Lab 3: BSM Replay Attack

Introduction

In Labs 1 and 2, you learned a little bit about the information contained in *basic safety messages* (BSMs) and observed how they are exchanged between vehicles to increase awareness and help avoid collisions. One important feature of BSMs is that they are *broadcasts*, meaning they are transmitted to every receiver within communication range (~1 km) of the transmitting vehicle. BSMs are also unencrypted, as confidentiality is not a major concern (the information in BSMs, like vehicle speed, is not secret) and decryption would require significant additional processing time at the receiver.

Unfortunately, since BSMs are both *broadcast and unencrypted*, they are a prime target for *replay* attacks. In a BSM replay attack, a malicious attacker listens for BSMs and stores any BSMs that are received over an indefinite time period. The attacker then, at a later time, re-transmits the BSMs that were received as if they are being sent by the original sender. Depending on the information contained in those BSMs, this could cause serious problems; for example, receiving vehicles will think there is a vehicle present at a location where the original sender was once located (but is not now), causing those receiving vehicles to react to avoid a collision with a non-existent vehicle. In this lab, you will use the V2Verifier testbed to execute a BSM replay attack, exploring both the potential consequences of such an attack as well as the security mechanisms in the IEEE 1609.2 standard that help mitigate these dangerous attacks in real V2V systems.

Required Equipment

Unlike previous labs, in this lab you should use DSRC rather than C-V2X. This is because C-V2X has some technical complexity (at the physical layer) which makes executing a replay attack more difficult, less reliable, and *significantly* more time-consuming (to configure) than for DSRC. Thus, the required equipment for this lab is as follows:

- 3x USRP B210 SDRs
- 3x antennas capable of operation on 5.9 GHz
- 3x PCs (or virtual machines) with USB 3.0 ports

Operating System

This lab is designed to be used with Ubuntu 18.04. These instructions may not work properly with older or newer versions. An Ubuntu 18.04 ISO is available at http://releases.ubuntu.com/18.04/. Instructions for installing Ubuntu on a PC are available here. Although a PC is recommended, if using VMs you may refer to Appendix 1 for Ubuntu installation instructions.

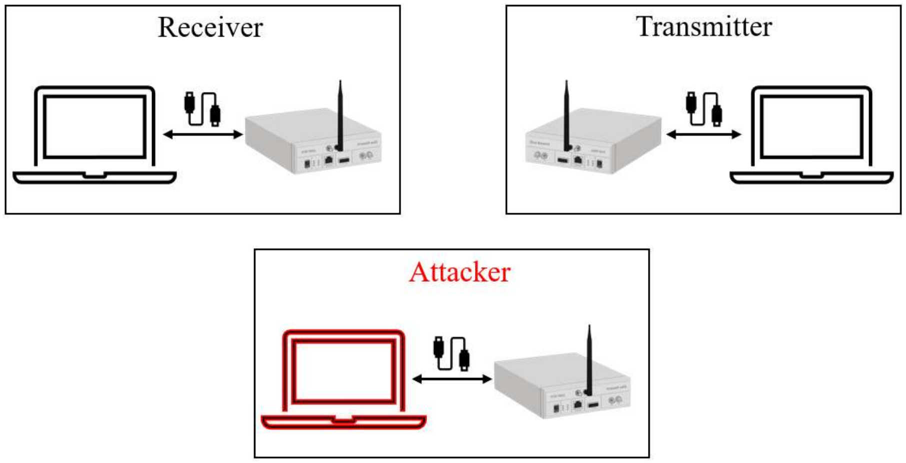
Activity 1 – Install the Operating System

Make sure you complete this activity on three PCs**.** If you are using VMs, you may choose to create one VM and then clone it for use on separate PCs (to save time).

1. Install Ubuntu 18.04 LTS. You can get an ISO from https://releases.ubuntu.com/18.04/.
    - If you are using a PC, follow the Ubuntu installation instructions.
    - If you are using a VM:
        i. Create a new **Ubuntu 18.04 LTS** virtual machine using VMware Workstation[1]. Do not attempt to download or use an existing Ubuntu VM, you **must** create a new VM for these instructions to work properly.
        ii. Give the VM a minimum of 4GB memory and 20GB storage and leave the remaining installation settings as their default values. Once installed, wait for the OS to automatically update, then reboot.
2. Run `sudo apt update` and `sudo apt -y upgrade` to make sure all installed software packages are up to date. Do not upgrade to a newer Ubuntu version if you are prompted to do so. When the update process completes, restart the PCs (or VMs) and proceed to the next activity.

Activity 2 – Installing the V2Verifier Testbed

This lab assumes that you have completed Lab 1 and either already have, or are already familiar with, the necessary steps to set up and configure the V2Verifier testbed for your chosen technology. If you have not completed Lab 1 (or just need a refresher), refer to Activity 2 in Lab 1 for detailed instructions. **Make sure to set up three (3) PCs** for this lab. You will need one PC each to transmit and receive BSMs, as well as a third PC to serve as the attacker. PCs will be referred to according to the diagram below as receiver, transmitter, and attacker in the remainder of this lab.



[1] If VMware Workstation is unavailable, we recommend using VirtualBox as a (free) alternative.

Activity 3 – BSM Replay Attack

First, you will execute a BSM replay attack with V2V security disabled. This will demonstrate the dangers of the attack and help contextualize our need for security in V2V.

1. On the receiver and transmitter PCs, open a terminal and run `gnuradio-companion` to launch GNURadio.
2. On the receiver PC:
   - Open and run the Wi-Fi receiver flowgraph (`wifi_rx.grc`) that is in the `grc` directory of V2Verifier.
   - Open a new terminal and `cd` into the cloned V2Verifier directory.
   - Launch the V2Verifier receiver program with the option (`-g`) to launch its web-based graphical interface.

     ```
     python3 v2verifier.py receiver -t dsrc -g tk -s false
     ```
3. On the transmitter PC:
   - Open and run the Wi-Fi transmitter flowgraph (`wifi_tx.grc`) that is in the `grc` directory of V2Verifier.
   - Open a new terminal and `cd` into the cloned V2Verifier directory.
   - Launch the V2Verifier transmitter program.

     ```
     python3 v2verifier.py transmitter -t dsrc
     ```
   - On the USRP, you should see a red blinking light on the port with the antenna attached (e.g., TX/RX) indicating that messages are being transmitted. If not, double check that your USRP is properly connected to the PC and that you have carefully followed all instructions above.
4. Returning to the receiver PC, you should see some activity on the V2Verifier GUI, confirming that BSMs are being sent and received. Make sure this is working properly before proceeding.
5. On the attacker PC:
   - Open and run the `wifi_rx.grc` flowgraph
   - In a new terminal, `cd` to `~/v2verifier/replay_attack`
   - Run the provided script with `python3 ./replay_attack.py 10`
   - The script will run for 10 seconds and collect BSMs transmitter during that time. When it completes, **do not press Enter** yet (even though it tells you to!)
   - Return to GNURadio Companion. Stop the `wifi_rx.grc` flowgraph. Open and run the `wifi_tx.grc` flowgraph.

- Return to the terminal window and press Enter to continue the replay attack

6. On the receiver PC, take a look at the V2Verifier GUI. What do you see that looks different now that the attack is running? Keep in mind that what you are looking at is the perception the receiving vehicle would have of its surrounding environment. Can you tell which messages are replayed and which are not?
7. Based on your observations in the previous step, what do you think are some potential consequences of a BSM replay attack?

Activity 4 – Mitigating BSM Replay Attacks with 1609.2 Security

For this activity, you will repeat Activity 3; however, this time you will enable 1609.2-based V2V security. Repeat steps 1-5 of Activity 3 as they are provided *with the exception* of Step 2 – for this step, use the following command instead of the one from Activity 3:

```
python3 v2verifier.py receiver -t dsrc -g tk -s true
```

Once you complete Step 5 from Activity 3, answer the following questions:

1. On the receiver PC, take a look at the V2Verifier GUI. Can you see any effects of the replay attack? What is different now than it was previously?
2. Based on your observations, do you think IEEE 1609.2 provides sufficient mitigation for BSM replay attacks?
3. On the receiving PC, stop the `wifi_rx.grc` flowgraph and open up the packet capture file in `/tmp/out.pcap`. Take a look in the packet contents (specifically, at the 1609.2 security fields). What field(s) do you see that are applicable to mitigating replay attacks? Consider your understanding of how a replay attack works and any lecture discussions you may have had when answering this question.