Lab - Bluetooth

Dino Lopez Pacheco
dino.lopez@univ-cotedazur.fr

1 Introduction

BlueZ provides the protocol stack implementation of the Bluetooth technology (from the Bluetooth 2.0 standard up to the Bluetooth 5.0 standard) for Linux-based systems. BlueZ is currently a stable technology. However, the latest features introduced in Bluetooth 5.0 (e.g. the mesh topology and the angle of arrival features) are in a experimental phase.

BlueZ provides both, a set of tools to manage, test and debug Bluetooth applications, and a rich API allowing the easy implementation of Bluetooth applications.

2 Objectives of this lab

In this lab, you will test the different phases of the Bluetooth secure link establishment to consolidate what you have learned during this lecture. Hence, you will be able to test one of the simplest Bluetooth attacks and understand how such an attack works.

To do the exercises proposed in this lab, you will need a Linux-based system with BlueZ, 2 Bluetooth USB dongles to be provided by the instructor of this lab, and another Bluetooth device that you might have with you (e.g. the integrated Bluetooth device in your laptop).

You can do the exercises using any Linux system, properly configured with the BlueZ stack. To make things easy, you can just use the VirtualBox Linux VM that we have made available for you (cf. slides from Pr. Yves ROUDIER).

Please, do screenshots to prove that you're successfully completing every suggested exercise.

3 A first glimpse on the Bluetooth device management

1. First, let's check that our Linux box successfully identifies our Bluetooth devices.
    a. Connect one of your Bluetooth USB dongles. Then execute the command "$ lsusb". If you get a line like "Bus 002 Device 003: ID 0a12:0001 Cambridge Silicon Radio, Ltd Bluetooth Dongle (HCI mode)", your Bluetooth device is probable working well. You can go to the next subitem.
    b. Become the root user and execute "# hciconfig" and give the Bluetooth address of your device and say if your device is enabled or not from the Linux kernel point of view (look at the 3rd line of the output).
    c. Execute the command "# hciconfig hci0 noscan".
2. In another machine (or VM), repeat the same operations in Q1 for your second Bluetooth USB dongle.
3. With the hciconfig command, you can enable/disable a Bluetooth interface, just like you can do with an Ethernet interface. Assuming your Bluetooth device is identified with the name "hci0" (see the output from point 1b), you can disable the interface with the command "hciconfig hci0 down".
    a. Disable, then enable again your Bluetooth device. Verify with hciconfig that your commands worked as expected.
4. To build piconets, Bluetooth devices need to find surrounding devices in the environment. The device looking for other ones is the master, and it needs to broadcast

inquiry messages in several channels. Devices listening for inquiries are the slaves and they need to enter an inquiry scan phase from time to time.

   a. With hciconfig, enable the inquiry scan state only in one of the Bluetooth devices
   b. At the laptop with the second device, execute the "# bluetoothctl" command. Then, once you're in bluetoothctl CLI, execute the command "scan on".
   c. Prove that you can find from bluetoothctl the device being in the inquiry scan mode.
   d. Give your commands

5. To exchange data through a Bluetooth link, a connection (either secured or not) must be created. SDP, the protocol to find the services offered by the Bluetooth devices, is one example of an application using an unsecured link.
   a. Test what happen if you try to explore the services offered by the device ($ sdptool browse *bt_mac_addr*) where you have previously enabled the inquiry scan mode.
   b. Explain why the SDP client (i.e. sdptool) works or not by looking at the slides if the lecture. If it doesn't work, find a solution.
   c. Like always, give your commands.
6. Using the Bluetooth USB dongle in inquiry and page scan, pair and connect to any of your Bluetooth device (except the second USB dongle that we have provided, because you will need it later for another exercise). To do so follow the instructions below:
   a. Execute the "# bluetoothctl" command to obtain the CLI (from the computer with your Bluettoth device in inquiry and page scan)
   b. Once at the CLI, scan for you device "scan on"
   c. After your own Bluetooth device show up, stop the scanning ("scan off") and pair to your Bluetooth device ("pair *bt:mc:addr*", where "*bt:mc:addr*" is the Bluetooth MAC address of your device)
   d. If the pairing process succeeds, create a long term encrypted link by connecting to it ("connect *bt:mc:addr*"). This will allow an automatic reconnection between the two devices when they are in the coverage area of each other.
   e. Verify that your devices are correctly linked.

4 Experiencing a DoS attack

While your Bluetooth devices are still connected, answer to the following question by reading the scientific article "Lounis, Karim and Mohammad Zulkernine. "Connection Dumping Vulnerability Affecting Bluetooth Availability." CRiSIS (2018)". Initially published in 2019, the Connection Dumping Vulnerability still affects several devices. This article is available for reading at https://web.archive.org/web/20210830005951/https://link.springer.com/chapter/10.1007/978-3-030-12143-3_16

7. by reading section "4 Connection Dumping Vulnerability"
   a. Explain what the "Connection Dumping Vulnerability" is.
   b. Describe the general scenario of such an attack.
   c. Say which assumptions must be satisfied for this attack to work.

8. Now that you understand how the CDV attack works, you're required to execute it on the Bluetooth connection you have set up on exercise # 6.
    a. Test the vulnerability following of the scenarios described in section "5 Attack Scenarios on Bluetooth Availability" of the article.
    b. While optimally you also need to spoof the class of device, this is not a hard requirement.
    c. To spoof the Bluetooth address of a device, you will need to use the "bdaddr" utility from Bluez.
    d. The "bdaddr" utility must be compiled, however. To do so,
        i. download and decompress the user space Bluez software (see http://www.bluez.org/download/)
        ii. install the following packages: build-essential libglib2.0-dev libdbus-1-dev libudev-dev libical-dev libreadline-dev python3-docutils
        iii. execute "sudo apt-get build-dep bluez"
        iv. go to the decompressed folder of the user space Bluez component (e.g. $HOME/bluez-5.65) and execute
        v. if everything went well, execute "make"
        vi. at the end, you must have the "bdaddr" utility under the "tools/" directory. Do not execute "make install".
    e. Proof with a video recording your attack