Bluetooth: network possibilities

Steven Raaijmakers

February 2017

1 Introduction

Most smart-phones, computers and even televisions are using WiFi or 3G/4G to make a wireless connection to the Internet. By having this connection to the Internet, numerous possibilities are added to such devices. For example streaming from the Netflix-server directly to your television is one of the possibilities. In this case both your smartphone and the server need to be connected to the Internet. Even though there are plenty of advantages of using WiFi, most people tend to forget there are more ways to communicate wirelessly. One alternative is Bluetooth. When people think of Bluetooth they are most likely to think of a connection between two devices. However, Bluetooth can be used to set up small networks.

2 Bluetooth Network

A Bluetooth-network consist of different bluetooth-devices. In such a network a device can complete various tasks: being a master or slave [4]. The master-device is usually a Bluetooth-device that successfully initiated a connection to a *remote device*. This remote device will become a slave device once it accepts the incoming connection.

This master/slave relationship is handled via two different threads:

- *Connect-thread* : initiates a connection
- *Listen-thread* : accepts an incoming connection

The master tries to connect its connect-thread to the listen-thread of the slave. The master can connect to max. 7 devices, and therefore can assign up to 7 different slaves, while a slave can only have one master [4]. It is not possible for a device to be a master and a slave at the same time, or vice versa. Therefore this network can be seen as a so called *piconet* [2].

When a Bluetooth-device decides to joins a piconetwork it can become either a master or a slave device, depending on the user. Before joining the user gets the option whether it wants to join the network as either a master or a slave device. Joining as master-device therefore means creating a new network, instead of joining an existing one.

2.1 Autoconnect

The *UUID* is a unique identifier used in the Bluetooth-threads [3]. When a listen thread is set up with an UUID, it will only accept a connection with a connect-thread with a matching UUID. By setting up the listen-thread with a *insecureRfcomm*, the bluetooth-pairing doesn't require an approval of the app-user anymore. The UUIDs are the same for all the devices using the app.

A master device knows which devices it can connect to, by discovering all the Bluetooth-devices within its range. When a master-device is finished discovering, it will automatically try to connect to all those devices, while only devices with a matching UUID will accept the incoming connection. Thus creating a virtual autoconnect, which is essential for creating an ad-hoc network.

3 Forwarding

Since each slave is only connected to a master, it is not possible to communicate directly between slaves in the same piconet. Whenever a slave wants to communicate with another slave, the master has to forward the messages between these two slaves, creating a virtual connection between two slaves.

The forwarding is handled by the master comparing the receiver-field of the incoming message. Whenever they receiver-address does not match the master's own address, the master tries to forward the message. This can happen in two ways:

- The receiver-address of the message is the address of another slave, and thus the message is forwarded to this device.
- The receiver-address is not found among the other slave-addresses. The master will determine which of all the devices in the piconet is most likely to connect to the supposed receiver within the shortest time-frame.

4 BluetoothMessage

The messages that are exchanged between two devices are called *BluetoothMessages*. Each BluetoothMessage consist of 4 fields; sender, receiver, type and payload (containing the actual data to be send). Since the maximum frame size of a message in a thread is set to 128 bytes [1] at default, the payload mostly contains chunks of the actual data, which are merged together at the receiving side.

Different type of BluetoothMessages can be send, using the forwarding-process. Various examples of these types *type* include *TOPOLOGY*, *NETWORK_INFO* and *FILE*.

4.1 NETWORK_INFO

The *NETWORK_INFO*-type is used to distribute information about the current devices in the piconet between the slaves and master. This information contains the addresses and names of all the devices in the current network. Because of this information, slaves know which slaves they are virtually connected to. The distribution of this info is handled by the master:

- Whenever the master successfully initiates a connection to a slave, it will update its own *connectedAddresses*, containing all the addresses and names of its current slaves.
- After this update the master will flood its updated connectedAddresses to all its slaves (include the newly joined slave).
- When a slave receives a NETWORK_INFO message, it will updates its own connectedAddresses to the received connectedAddresses, now containing the names and addresses of the other slaves in the piconet.
- Whenever a slave disconnects from its master, the master will again update the connectedAddresses by removing the disconected slave. After the update this information is again flooded to its slaves.
- When a master leaves the network, all the connectedAddresses of the former network participants will be reset.

The same principle is used to spread the NetworkTopology (containing all the historic connections of a device) accross a piconet.

5 Research

One of the biggest disadvantages of using the described form of a piconetwork is that it is not scalable to a network with more then 7 devices. Therefore we can use the algorithm described earlier. Each device keeps track of the device it once had in its piconetwork, the time they where connected and how much. Based on this information a device determines which device in its current network is the leading contender on meeting the supposed receiver.

Hereby one device will act as a messenger between the sender and the receiver, thus creating one big virtual network which keeps on changing. The research is based on determining with the available information which devices will the best messengers, and why.

References

[1] Bluetooth.org. “RFCOMM WITH TS 07.10”. In: (2012).

[2] Jennifer Bray. “Masters and Slaves: Roles in a Bluetooth Piconet”. In: (2001).

[3] Peter Leow. *Android Connectivity*. https://www.codeproject.com/Articles/814814/Android-Connectivity. 2014.

[4] Paul; O'Connor Gerald; Connaughton Paul Tsang Will; Carey. "Bluetooth Terminology". In: (2001).