enable the teaching of these concepts and can allow students and educators to test CAV applications, such as:

- SPaT applications, that show how messages may be used to provide an efficient method of communication between various infrastructure components to be translated and broadcast to oncoming vehicles.
- Basic safety message (BSM) applications, to study how vehicles transmit their vehicle data, including but not limited to their location, speed, and heading.

Details of the components of each CAVe-in-a-box toolkit, the relevant standards and protocols, and the potential data flows can be found in the following chapters:

- Chapter 2 describes the system design of the toolkit.
- Chapter 3 focuses on the data flows in different components.
- Chapter 4 describes software configurations required to setup the kits.
- Chapter 5 highlights a few use cases for CAVe-Lite.
- Chapter 6 concludes and provides an overall summary of the toolkit.

# CHAPTER 2. SYSTEM DESIGN

CAVe-Lite is the trimmed-down, simulated version of CAVe-in-a-Box. The interconnected ITS components found in CAVe-in-a-box are replaced with various scripts that mirror their function. **This system is designed to further increase accessibility to educational materials by condensing all functionality onto a single piece of equipment, allowing for faster deployment and minimizing points of failure**.

CAVe-Lite contains the following features:

- A design with functionality in any location, where the system may be used anywhere without the need for a license to operate or a GPS signal.
- An extremely portable setup that can be installed on any single-board computer.
- Software that mirrors the functionality of standard ITS equipment.
- A single design that can be modified to function as either:
    - Infrastructure kit.
    - Mobile kit.
- Available updates and extension capabilities.
- An example application, demonstrating a real-world use case.

## CAVE-LITE INFRASTRUCTURE KIT

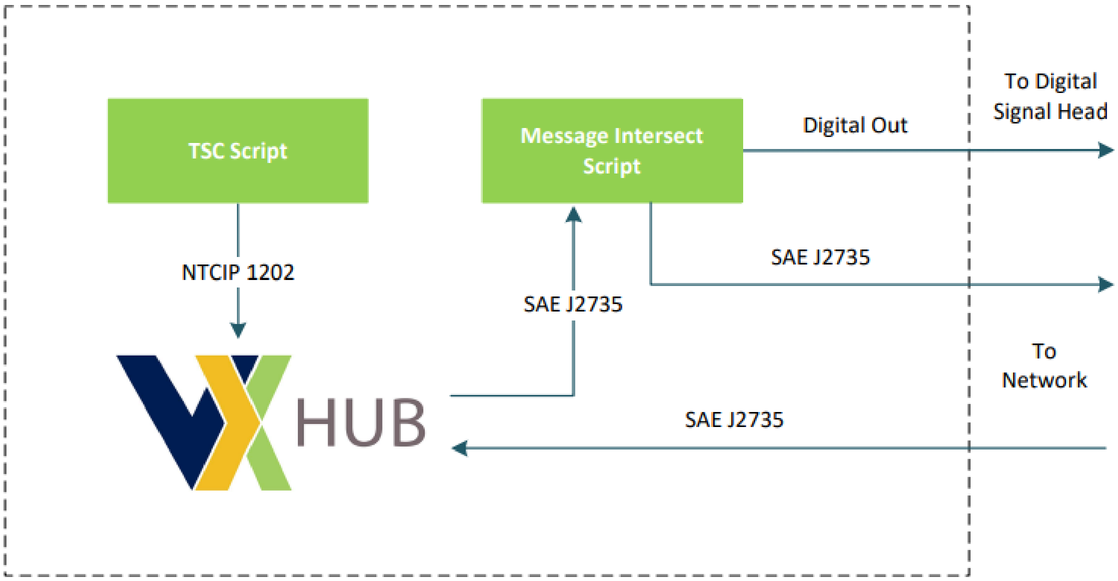
The infrastructure kit provides all functions of a connected intersection. This normally includes a traffic signal controller (TSC) that generates National Transportation Communications for ITS Protocol (NTCIP) 1202 messages and a DSRC or C-V2X roadside unit (RSU). V2X Hub, open-source software that handles Society of Automotive Engineers (SAE) J2735 messages, is used to receive the NTCIP 1202 data stream and converts it to SAE J2735 SPaT messages. The SPaT messages generated by V2X Hub can be forwarded to the RSU, which will broadcast any received SAE J2735 messages. These three components are typically within one local network and these communications on the transport layer are done using the User Datagram Protocol (UDP).

CAVe-Lite simulates this system within a computer using pre-recorded data, V2X Hub, UDP forwarding scripts, decoder scripts, and any accessible Wi-Fi network.

The components for the CAVe-Lite infrastructure kit are listed below:

- TSC script.
- Message Intersect script.
- V2X Hub software.

The functional design of the infrastructure kit is shown in Figure 1.




Source: FHWA.

**Figure 1. Diagram. Functional Design of CAVe-Lite Infrastructure Kit.**

## TSC SCRIPT

In order to bypass large, physical equipment, a full, pre-recorded SPaT cycle from a TSC is used. The content of this packet capture (PCAP) file is continually forwarded by the TSC script to V2X Hub. At this point, V2X Hub takes those NTCIP 1202 messages and converts them into SPaT messages. Refer to Figure 1.

The included recording was captured from a four-way intersection and contains the following timing:

- Green state = 15 seconds.
- Yellow state = 3 seconds.
- Red state = Green + Yellow + 2 second all phases red = 22 seconds.

## MESSAGE INTERSECT SCRIPT

The message intersect script produces the most functionality for CAVe-Lite. It serves to receive, decode, and broadcast any valid J2735 messages it receives.

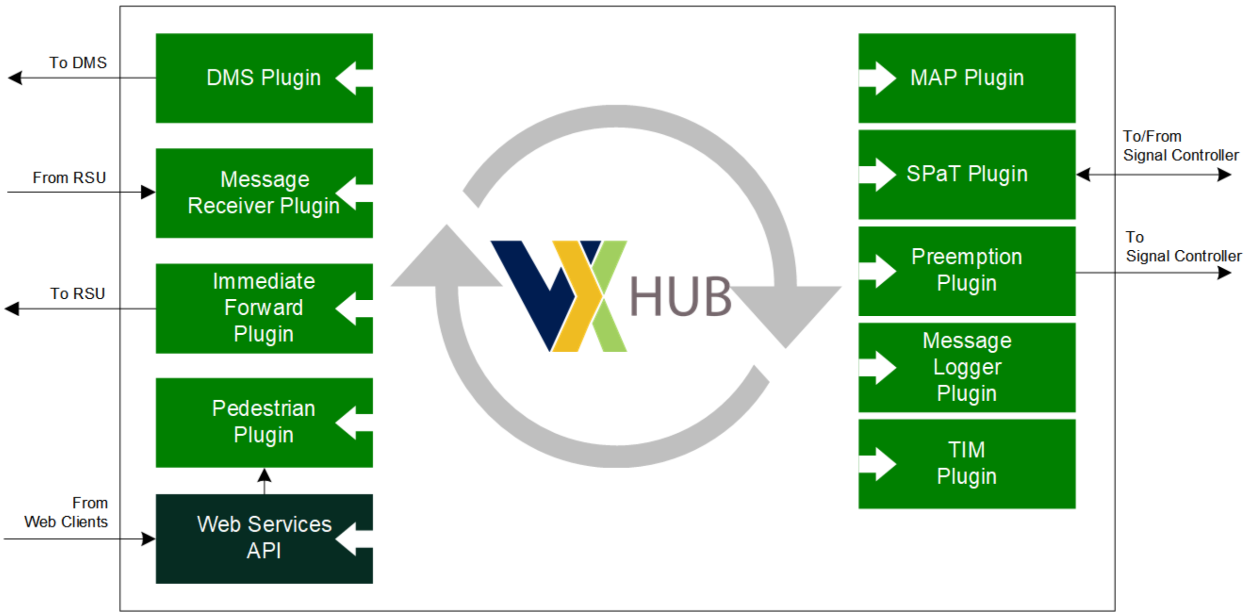
In order to visualize messages, the script includes a decoding section at its core. The decoder enables the optional use of an external digital signal head or LEDs to physically see the current state of a single phase. These decoded messages must be valid J2735 messages, which is the same standard an RSU receives.

SPaT messages are a message type in the SAE J2735 standard and can either be logged or forwarded to an RSU. The RSU will immediately broadcast any J2735 message received as a UDP packet over its respective 5.9 GHz band. An RSU must also adhere to the RSU 4.1 specification, where one of the requirements is having an immediate forward port. The message intersect script replaces this functionality. It mimics the RSU by opening and listening for incoming messages at a UDP port. Once valid messages are received, they are broadcast to the Wi-Fi network it is connected to.

**V2X HUB**

V2X Hub is a middleware that provides an application stack with the purpose of being used as a translator for SAE J2735 and NTCIP 1202 standard messages. Any messages exchanged between the infrastructure and vehicles a required to be a message type defined in the SAE J2735 standard. V2X Hub converts messages from the various infrastructure-based components to SAE J2735 messages for broadcast. V2X Hub also has a logging feature that can log certain messages in the JavaScript Object Notation (JSON) format.

V2X Hub is used by CAVe-Lite to translate NTCIP 1202 messages to SAE J2735, while also receiving BSMs. These functions are carried out by the SPAT and Message Receiver plugins respectively. A plugin for V2X Hub is an independent sub-component application that uses V2X Hub for its messaging and coding service while supporting an ITS application, such as SPaT messages, traveler information messages (TIMs), MAP messages, personal safety messages (PSMs), and traffic performance analysis, to name a few. Figure 2 shows the set of plugins developed for V2X Hub that are available for deployment.




Source: FHWA

**Figure 2. Diagram. V2X Hub operational design diagram.**

The following are brief descriptions of the used V2X Hub plugins:

- **Message Receiver Plugin.** This is the receiver plugin that handles incoming J2735 BSMs. The plugin decodes/parses the BSMs and internally broadcasts them using message handlers for other subscriber plugins.
- **Immediate Forward Plugin.** This plugin is used to externally broadcast J2735 messages through an RSU. The plugin interfaces with the RSU 4.1 specification forwarding feature, which is also used by the CAVe-Lite Message Intersect script.
- **SPaT Plugin.** This plugin connects with a TSC, reads NTCIP 1202 messages, converts them into SAE J2735 messages, and internally broadcasts them using message handlers for other subscriber plugins.

## CAVE-LITE MOBILE KIT

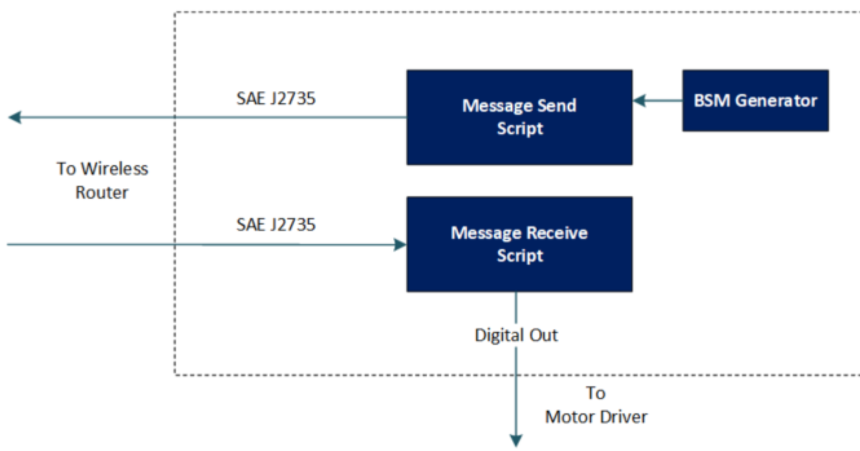
The scripts that form the mobile kit mimic the functionality of a DSRC or C-V2X onboard unit (OBU). OBUs generate BSMs based on GPS data and transmit them on the respective 5.9 GHz ITS Band. Message contents include vehicle position, acceleration, speed, heading, and other fields. They may also be used as simple radios, as they are in the case of OBUs used with the CARMA Platform℠.

In the case of CAVe-Lite, the mobile kit may be used with a CARMA 1Tenth vehicle or other 1/10 model vehicle to generate BSMs and broadcast them, using the Message Send script, to the network it is connected to. At the same time, the Message Receive script allows for messages broadcasted on the network to be received and manipulated to control the 1Tenth vehicle.

The components for the CAVe-Lite mobile kit are listed below:

- Message Receive script.
- Message Send script.
- BSM Generator.

The functional design of the mobile kit is shown in figure 3.




Source: FHWA.

**Figure 3. Diagram. Functional Design of CAVe-Lite Mobile Kit.**

## MESSAGE RECEIVE SCRIPT

The Message Receive script is used to replace the receiving functionality of an OBU. The script listens to the 255.255.255.255 broadcast and can be configured to receive any SAE J2735 message type. The script includes an additional decoding capability. It is set to decode SPaT messages and use that data to control a motor via a motor driver as it approaches a user-specified distance. The vehicle will either:

- Start.
- Stop.
- Accelerate.
- Decelerate.

This demonstrates the concept of the Eco-Approach and Departure at Signalized Intersections application.

## MESSAGE SEND SCRIPT

OBUs have the capability to serve two transmitting functions. They may either transmit BSMs generated with GPS and other vehicular data, or they may transmit any user-specified SAE J2735 message type that is forwarded to it over the transport layer, using UDP. The message send script was designed to function as the latter.

The script does not generate its own BSMs but is rather able to broadcast a saved BSM message or broadcast any other J2735 message. This is the similar process the Message Intersect script follows for broadcasting.

## BSM GENERATOR

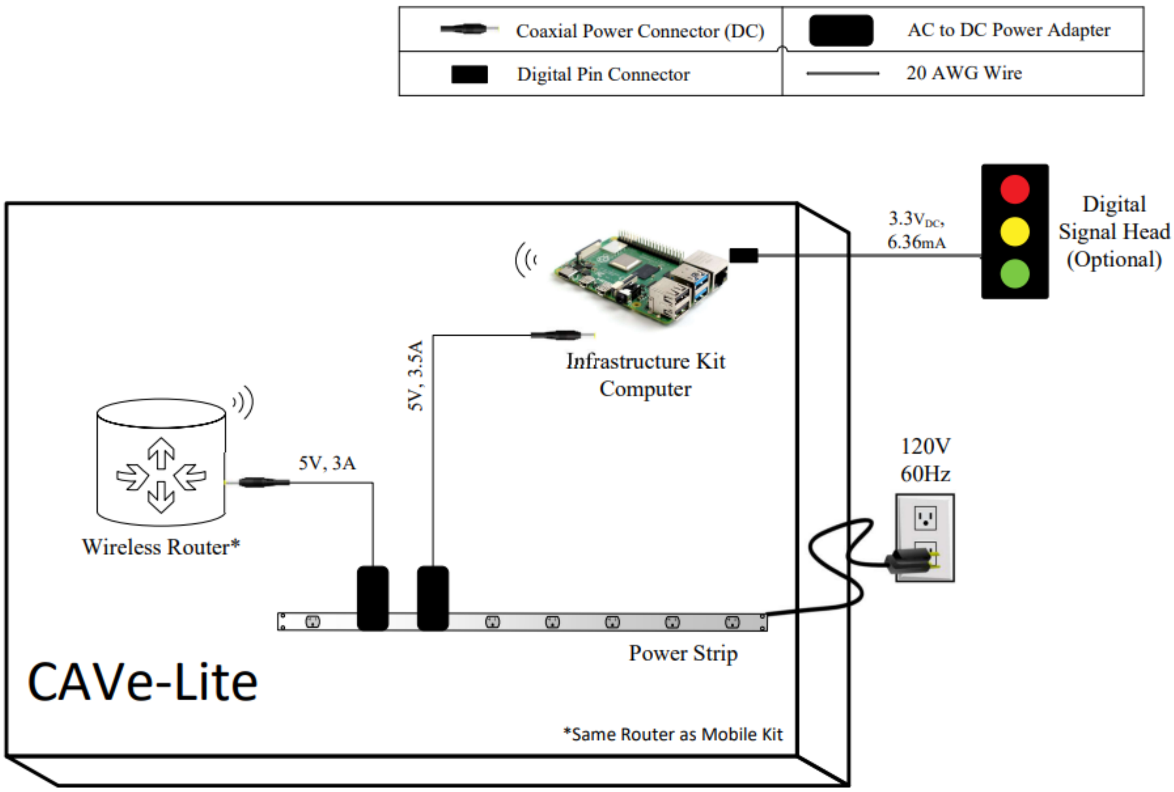
The BSM Generator script serves to populate the files used by the message send script. This script takes vehicle information and data from the message receive script to generate speed and acceleration data. The data is continually inserted to a BSM JSON file, where all other required fields are left as defaults. Simultaneously, the BSM generator script encodes the contents of the JSON file to create BSMs.

If a 1/10th vehicle is not used, a pre-recorded BSM is included and may be used. Alternatively, the BSM Generator script can be modified to include any other desired BSM fields.

## INTEGRATED SYSTEM DIAGRAMS

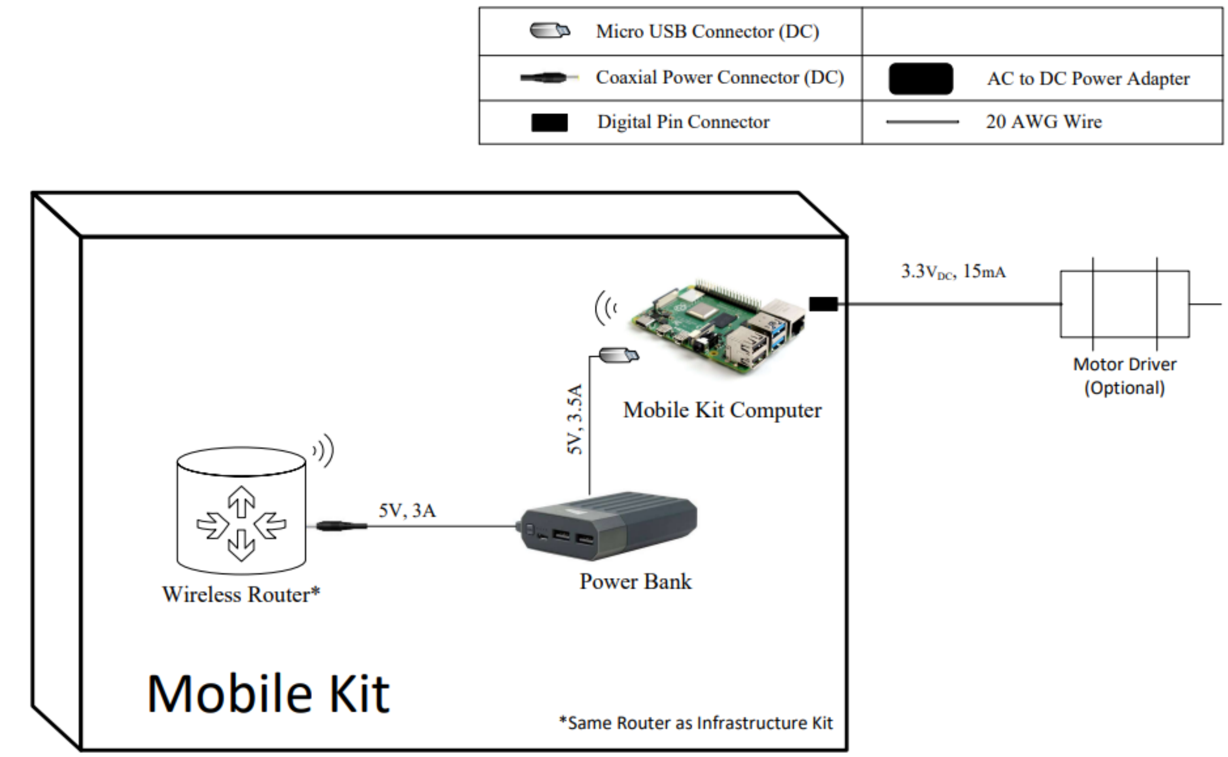
CAVe-Lite is made of various components, but only requires few physical connections. The infrastructure and mobile kits must be installed on two separate computers of either the same or different types and communicate to the same router using Wi-Fi. Optionally, the infrastructure kit may be connected to the router via ethernet. The interconnected diagrams for both components, found in figure 4 to figure 5, include connectors, physical interfaces, and power supplies used in both kits.

CAVe-Lite uses static IPv4 addresses that can be changed, allowing for use with a system-designated wireless router or any available wireless router. This setup is vendor-specific; certain connectors and interfaces may vary by vendor. It is recommended to reference the vendor-specific manuals.




Source: FHWA.

**Figure 4. Diagram. Interconnected System Diagram of CAVe-Lite Infrastructure Kit.**




Source: FHWA.

**Figure 5. Diagram. Interconnected System Diagram of CAVe-Lite Mobile Kit.**

## VENDOR LIST

Focusing on accessibility, CAVe-Lite can be used with almost any hardware vendor. The two major components needed are a computer and wireless router. Table 1 includes the used vendor and commonly available vendors. Options for the CAVe-Lite infrastructure and mobile kits are in abundance. Any ARM64 or AMD64 processor-based computer may be used, with a minimum 2 GB RAM and no required dedicated graphics processor. The same applies to the router. Any wireless router with minimum 802.11b standard is acceptable.

**Table 1. Vendor list for CAVe-Lite.**

| Component | Used in Document | Vendor |
|---|---|---|
| CAVe-Lite Computer (2) | Raspberry Pi 4 | Raspberry Pi, Simply NUC, Dell, HP, etc. |
| Wireless Router | GL-iNet SFT1200 | GL-iNet, Netgear, TP-Link, Linksys, etc. |

**Table 2. Optional component vendor list.**

| Component | Used in Document | Vendor |
|---|---|---|
| 3-Hole Indicator Switch Case | Schneider Electric XALD03h7 | Schneider Electric, IDEC Corp., Uxcell |
| *Red/Yellow/Green Indicator Panel LED, 12VDC, 22mm | VCC PCL2212V100B | VCC, RS Pro, Uxcell |
| Protoboard | SparkFun Snappable Protoboard | SparkFun, Adafruit |
| Motor Driver | SparkFun TB6612FNG | SparkFun, Pololu, Adafruit |
| 5V, 2A Voltage Regulator | Pololu S13V20F5 | Pololu, SparkFun, Adafruit |
| 20 AWG wire | Remington 20UL1007 | Remington, Adafruit, SparkFun |
| Resistor Kit, 1/4W | SparkFun Resistor Kit | SparkFun, Adafruit, Vishay |
| 2N2222A NPN Transistors | ON Semiconductor P2N2222A | ON Semi, ST Micro, TI |
| 1/10 Vehicle | ECX03028T1 | EXC, RaceCarJ Base Kit |
| 11.1V LiPo Battery | Traxxas 2872X | Traxxas, Spektrum, Zeee |

*** NOTE:** The external LEDs/Signal Head require custom electronics and soldering. A schematic can be provided from CAVSupportServices@dot.gov.

# CHAPTER 3. SOFTWARE CONFIGURATION

This chapter provides details on the software configurations required to set up different components in CAVe-Lite for operation. Due to the technical nature of this chapter, additional instructions for first-time Linux users are included. The IP addresses and ports provided in this instruction set are intended to be used as examples. Any changes to these values will require corresponding changes in any linked configurations.

## V2X HUB INSTALLATION AND INITIALIZATION

### V2X HUB

V2X Hub is available as a Docker image that may be run in a containerized environment (FHWA, 2023). The primary requirement for running V2X Hub is a local computer that can support Docker. Having root access to the device is preferred. For the development of this documentation, a Raspberry Pi 4 board using Raspberry Pi OS (64-bit), a variant of Debian 11 (Debian, 2022), was used.

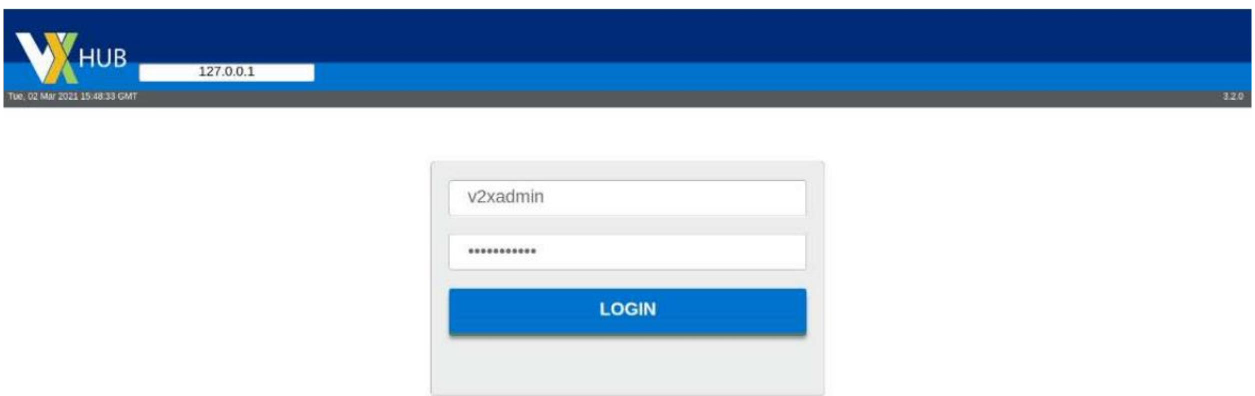
V2X Hub is installed on the same computer running the infrastructure kit scripts. It is recommended to configure the computer to use ***192.168.0.146*** as the static IPv4 address to be able to mimic the configurations stated within this document.

Further documentation for AMD or ARM 64-bit architecture processors are available through the V2X Hub GitHub repository and Confluence page. Installation can be completed using the following steps, although they may be subject to change:

1. Open a terminal to start download and installation.
2. Download the repository to your Linux computer's home directory:
    a. `cd`
    b. `sudo git clone https://github.com/usdot-fhwa-OPS/V2X-Hub.git`
3. Once downloaded, navigate to a directory corresponding to your computer's processor (a) or (b):
    a. `cd ~/V2X-Hub/configuration/arm64/`
    b. `cd ~/V2X-Hub/configuration/amd64/`
4. Currently in `~/V2X-Hub/configuration/arm64/`. Run the initialization script:
    a. `sudo ./initialization.sh`
        i. Note: There will be a prompt to create two mysql passwords. These serve for access to the database used by V2X Hub.
    b. Installation complete. Terminal will show "Done." Next to the four containers.
5. A V2X Hub user must now be added:
    a. Follow the prompts to add a user/pass. Example used in this documentation:
        i. User: `v2xadmin`
        ii. Pass: `V2xHub#321`
        iii. MySQL Root Password: *`Initial password entered in prompt`*

Installation completed. V2X Hub is now accessible at the computer's IPv4 address from anywhere in the network. Using either the Raspberry Pi or any other computer on the same network:

1. Open an internet browser and go to:
    a. http://192.168.0.146
    b. The screen will be in a continual loading phase. Open a new tab and go to:
        i. https://192.168.0.146:19760
        ii. Accept the credentials on the page. They may be under an advanced option
    c. Return to the first tab. You will have to change the Internet Protocol (IP) address in the text box to:
        i. 192.168.0.146
        ii. Default is 127.0.0.1 if logging in from the computer running V2X Hub.
        iii. Refer to figure 6 for the text box.




Source: FHWA.

**Figure 6. Diagram. V2X Hub Login Page.**

This document will address basic configuration for the V2X Hub applications. For further reading, please visit the V2X Hub GitHub repository and Confluence page.

**PLUGIN CONFIGURATION**

For successful CAVe-Lite implementation, please configure the plugins with the following values.

**IMMEDIATE FORWARD PLUGIN**

The infrastructure kit will be configured to the IP address shown in table 2. The port number provided in this document is from the RSU 4.1 standard, used by all RSUs and CAVe-Lite. The Provider Service Identifier (PSID) configuration is necessary to enable complete utilization of

CAVe-Lite features. Default configurations must be changed to function with CAVe-Lite. Table 2 shows the values used by the Immediate Message Forward plugin for CAVe-Lite.

**Table 2. Sample values for the Immediate Forward plugin.**

| Key | Default Value | Description |
|---|---|---|
| Messages_Destination_1 | { "Messages": [ { "TmxType": "SPAT-P", "SendType": "SPAT", "PSID": "0x8002", "Channel": "183" }, { "TmxType": "MAP-P", "SendType": "MAP", "PSID": "0x8002", "Channel": "183" }, { "TmxType": "PSM", "SendType": "PSM", "PSID": "0x8002", "Channel": "183" } ,{ "TmxType": "TMSG07", "SendType": "TMSG07", "PSID": "0x8002", "Channel": "183" },{ "TmxType": "TMSG03-P", "SendType": "TMSG03-P", "PSID": "0xBFEE", "Channel": "183" },{ "TmxType": "TMSG05-P", "SendType": "TMSG05-P", "PSID": "0x8003", "Channel": "183" }] } | JSON data defining the message types and PSIDs for messages forwarded to the V2X radio at destination 1. |
| Messages_Destination_2 | { "Messages": [ ] } | JSON data defining the message types and PSIDs for messages forwarded to the V2X radio at destination 2. |
| Messages_Destination_3 | { "Messages": [ ] } | JSON data defining the message types and PSIDs for messages forwarded to the V2X radio at destination 3. |
| Messages_Destination_4 | { "Messages": [ ] } | JSON data defining the message types and PSIDs for messages forwarded to the V2X radio at destination 4. |
| Destination_1 | 192.168.0.146:1516 | The destination UDP server(s) and port number(s) on the V2X radio for all messages specified by Messages_Destination_1. |
| Destination_2 | 0 | The destination UDP server(s) and port number(s) on the V2X radio for all messages specified by Messages_Destination_2. |
| Destination_3 | 0 | The destination UDP server(s) and port number(s) on the V2X radio for all messages specified by Messages_Destination_3. |
| Destination_4 | 0 | The destination UDP server(s) and port number(s) on the V2X radio for all messages specified by Messages_Destination_4. |
| Signature | False | True or False value indicating whether the RSU should sign the messages being transmitted. |
| signMessage | 0 | 0 if False and 1 if True indicating whether should sign outgoing J2735 messages. |
| HSMurl | http://<softHSM IP>:3000/v1/scms/ | REST API endpoint url needed to make HSM calls |
| MuteDsrcRadio | 0 | If true (1) then do not send packets to the radio for broadcast |

## SPAT PLUGIN

The SPaT plugin is used to convert NTCIP 1202 messages from the TSC script to SAE J2735 SPaT messages. The values in the Local IP and Local UDP Port fields must be changed to match the values used by the TSC script. Table 3 shows the values used for CAVe-Lite.

**Table 3. Sample values for the SPaT plugin.**

| Key | Default Value | Description |
|---|---|---|
| Intersection_Id | 1 | The intersection ID for SPAT generated by this plugin. |
| Intersection_Name | Intersection | The intersection name for SPAT generated by this plugin. |
| SignalGroupMapping | {"SignalGroups":[ {"SignalGroupId":1, "Phase":1, "Type":"vehicle"}, {"SignalGroupId":2, "Phase":2, "Type":"vehicle"}, {"SignalGroupId":3, "Phase":3, "Type":"vehicle"}, {"SignalGroupId":4, "Phase":4, "Type":"vehicle"}, {"SignalGroupId":5, "Phase":5, "Type":"vehicle"}, {"SignalGroupId":6, "Phase":6, "Type":"vehicle"}, {"SignalGroupId":7, "Phase":7, "Type":"vehicle"}, {"SignalGroupId":8, "Phase":8, "Type":"vehicle"}, {"SignalGroupId":22,"Phase":2,"Type":"pedestrian"}, {"SignalGroupId":24,"Phase":4,"Type":"pedestrian"}, {"SignalGroupId":26,"Phase":6,"Type":"pedestrian"}, {"SignalGroupId":28,"Phase":8, "Type":"pedestrian"}]} | JSON data defining a list of active SignalGroups and phases. |
| Local_IP | 192.168.0.146 | The IPv4 address of the local computer for receiving TSC Broadcast Messages. |
| Local_UDP_Port | 6053 | The local UDP port for reception of TSC Broadcast Messages |
| TSC_IP | 127.0.0.1 | The IPv4 address of the destination TSC. |
| TSC_Remote_SNMP_Port | 501 | The destination port on the TSC for Simple Network Management Protocol (SNMP) NTCIP 1202 v2 communication. |

## MESSAGE RECEIVER PLUGIN

The Message Receiver plugin only requires pointing to the IP address of the infrastructure kit computer in order to receive broadcasted messages. The port number needs to be configured to ***26789*** to ensure successful functionality, due to configurations in corresponding scripts. Table 4 shows the values used for CAVe-Lite.

**Table 4. Sample values for the Message Receiver plugin.**

| Key | Default Value | Description |
|---|---|---|
| IP | 255.255.255.255 | IP address for the incoming message network connection. |
| messageid | 0012,0013,0014,001D,00F0, 00F1,00F2,00F3,00F4,00F5, 00F6,00F7 | Comma separated list of SAE J2735 message ID values. |
| Port | 26789 | Port for the incoming message network connection. |
| RouteV2X | False | Set the flag to route a received J2735 message. |
| EnableSimulatedBSM | True | Accept and route incoming BSMs from a V2X Hub simulator. |
| EnableSimulatedSRM | True | Accept and route incoming SRMs from a V2X Hub simulator. |
| EnableSimulatedLocation | True | Accept and route incoming GPS location messages from a V2X Hub simulator. |
| EnableVerification | 0 | If enabled, security HSM features are enables and MessageReceiver Plugin will try to verify incoming message. |
| HSMLocation | <SoftHSM library location> | Location of HSM module static library |
| HSMurl | http://<SoftHSM IP>:3000/v1/scms/ | REST API endpoint url needed to make HSM calls |

## CAVE-LITE CONFIGURATION

The CAVe-Lite scripts making up the infrastructure and mobile kits require the same software in order to be used. As the kits are installed on two separate computers, their IPv4 addresses must be statically set in order to function without errors. The following sections provide guidance on installing software prerequisites, setting IP addresses, and configuring a wireless router.

### SOFTWARE PREREQUISITES

The following software and libraries are required in order to run CAVe-Lite:

- V2X Hub.
- Python 3.
- Pip 3.
- Pycrate.
- *Optional*: Gpiozero.

V2X Hub installation may be completed using the section above and is only installed on the infrastructure kit computer. To install the other prerequisites, run the following commands on both computers:

1. Open a new terminal and run the following:
    a. `sudo apt install python3`
    b. `sudo apt install python3-pip`
    c. `sudo apt install pycrate`
    d. *Optional*: `sudo apt install gpiozero`

### IP ADDRESS SETUP

The IPv4 address of any computer can be changed using a GUI or the command line. Methods for changing the IP address of a computer varies for every kind of OS and even between Linux distributions. IPv4 addresses for computers using CAVe-Lite must be set to static and not dynamic/DHCP. Instructions on changing a computer's IP address can be found from the specific distributor's website. The following steps are for the command line using Raspberry Pi OS (64-bit).

1. Open the file used to set network interface configurations:
    a. `sudo nano /etc/dhcpcd.conf`
2. The file may or may not include some of the following fields. Commented (#) lines may be left alone. Edit the file to include the following blocks:
    a.
```
interface eth0
static ip_address=192.168.0.146
static routers=192.168.0.1
static domain_name_servers=
static domain_search=
```

```
interface wlan0
static ip_address=192.168.0.146
static routers=192.168.0.1
static domain_name_servers=
static domain_search=
```

3. Save and exit the file:
    a. `<Ctrl-S>`
    b. `<Ctrl-X>`
4. Open the dns nameserver file:
    a. `sudo nano /etc/resolv.conf`
5. Edit the file to only contain the following line:
    a. `nameserver 8.8.8.8`
6. Save and exit:
    a. `<Ctrl-S>`
    b. `<Ctrl-X>`
7. Reboot: `sudo reboot`

## WI-FI ROUTER CONFIGURATION

The wireless router used by both components of CAVe-Lite must be configured with the IP address of *192.168.0.1*. The client list must also use the same subnet, including all IPv4 addresses within 192.168.0.*.

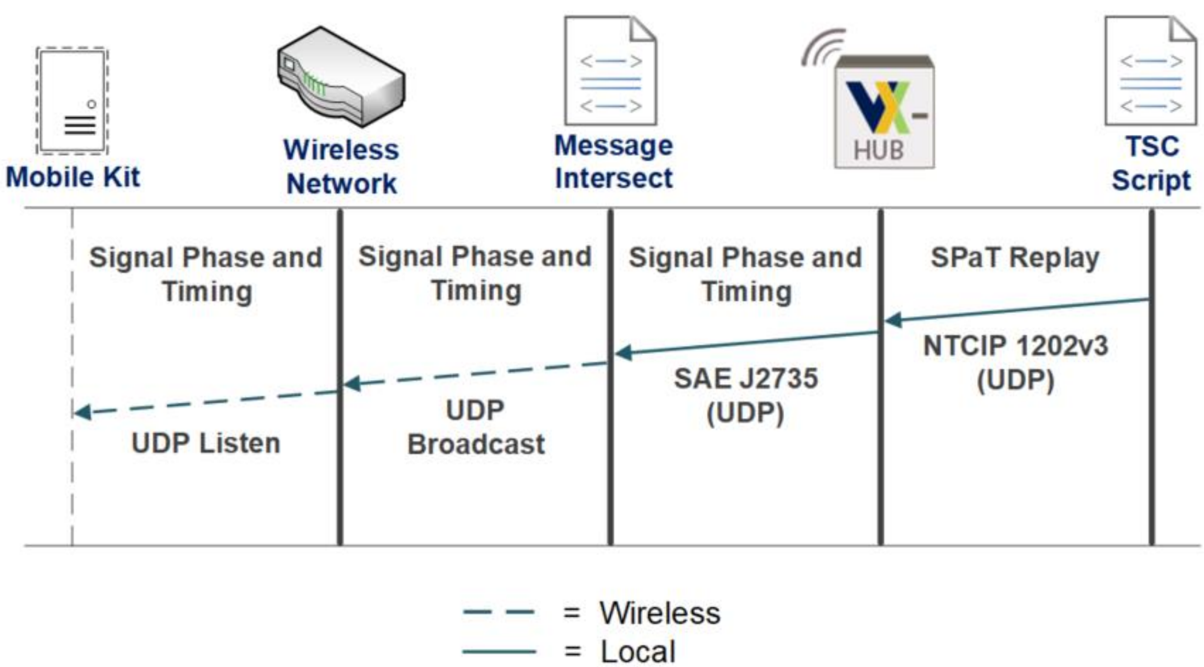
These settings are included by default by most off-the-shelf, consumer routers. To reset to these default settings, follow the vendor-specific instructions.

# CHAPTER 4. DATA FLOWS

There are two data flows that are discussed in this chapter. These data flows cover both directions from infrastructure-to-vehicle and vehicle-to-infrastructure, which are within the scope of most CAV applications. A portion of these data flows are executed in CAVe-Lite over Wi-Fi networks instead of V2X radios, but both methods of transmission share the same high-level process.

While there will exist multiple interacting components in a physical ITS, CAVe-Lite includes most of those processes internally using scripts. There are two message types that are used within this system. NTCIP 1202 and SAE J2735 messages can be either pre-recorded or generated manually, using the included JSON or XML examples. The scripts that make up the infrastructure and mobile kits take the messages and forward them to V2X Hub for conversion or broadcast them to the network to be received by the opposing kit.

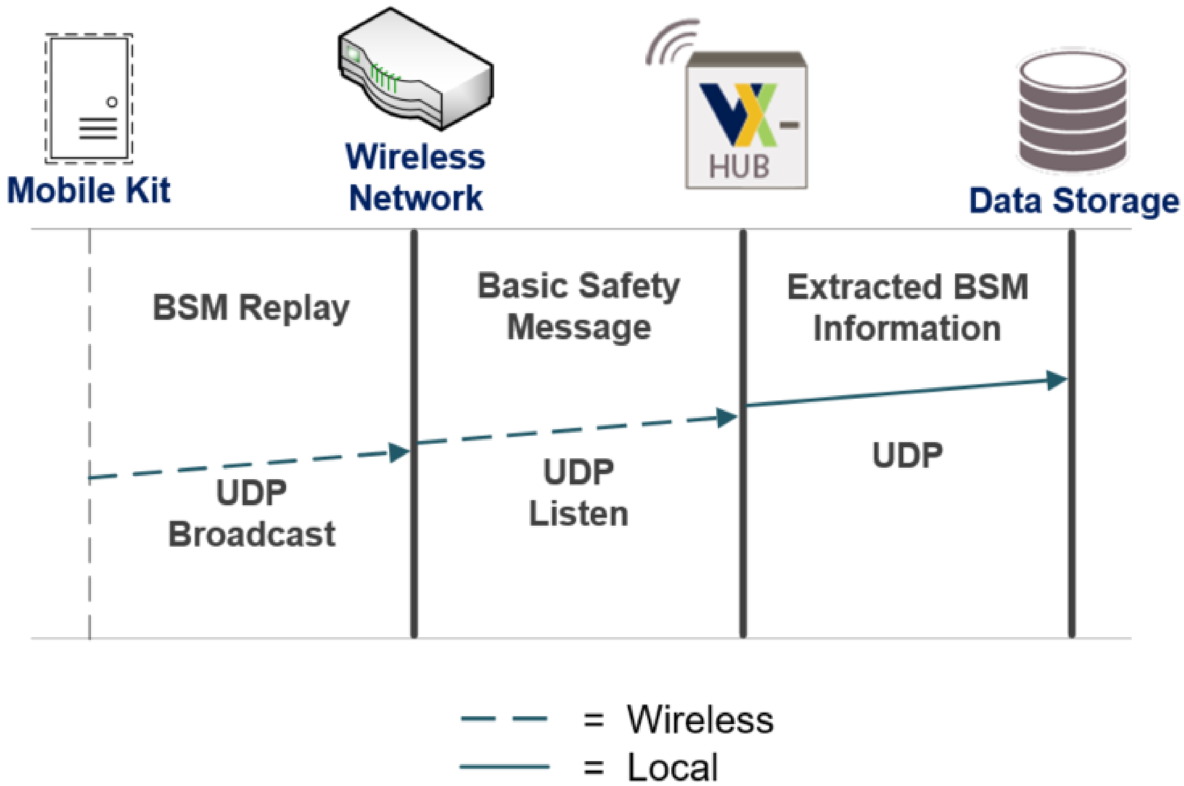
The data flow from infrastructure to mobile kits, shown in figure 7, demonstrates the SPaT application. The message originates as a pre-recorded PCAP file, containing the NTCIP 1202 message. The contents of the file are extracted by the TSC script and forwarded to the SPAT plugin in V2X Hub, using UDP. UDP is a Transport Layer protocol that is used for loss-tolerant applications with data flowing in a single direction. The plugin converts the packets to SAE J2735 SPaT messages, which are then UDP-forwarded to the Message Intersect Script. The script acts as an RSU and broadcasts the message to the wireless network, where the mobile kit is able to receive the SPaT message.




Source: FHWA.

**Figure 7. Diagram. SPaT Application Data Flow.**

Figure 8 shows the opposing BSM application data flow from vehicle-to-infrastructure. BSMs contained in the mobile kit are PCAP or Text files, containing SAE J2735 BSM messages. The BSM Send script extracts the payload from the contained file and continually broadcasts the message to the wireless network at ***255.255.255.255***, port ***26789***. The Message Receiver plugin in V2X Hub is configured to listen to the specific broadcast port. Any received BSMs can be decoded and logged by the BSMLogger plugin in V2X Hub.




Source: FHWA.

**Figure 8. Diagram. BSM Application Data Flow.**

# CHAPTER 5. USE CASES

This chapter covers the highlights of the included CAVe-Lite use case and a planned use case that would enable further applications and development with this tool. CAVe-Lite can be used to test various simulated CAV applications, excluding any V2X radio performance testing.

## CONNECTED VEHICLE (CV)

This use case enables the teaching of CV communication and control fundamentals. CAVe-Lite enables this application by providing all the communication features needed to simulate a connected intersection and a CV. With bidirectional communication happening with the infrastructure kit and mobile kit, each component is capable of decoding and manipulating the received data.

This feature is enabled by default in both kits at startup. On the infrastructure side, the software is continually broadcasting SPaT messages and decoding those transmitted messages. This feature is only decoding a single phase and printing data for visualization in a terminal, including the phase number, phase state, and time to next state. The decoding script also provides a digital output for the traffic light. Having only one decoded phase provides an opportunity for users to add additional decoding capabilities to the Message Intersect script.

At the mobile kit, received SPaT messages are also decoded and data is used to provide digital outputs that may be used to control the speed of an electric motor. To control the speed of the motor, an arbitrary distance to the intersection is manually input by the user, when prompted. The speed of the motor is set by the amount of rotations completed, phase state, and phase timing. The same phase is decoded in both kits, allowing for a visual comparison, validation of communication, and validation of speed control.

## SIGNAL PHASE AND TIMING (SPAT) GENERATOR

In its current state, CAVe-Lite only includes one use case. A second use case is planned for including capabilities similar to a traffic signal controller (TSC). The infrastructure kit currently uses pre-recorded NTCIP 1202 messages from a TSC. The messages contain information for a 4-way intersection's signal phase and timing. While it is useful for some CV use cases, it can be limiting.

The proposed SPaT generator will allow users to edit the number, state, and timing of phases used by CAVe-Lite. Having this feature will provide users with the opportunity to create their own SPaT messages, removing any reduction in use cases. Additionally, creating SPaT messages will provide an understanding of message encoding, navigating the SAE J2735 standard, and general understanding of signal phase and timing requirements.

# CHAPTER 6. CONCLUSION

This document has presented CAVe-Lite and the details associated with the included CV application. CAVe-Lite serves an educational goal, with a focus on networking in an ITS and the use of data from SAE J2735 messages for CAV applications. For this reason, the presented materials may be changed or expanded during training to provide a hands-on experience with different network configurations and understanding the SAE J2735 standard.

The installation chapter further provides detailed instructions on installing and running both CAVe-Lite components. This section provides a hands-on experience in using Linux for first-time users and covers many commonly used commands in this OS, which is used in almost every ITS equipment. By including instructions and explanations of commands, users can learn and obtain a good understanding of how other equipment is used.

CAVe-Lite was designed with a focus on greater portability and usability than other existing ITS equipment, including CAVe-in-a-Box. The system is not resource-heavy and can be installed on most modern computers, including single-board computers. This further reduces the level of entry, allowing for quick deployment and use by anyone with varying knowledge on intelligent transportation systems. Resources for the networking protocols used can be researched by users to better understand the system. Understanding the IP Protocol Stack, specifically the Transport Layer, is the recommended place to start: https://gaia.cs.umass.edu/kurose_ross/ppt.php

CAVe-Lite may be installed by anyone for immediate use. Additionally, a limited number of CAVe-Lite kits are available for lease through an equipment loan program hosted at STOL in Turner-Fairbank Highway Research Center (TFHRC). For more information on the program, please email CAVSupportServices@dot.gov.

For feedback/comments, please email CAVSupportServices@dot.gov.

## REFERENCES


USDOT ITS JPO. 2020. ITS Professional Capacity Building Program. Accessed April 22, 2022. https://www.pcb.its.dot.gov/CV_deployer_resources.aspx

USDOT ITS Standards Program. 2009. NTCIP 1202 - Object Definitions for Actuated Traffic Signal Controller Units. Accessed April 22, 2022. https://www.standards.its.dot.gov/Factsheets/Factsheet/22

SAE. 2020. V2X Communications Message Set Dictionary™ Set J2735SET_202007. Accessed February 22, 2022. https://www.sae.org/standards/content/j2735set_202007/

usdot-fhwa-OPS. 2022. V2X Hub. Accessed April 19, 2022. https://github.com/usdot-fhwa-OPS/V2X-Hub

usdot-fhwa-stol. 2022. CARMA Platform. Accessed April 18, 2022. https://github.com/usdot-fhwa-stol/carma-platform

usdot-fhwa-stol. 2022. CARMA 1Tenth. Accessed February 22, 2022. https://github.com/usdot-fhwa-stol/carma-1-tenth

USDOT, Eco-Approach and Departure at Signalized Intersections. N.D. Available at: https://www.fhwa.dot.gov/publications/research/operations/15011/15011.pdf

J.F. Kurose, K.W. Ross. 2020. Computer Networking: A Top-Down Approach 8th edition PowerPoint Slides. Accessed April 22, 2022. https://gaia.cs.umass.edu/kurose_ross/ppt.php