

# IoT Security

Android and IoT Bluetooth devices

PV285

Václav Oujezský, Karel Slavíček, Tomáš Pitner

# Bluetooth

- Classic Bluetooth - streaming and communication
- Bluetooth LE - low power requirements devices

The Bluetooth 4.0 specification includes both classic Bluetooth and Bluetooth LE. The term 'classic Bluetooth' isn't completely accurate. In the Bluetooth 4.0 specification, the SIG (Special Interest Group) defined four Bluetooth controller technologies, that means there is only one Bluetooth, which is the SIG's Bluetooth. The Bluetooth technology itself includes four types (modes): BR (Basic Rate), EDR (Enhanced Data Rate), AMP (Alternate MAC and PHY Extension), and LE (Low Energy). Since LE was proposed in 2010, people call the former BR/EDR/AMP technology 'classic Bluetooth' technology for convenience. **Don't make the mistake of thinking that Bluetooth only has the LE mode after Bluetooth 4.0!**

# Typical IoT applications

Smartphones can work with both specifications

Application scenario	Audio stream application	Data transmission application	Location services application	Device network application
	Wireless head-phones Wireless speaker Vehicle-mounted entertainment	Sports and fitness equipment Medical and health equipment Peripherals and accessories	Beacon services Indoor navigation service Asset tracking	Control system Monitoring system Automation system
Communication mode	One-to-one	One-to-one	One-to-many (broadcast)	Many-to-many (mesh)
Radio frequency mode	Classic Bluetooth BR/EDR	Bluetooth low energy (Bluetooth LE)		

# Bluetooth Stack

## Profiles:

- SPP - Serial Port Profile
- GAP - Generic Access Profile
- GATT - Generic Attribute Profile
- ATT - Attribute Profile
- ...

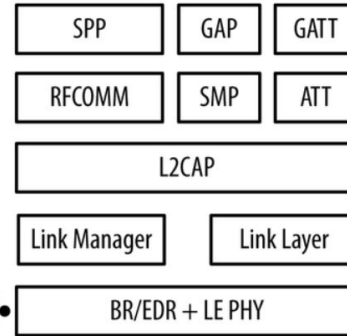
<https://www.bluetooth.com/specifications/specs/>



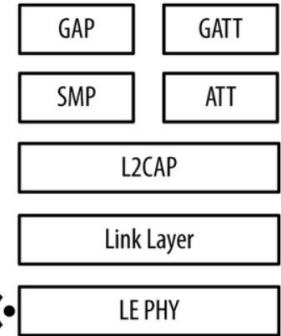
(classic or BR/EDR)



(dual mode or BR/EDR/LE)



(single mode or BLE)



# Bluetooth overview

**Bluetooth Profile** - to regulate the communication between heterogeneous Bluetooth devices. There are more than 30 profiles standardized by Bluetooth SIG aka Headset Profile (HSP), which specifies how a Bluetooth headset can be used with mobile phones.

**Bluetooth Connection** - one device in discoverable mode responding to an inquiry. If the initiator knows the MAC address (48 bit), the inquiry has to be answered. Pairing procedure follows after the information exchange. Pairing involves certain user interaction to confirm the identity of the remote device. If the pairing is successful, a shared secret named *link key* is created to encrypt their communication and the two devices are set to be bounded.

# Bluetooth characteristic

- No supporting infrastructure is required,
- Spontaneous (ad hoc) star network formation pico-net
- Pico-network is not large, many establishments would cause chaos
- A pico-network can consist of up to 8 devices
- Devices find each other and form a pico-net
- Once two BT devices have contacted each other, they can invite their user to start a communication session
- the user can accept or reject the request to start a session
- only Bt devices approved by the user participate in the session
- are (slave) secondary devices that approve the connection implicitly

# Bluetooth characteristic

The pico-network is controlled by one master (primary) station which is providing

- allocation of communication channels
- control of communication processes
- definition of radio space sharing (FHSS, TDD)
- synchronisation of the running time throughout the pico-network

The pico-network also consists of up to seven (active) slave stations

- Slave stations synchronize with the master station
- Slave stations cannot communicate directly with each other (except during the existence detection phase)

**Topics covered in course PA151**

# Android Bluetooth

The first Android versions used Linux's BlueZ stack.

- Since Android 4.2 - Bluedroid or Fluoride stack by Google
- To interact with the stack - Android Bluetooth API.
  - Classic Bluetooth API
  - Bluetooth LE API



# Android Bluetooth permissions

Access to the Bluetooth functionalities on Android is mediated by a set of permissions.

- **BLUETOOTH\_ADMIN** - A third-party app can initiate the discovery of nearby Bluetooth devices or change the Bluetooth settings
- **BLUETOOTH** - app can perform Bluetooth communication with another device
- **ACCESS\_COARSE\_LOCATION** - Bluetooth discovery may reveal the location of the user, from Android 6.0, if an app requests to scan nearby devices, it has to declare either the dangerous-level **ACCESS\_COARSE\_LOCATION** or **ACCESS\_FINE\_LOCATION** permission. By default, the pairing process needs the user's interaction.

Permissions are defined in **AndroidManifest.xml** file.

# Android Bluetooth permissions

```
bluetooth-test - AndroidManifest.xml [bluetooth-test.app.main]
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
bluetoothtest app > src > main > AndroidManifest.xml
Project
  Android
    app
      manifests
        AndroidManifest.xml
      java
        com.example.bluetooth_test
        com.example.bluetooth_test (androidTest)
        com.example.bluetooth_test (test)
      java (generated)
      res
        drawable
        layout
          activity_connection.xml
          activity_main.xml
        mipmap
        values
        xml
      res (generated)
      Gradle Scripts
        build.gradle (Project: bluetooth-test)
        build.gradle (Module: bluetooth-test.app)
        gradle-wrapper.properties (Gradle Version)
        proguard-rules.pro (ProGuard Rules for bluetooth-test)
        gradle.properties (Project Properties)
        settings.gradle (Project Settings)
        local.properties (SDK Location)
Resource Manager
Time

bluetooth-test - AndroidManifest.xml [bluetooth-test.app.main]
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:tools="http://schemas.android.com/tools"
4   package="com.example.bluetooth_test">
5   <!-- Request legacy Bluetooth permissions on older devices. -->
6   <uses-permission android:name="android.permission.BLUETOOTH" />
7   <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
8   <!--
9     Needed only if your app looks for Bluetooth devices.
10    If your app doesn't use Bluetooth scan results to derive physical
11    location information, you can strongly assert that your app
12    doesn't derive physical location.
13   -->
14   <uses-permission android:name="android.permission.BLUETOOTH_SCAN" />
15   <!--
16     Needed only if your app makes the device discoverable to Bluetooth
17     devices.
18   -->
19   <uses-permission android:name="android.permission.BLUETOOTH_ADVERTISE" />
20   <!--
21     Needed only if your app communicates with already-paired Bluetooth
22     devices.
23   -->
24   <uses-permission android:name="android.permission.BLUETOOTH_CONNECT" /> <!-- Needed only if your app uses Bluetooth scan results to derive physical location. -->
25   <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
26   <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" /> <!-- Bluetooth classic is required for your app -->
27   <uses-feature
28     android:name="android.hardware.bluetooth"
29     android:required="true" />
30
31   <application
32     android:icon="@mipmap/ic_launcher"
33     android:label="@string/app_name"
34     android:theme="@style/Theme.Blueoothtest">
```

# Weaknesses - examples

- **Inconsistent Authentication Process on Profiles** - (pairing principles) If the device makes changes on the profiles, it still gets trusted since pairing has already done.
- **Overly Openness to Profile Connection** - once the bond is created, the host will try its best to connect to all the profiles claimed by the remote device, without explaining the risk.
- **No Permission Management for Profile** - when a third-party app is granted BLUETOOTH\_ADMIN permission, for example Bluetooth keyboard, the keyboard becomes accessible automatically. Therefore, the app can further utilize the keyboard to inject inputs and take control of the connected device.

# Attacks - examples

**BLUETOOTH** and **BLUETOOTH\_ADMIN** are the standard and common permissions for typical Bluetooth apps. Google Play may grant them to a malicious app without user confirmation.

- **Changeable Profile** - instruct device to add an profile after pairing
- **Silent Pairing** - configuring remote to work without pairing and calling API `createBond()`
- **Connecting Sensitive Profile** - by using different framework instead of Android SDK

<https://sites.google.com/view/bluetoothvul/?pli=1>

# HC-05

## Most useful AT commands are

- AT : Check the connection
- AT+NAME : See default name
- AT+ADDR : See default address
- AT+VERSION : See version
- AT+UART : See the baudrate
- AT+ROLE: See role of bt module(1=master/0=slave)
- AT+RESET : Reset and exit AT mode
- AT+ORGL : Restore factory settings
- AT+PSWD: See default password

<https://www.laskakit.cz/bluetooth-modul-hc-05-ttl/>

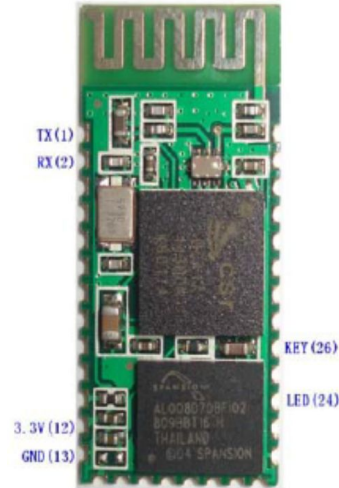


Figure 1 HC-06

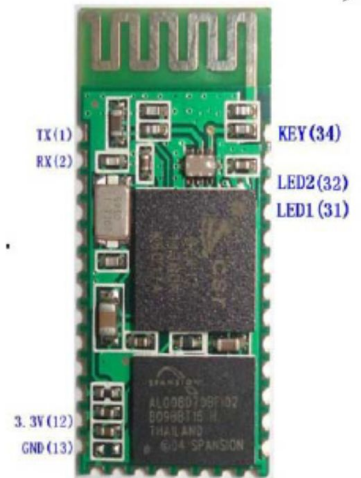


Figure 2 HC-05

# HC-05 specification

## Specifications:

Chips: 29LV800, BC417

Bluetooth version: V2.0+EDR

Basic baud rate: 9600,8,1,n.

Integrated Antenna.

Range max.10m

IO voltage: 3.3V

Input voltage: 3.3~6V

Including LED indication, uses 150mA and 3.3V

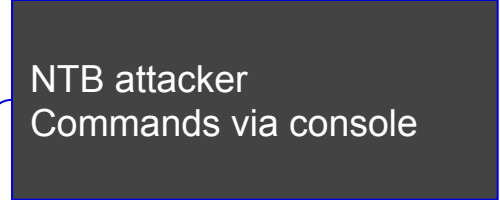
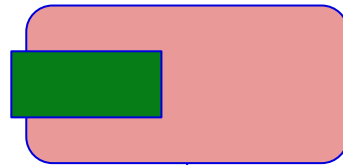
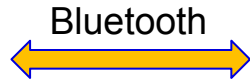
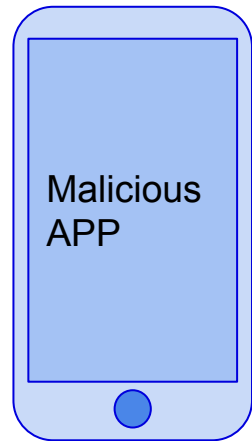
Compatible with bluetooth master modules or master-slave modules.

# LAB scenario

Android Studio, Arduino IDE

HC-05 module assembled on board (red)

STM 32 board



NTB attacker  
Commands via console

A dark gray rectangular box representing a terminal or console window. A blue curved line connects the bottom of the red board to the top-left corner of this box.

# LAB resources

Initial application Android code and Arduino IDE code:

[https://drive.google.com/drive/folders/1QKnLKP5FCpPZ-t\\_LG-sSeQO4QnQ1a\\_ql?usp=sharing](https://drive.google.com/drive/folders/1QKnLKP5FCpPZ-t_LG-sSeQO4QnQ1a_ql?usp=sharing)