Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Master Projekt 2

Autor: Heiner Perrey

Performance Analysis of Bluetooth Low Energy with Merkle's Puzzle

Date: May 20, 2012

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Contents

1 Introduction

Note

Part of this work has been published at WiSec '11 and in ACM SIGMOBILE [8].

In the previous work "The Concept of Bluetooth Low Energy with Merkle's Puzzle" [7] the approach of integrating Merkle's Puzzle (MP) in the Bluetooth low energy (BLE) specification has been depicted. This work is designed to analyze the performance of this approach. BLE is a new technology introduced in the Bluetooth specification version 4.0 [2]. Its main goal is to connect energy restricted devices with one another. Since its specified bit rate is not sufficient enough for audio communication, most use cases include sensors like health or sports equipment [15]. For instance, a prototype watch by Casio with a BLE interface can run on a coin cell battery for about two years [14]. Also the iPhone 4S which was released in October 2011 is equipped with BLE [17].[1]

MP is a lightweight key agreement protocol proposed by Ralph Merkle in 1974 and published in 1978 [4, 18]. MP only offers security for a defined period of time. Therefore when using MP for key exchange the attacker will eventually gain access to the secret key. However with the security parameters of MP, the point in time can roughly be defined. As depicted in [8] this work assumes use cases in which one or many restricted functional devices (RFD) want to share a key with a fully functional device (FFD). Since the RFD cannot run complex key agreement protocols like Elliptic Curve Diffie-Hellman (ECDH), the FFD has to run the costly computations. Using MP the workload can be shifted to the FFD.[2]

This document is structured as follows. Chapter 2 gives an introduction to the BLE hard- and software which was used during the research of this work and evaluates its usability for future research. Chapter 3 depicts a performance analysis of the BLE with MP approach. Hereby different factors are taken into account. For one the optimal bit rate of BLE is compared

[1] More details about BLE can be found in [2, 8, 9, 19, 20].
[2] For a detailed description of the MP protocol see [1, 4, 5, 8].

with the expected bit rate. First benchmarks demonstrate the type of modification of the BLE specification needed to integrate MP into BLE.

2 The TI CC2540 Mini DK

This chapter describes the *Texas Instruments Bluetooth low energy CC2540 Mini Development Kit* (CC2540 DK). Section 2.1 introduces the hardware components of the CC2540 DK. Section 2.2 addresses the software used in combination with the CC2540 DK. In this context the development environment used for the implementation of BLE applications for the CC2540 DK and the TI BLE stack are depicted. The software used for capturing BLE packets is presented in 2.2.2.

2.1 CC2540 DK Hardware

The hardware consists of the BLE CC2540 DK by Texas Instruments (TI). Figure 2.1 shows a picture of the CC2540 DK. Some technical details are listed below [9]:

- CC2540 BLE System-On-Chip
- 8051 Microcontroller
- 8 kByte RAM
- 128 / 256 kByte flash memory

A more detailed description can be found in the data sheet [11] or in [9]. The CC2540 DK contains a USB dongle, a key fob and a debugger. The key fob is powered with a coin cell battery. The USB dongle and the debugger are powered by the USB port and do not need an extra power cord.

2.2 Software for the CC2540 DK

The environment used for development of the BLE applications is the *Embedded Workbench for 8051 v. 8.x* by *IAR Systems* [16]. Since Texas Instruments works closely with IAR Systems the support for the CC2540 DK by the Embedded Workbench is very comprehensive. Figure 2.2 shows part of a screenshot of the environment.

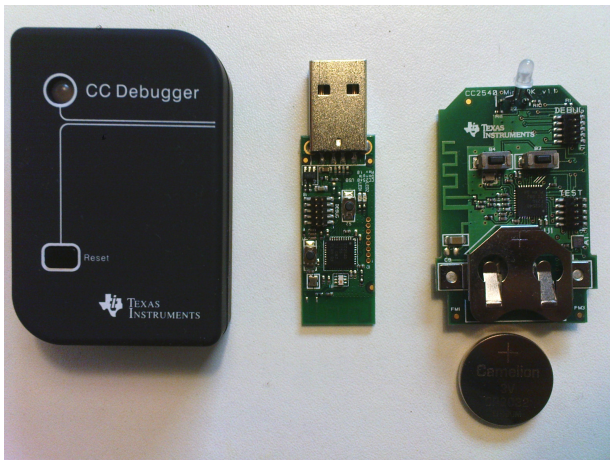

Figure 2.1: *BLE CC2540 Mini DK by Texas Instruments.*

To use the CC2540 DK hardware the BLE stack of TI is used. It is a closed source implementation of the Bluetooth Specification 4.0 [12]. It offers all necessary interfaces to develop a BLE application. To develop the MP+BLE application the sample project *SimpleBLEPeripheral* which is provided by Texas Instruments serves as the basis.[1] The sample application is written in the programming language *C*.

2.2.1 Sample Project: SimpleBLEPeripheral

The SimpleBLEPeripheral project is a sample provided by the TI BLE stack. Combined with the *SimpleBLECentral* sample project, it contains an application which is used to connect the USB dongle with the key fob. This is for demonstration purposes of the operations of basic GAP and GATT procedures [12].

This project has been used during the research of this work. It proved to be a very helpful assistant in finding the relevant sections in the Bluetooth specification to define the necessary changes. It also served as an aid to understand the correlation of the different layers of the BLE stack. Therefore a very simple *proof-of-concept* has been implemented using the *SimpleBLECentral* sample project. Every time a practical observation did not match the assumption, the event was cross-checked with the specification, hereby eliminating the (unlikely) possibility of the BLE stack not being conform with the BLE specification.

[1] More details on the used BLE stack and sample projects are documented in [10].

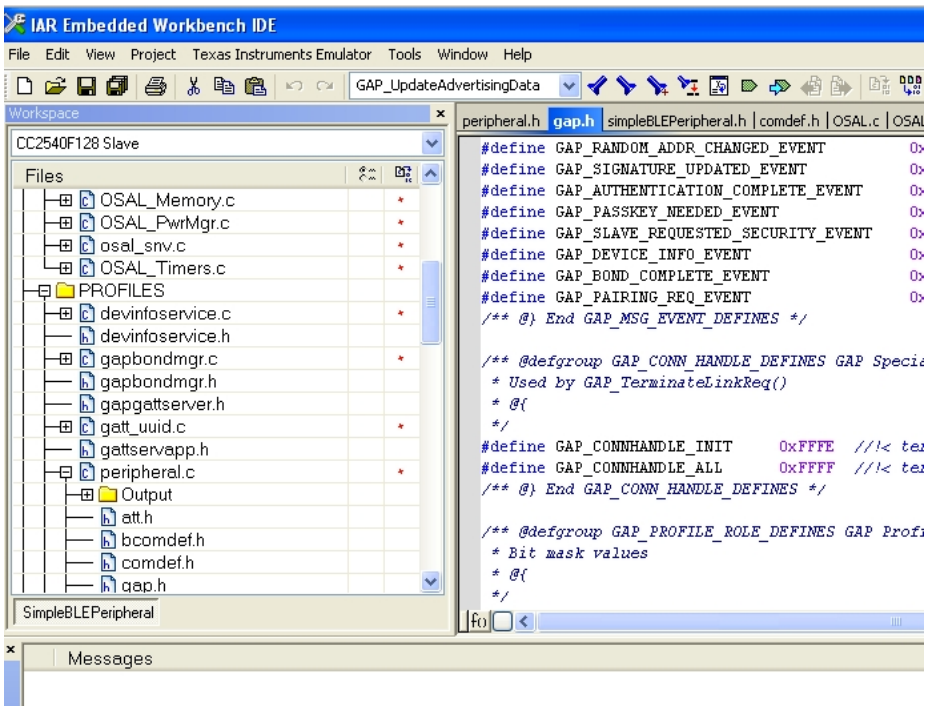

Figure 2.2: IAR Embedded Workbench: Screenshot (detail) of the application window

2.2.2 SmartRF Protocol Packet Sniffer

The *SmartRF Protocol Packet Sniffer* is a tool by Texas Instruments to capture RF packets. The packets can be displayed and stored. The tool supports many protocols, including Zigbee and BLE [13]. Figure 2.3 shows a screenshot of captured BLE advertising packets. The advertisements are generated by the key fob of the CC2540 DK. Therefore the key fob is programmed with a slightly modified version of the SimpleBLEPeripheral project, which is described in section 2.2.1. The USB dongle is flashed with the sniffing tool firmware and set to be the capturing device.

By starting to advertise packets, the key fob transmits connectable undirected advertising events (*ADV_IND*), as depicted in the *Adv PDU type* field in figure 2.3. Using these events a connection between the sniffing device and the key fob can be established. The sniffing tool also supports the creation of a connection [13]. Using the sniffing tool, one advertisement channel can be configured. For the configuration in figure 2.3 channel 37 ($0x25$) was used. For more details on remaining fields of an advertisement packets see [2, 7, 8].

During the research and experiments with the CC2540 DK the sniffing tool was used to analyze and test the implemented program. A first benchmark was done with the SmartRF

Figure 2.3: SmartRF sniffing tool: Screenshot (detail) of the application window [13]

tool by counting the advertisement packets which were sent during a defined period of time. Since the main concept of BLE with MP is based on the design of each advertisement packet, the SmartRF tool was quite helpful while getting started on the software development of BLE applications.

3 Performance Analysis

As depicted in [7], the newly introduced advertisements are used to broadcast the puzzles. Since MP only offers security for a limited period of time, BLE with MP is a time critical protocol. Therefore the key agreement phase has to finish within a defined time interval. In this section the use case assumes a secret key to be valid for seven days. This chapter depicts a performance analysis considering the optimal and the measured bit rate, as well as the practical performance, using an unmodified BLE stack.

3.1 Setup of the Performance Analysis

The effort that has to be invested by each party is analyzed in a way similar to [1]. Assuming the FFD is limited by the BLE data rate of 1 Mbit/s. The RFD is equipped with a 4 MHz microcontroller. Eve is assumed to be about 500 times stronger than the RFD. Furthermore, for a sufficient level of security it is assumed that Eve should need at least seven days to gain access to the shared strong key K. The length of a weak key k is set to 17 bit.

If the RFD need 0.26 ms to break one possible key and therefore an average of 17.04 s to break a puzzle, Eve only has to invest $\frac{0.26\,\text{ms}}{500} = 0.00052\,\text{ms}$ for one key k and $0.00052\,\text{ms} \cdot \frac{2^{17}}{2} = 34.08\,\text{ms}$ to break one puzzle. The number of puzzles that have to be broadcasted in order to distract Eve for approximately one week can be calculated with the expression:

$$n = \frac{604800000\,\text{ms}}{34.08\,\text{ms}} \cdot 2 \approx 35494291 \tag{3.1}$$

To identify all puzzles $\log_2 35494291 \approx 26$ bit are required. As depicted in [1] about 12 bit for padding are needed. Since each PDU payload is 248 bit, this leaves space for 210 bit for each K.

However K needs to be chosen in regard to the length of k and n. For k and n set to the values above, the *break-even point* for K is 42 bit. For $K = 41$ bit, Eve would simply have to invest 6.6 days to break K, instead of 7 days for $\frac{n}{2}$ puzzles. According to this a K stronger than 42 bit does not provide any more security. Regarding the block size of RC5 two puzzles of 96 bit each may be put in every PDU, thus resulting in an LL packet size of 320 bit. When

sending each advertisement exactly once, the effort invested by the FFD to send all packets is calculated as follows:

$$t_{\text{FFD}} = \frac{17747146 \cdot 320\,\text{bit}}{1000000\,\frac{\text{bit}}{\text{s}}} \approx 1.6\,\text{h} \tag{3.2}$$

Since the puzzles in each PDU are not connected with one another and the RFD waits until all packets are sent to announce its choice, this does not mean any less security.

3.2 Performance Analysis with Measured Bit Rate

The assumption of BLE with MP operating with 1 Mbit/s is quite optimistic. To put this approach into a more practical perspective, this section is devoted to calculating a realistic throughput when broadcasting the puzzles via BLE. In [3] the authors present an analysis of BLE's maximum throughput. Although their work offers an maximum bit rate of 236.7 kbit/s in practice, this may not be guaranteed for BLE with MP. For one they concentrate on connection events and thus the data actually transferred between two paired devices. In the BLE with MP approach advertisements are used, which do not require a connection. Secondly the measured maximum bit rate is observed in the absence of bit errors [3].

For demonstration purposes the bit rate is assumed to be 236.7 kbit/s as presented in [3]. Considering this, the equation 3.1 in section 3.1 is updated to:

$$t_{\text{FFD}} = \frac{17747146 \cdot 320\,\text{bit}}{236700\,\frac{\text{bit}}{\text{s}}} \approx 6.7\,\text{h} \tag{3.3}$$

This concludes that the actual performance is about four times lower than assumed in section 3.1. However the bit rate claimed in [3] is the maximum, not the average bit rate. Therefore 6.7 hours is the minimum performance of BLE with MP. To actually measure the performance many factors must be considered, for instance, the amount of devices communicating in range of the FFD and RFD, or obstacles blocking the radio signals and therefore decreasing the bit rate. The next section analyzes the performance and also uses actual benchmark results.

3.3 Performance with Timeout

This section demonstrates the performance of BLE with MP solely using the supported advertisement types. To broadcast the puzzles, one of four advertisements types defined in the

specification may be used. The following list shows all available types, including the timeout (*TO*) between each data packet [2]:

- ADV_IND: connectable undirected advertising event ($TO >= 20\,\text{ms}$)
- ADV_DIRECT_IND: connectable directed advertising event ($TO >= 20\,\text{ms}$)
- ADV_NONCONN_IND: non-connectable undirected advertising event ($TO >= 100\,\text{ms}$)
- ADV_SCAN_IND: scannable undirected advertising event ($TO >= 100\,\text{ms}$)

In addition to *TO* a randomly chosen timeout, TO_{rand}, is added. It may vary between 0 to 10 ms [2]. Hereby *TO* is the major flaw when applying MP. A payload of 31 byte is required as described in [8]. For further calculation the advertisement type *ADV_IND* is chosen, since it has the smallest *TO*. For the use with MP the latency, Γ, is described by

$$\Gamma_{TO} = \sum_{1}^{n} TO + TO_{rand} \tag{3.4}$$

As depicted in section 3.1, 35 494 291 puzzles and therefore 17 747 146 advertisement packets have to be sent in this use case. Considering *TO* to be 20 ms and a TO_{rand} having an average value of 5 ms, the period of busy waiting can be calculated using the equation:

$$\sum_{1}^{17\,747\,146} 25\,\text{ms} = 443\,678\,650\,\text{ms} \approx 5.14\ \text{days} \tag{3.5}$$

When running first benchmarks with the CC2540 DK, an average of 40 packets per second has been observed using the SmartRF tool [6]. This correlates with the calculated value of:

$$\frac{17\,747\,146\,\text{adv}}{40\,\text{adv / s}} = 443\,678.65\,\text{s} \approx 5.14\ \text{days} \tag{3.6}$$

This concludes that the actual transmission of advertisements is including in *TO*. Therefore the result from section 3.2 must not be added to equation 3.3 respectively 3.3. For the FFD and RFD to take 5.14 days to agree upon a key which is secure for seven days is highly unpractical. Therefore a new advertisement type (*ADV_BLEMP*) as depicted in [7] is required. To actually obtain expressive benchmarks the value of *TO* has to be excluded from the procedure. Therefore an open source BLE stack is required. Since the CC2540 DK only offers an closed source stack, it is not considered for future research.

4 Overview and Outlook

This work gave an overview of the CC2540 DK by Texas Instruments. In Chapter 2 the hardware of the CC2540 DK, which consists of a key fob, a USB dongle and the CC Debugger, was introduced. Section 2.2 gave an insight to the associated software. The IAR Embedded Workbench was used for the implementation of a BLE application which simply advertises data packets. Hereby the source code of the SimpleBLEPeripheral has been modified. The SmartRF tool was used for capturing the advertisements and to run a first benchmark.

Chapter 3 depicted a performance analysis making three different assumptions. First the optimal bit rate for a certain use case has been used to estimate the performance. Secondly the measured and more practical bit rate by [3] was used, thus resulting in a lower performance. Finally the timeouts defined in the BLE specification have been considered. This concluded the need for a new advertisement type as depicted in [7]. To actually implement a version of a BLE stack supporting MP and to state expressive benchmark results, an open source BLE stack is vitally important. Therefore the further use of the CC2540 DK is not intended.

MP coupled with BLE is a relatively slow protocol. Although some modifications have been proposed which increase the performance, a key agreement in a *over-night-scenario* seems feasible. Goal of this work is to establish a concept of a BLE specification that includes MP as an optional security feature. Use cases will be provided, in which this procedure holds many advantages, such as medical, biological, or sport applications. Once an open source stack and supporting hardware is available, an actual implementation is viable.

Bibliography

[1] F. Armknecht and D. Westhoff. Using Merkle's Puzzle for Key agreement with Low-end Devices. *IEEE 34th Conference on Local Computer Networks, 2009. LCN 2009.*, pages 858–864, December 2009.

[2] BLUETOOTH SPECIFICATION Version 4.0. Document - Bluetooth SIG, June 2010.

[3] C. Gomez, I. Demirkol, and J. Paradells. Modeling the Maximum Throughput of Bluetooth Low Energy in an Error-Prone Link. *Communications Letters, IEEE*, 15(11):1187 –1189, november 2011.

[4] R. Merkle. Secure Communications Over Insecure Channels. *Communications of the ACM*, pages 294–299, April 1978.

[5] H. Perrey. Angriffe auf Funknetzwerke. Ausarbeitung Masterkurs „Anwendungen 1", Hochschule für Angewandte Wissenschaften, Hamburg, February 2011.

[6] H. Perrey. Attacks on Wireless Networks: Outline. Ausarbeitung Masterkurs „Master Seminar", Hochschule für Angewandte Wissenschaften, Hamburg, February 2012.

[7] H. Perrey. The Concept of Bluetooth Low Energy with Merkle's Puzzle. Ausarbeitung Masterkurs „Projekt 1", Hochschule für Angewandte Wissenschaften, Hamburg, February 2012.

[8] H. Perrey, O. Ugus, and D. Westhoff. WiSec' 2011 poster: Security Enhancement for Bluetooth Low Energy with Merkle's Puzzle. *ACM SIGMOBILE: Mobile Computing and Communications Review*, 15(3):45–46, July 2011.

[9] B. Tanner and G. Gräni. Sensornetzwerk mit Bluetooth Low Energy. Bachelorarbeit, Zürcher Hochschule für Angewandte Wissenschaften, Winterthur, June 2011.

[10] Bluetooth low energy software stack and tools. Texas Instruments. http://focus.ti.com/docs/toolsw/folders/print/ble-stack.html?DCMP=RF/IFANDZIGBEE& - last checked: 11.03.2012.

[11] 2.4-GHz Bluetooth® low energy System-on-Chip. Texas Instruments - Document. http://www.ti.com/lit/ds/symlink/cc2540.pdf - last checked: 19.02.2012.

[12] Bluetooth® Low Energy Software Developer's Guide v1.2. Texas Instruments - Document. http://www.ti.com.cn/cn/lit/ug/swru271b/swru271b.pdf - last checked: 09.03.2012.

[13] SmartRFTM Packet Sniffer User Manual. Texas Instruments - Document. http://www.ti.com/general/docs/lit/getliterature.tsp?literatureNumber=swru187f&fileType=pdf - last checked: 18.05.2012.

[14] Casio Bluetooth Watch Puts Your Phone's Info on Your Wrist. webpage, February 2012. http://gizmodo.com/5725608/casio-bluetooth-low-energy-watch-prototype-has-2-year-battery-life last checked: 10.02.2012.

[15] Bluetooth.com. webpage, 2012. http://www.bluetooth.com/ - last checked: 02.03.2012.

[16] IAR Systems Homepage. webpage. http://www.iar.com/ - last checked: 19.02.2012.

[17] iPhone 4S. webpage, March 2012. http://www.apple.com/de/iphone/specs.html last checked: 02.03.2012.

[18] Homepage of Ralph Merkle. webpage. http://www.merkle.com/ - last checked: 19.02.2012.

[19] mossmann's blog. webpage. http://ossmann.blogspot.com/ - last checked: 19.02.2012.

[20] Project Ubertooth. webpage. http://ubertooth.sourceforge.net/ - last checked: 19.02.2012.