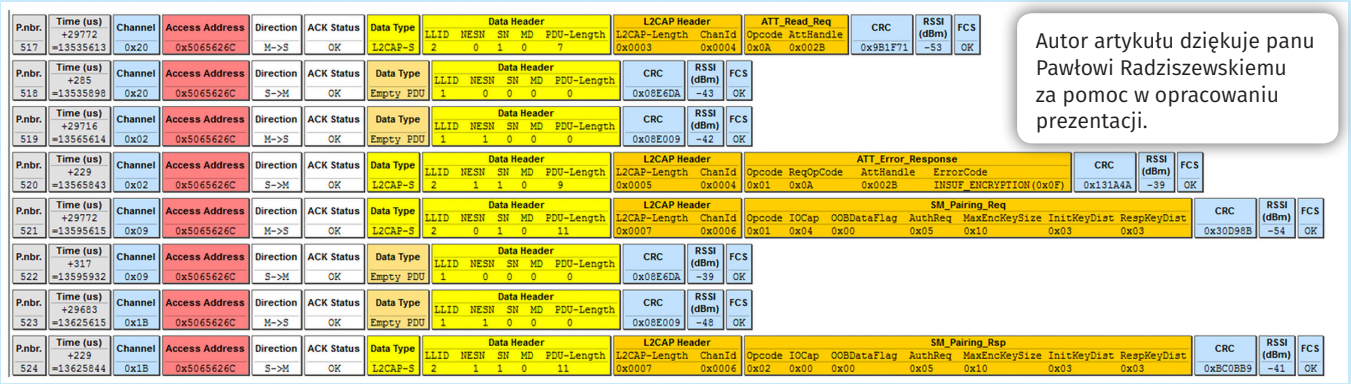


P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header	L2CAP Header	ATT_Read_Req	CRC	RSSI (dBm)	FCS
							LLID NESN SN MD PDU-Length	L2CAP-Length ChanId	Opcode AttHandle			
517	+29772 =13535613	0x20	0x5065626C	M->S	OK	L2CAP-S	2 0 1 0 7	0x0003 0x0004	0x0A 0x002B	0x9B1F71	-53	OK

P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header	CRC	RSSI (dBm)	FCS
							LLID NESN SN MD PDU-Length			
518	+285 =13535898	0x20	0x5065626C	S->M	OK	Empty PDU	1 0 0 0 0	0x08E6DA	-43	OK
519	+29716 =13565614	0x02	0x5065626C	M->S	OK	Empty PDU	1 1 0 0 0	0x08E009	-42	OK

P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header	L2CAP Header	ATT_Error_Response	CRC	RSSI (dBm)	FCS
							LLID NESN SN MD PDU-Length	L2CAP-Length ChanId	Opcode ReqOpCode AttHandle ErrorCode			
520	+229 =13565843	0x02	0x5065626C	S->M	OK	L2CAP-S	2 1 1 0 9	0x0005 0x0004	0x01 0x0A 0x002B INSUF_ENCRYPTION(0x0F)	0x131A4A	-39	OK

P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header	L2CAP Header	SM_Pairing_Req	CRC	RSSI (dBm)	FCS
							LLID NESN SN MD PDU-Length	L2CAP-Length ChanId	Opcode IOCap OOBDataFlag AuthReq MaxEncKeySize InitKeyDist RespKeyDist			
521	+29772 =13595615	0x09	0x5065626C	M->S	OK	L2CAP-S	2 0 1 0 11	0x0007 0x0006	0x01 0x04 0x00 0x05 0x10 0x03 0x03	0x30D98B	-54	OK

P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header	CRC	RSSI (dBm)	FCS
							LLID NESN SN MD PDU-Length			
522	+317 =13595932	0x09	0x5065626C	S->M	OK	Empty PDU	1 0 0 0 0	0x08E6DA	-39	OK
523	+29683 =13625615	0x1B	0x5065626C	M->S	OK	Empty PDU	1 1 0 0 0	0x08E009	-48	OK

P.nbr.	Time (us)	Channel	Access Address	Direction	ACK Status	Data Type	Data Header	L2CAP Header	SM_Pairing_Rsp	CRC	RSSI (dBm)	FCS
							LLID NESN SN MD PDU-Length	L2CAP-Length ChanId	Opcode IOCap OOBDataFlag AuthReq MaxEncKeySize InitKeyDist RespKeyDist			
524	+229 =13625844	0x1B	0x5065626C	S->M	OK	L2CAP-S	2 1 1 0 11	0x0007 0x0006	0x02 0x00 0x00 0x05 0x10 0x03 0x03	0xBC0BB9	-41	OK

Autor artykułu dziękuje panu Pawłowi Radziszewskiemu za pomoc w opracowaniu prezentacji.

Systemy dla Internetu Rzeczy (19)

Bezpieczeństwo transmisji z protokołem Bluetooth Low Energy

Jednym z najważniejszych wymagań stawianych systemom Internetu Rzeczy (IoT) jest szeroko rozumiane bezpieczeństwo danych. Dużym problemem jest bezpieczne przekazanie informacji pomiędzy węzłem centralnym i węzłami sensorów oraz dołączanie nowego węzła do sieci. Rozwiązaniem problemu może być zastosowanie metod bezpieczeństwa oferowanych przez protokół Bluetooth Low Energy (BLE).

Rozwój układów scalonych przeznaczonych dla komunikacji bezprzewodowej, a szczególnie układów z obsługą protokołu BLE 5, doprowadził do udostępnienia sprzętowego wspomagania zaawansowanych metod szyfrowania [2]. Daje to możliwość zastosowania szyfrowania z algorytmami na krzywych eliptycznych.

Układy scalone z rodziny CC13x2/CC26x2 mają moduły sprzętowe wspomagania bezpieczeństwa: AES-256 (Advanced Encryption Standard), TRNG (True Random Number Generator), unikalny numer ID, bezpieczna pamięć RAM i Flash, bezpieczny transfer DMA, moduł Hash (SHA-2) oraz moduł Public Key Acceleration (PKA) [1]. Moduł AES może pracować w trybach: ECB, CBC; CBC-MAC, CTR, CCM oraz GCM. Ze sprzętowym wspomaganiem obliczeń na krzywych eliptycznych. AES odnosi się do szyfrowania symetrycznego; krzywe eliptyczne – do asymetrycznego. Pozwala to na pełną realizację metod bezpieczeństwa zalecanych przez specyfikację BLE 5 [14]. Zaawansowane szyfrowanie i deszyfrowanie odbywa się „w locie" (przepustowość 118 Mbps). Pozwala to znacząco odciążyć CPU i zmniejszyć pobór mocy.

Typowym problemem podczas dołączania nowego węzła do sieci jest bezpieczna wymiana kluczy. Konieczne jest uniknięcie podsłuchania transmisji klucza przez hakera. W celu uniknięcia tego problemu najczęściej stosowane jest szyfrowanie asymetryczne. W takim przypadku węzeł generuje klucz prywatny i klucz publiczny. I przesyła klucz publiczny do węzła centralnego. Węzeł ten szyfruje informacje z użyciem tego klucza a deszyfruje z użyciem klucza prywatnego. Klucz prywatny nigdy nie jest przesyłany przez radio, dlatego nie jest widoczny dla podsłuchiwacza.

Kolejnym problemem jest unikanie ataku MITM (Man in the Middle). W tym przypadku haker może „wtrącić się" pomiędzy dwa komunikujące się urządzenia. W celu uniknięcia tego problemu wiele systemów wykonuje uwierzytelnianie poprzez osobny kanał komunikacyjny (Out of Band). Dobrym przykładem jest przesyłanie kodu PIN przez telefon komórkowy podczas transakcji internetowych. Rozwiązaniem może być zastosowanie technologii NFC (Near Field Communication). Krótki zasięg takiej transmisji jest skuteczną przeszkodą dla hakera.

Pierwszą operacją przy zestawianiu połączenia w standardzie BLE jest parowanie (patrz ramka „Słownik określeń"). Serwer GATT stosu BLE może definiować zezwolenie dostępu osobno dla każdej charakterystyki [13]. Niektóre charakterystyki mogą mieć dostęp dla każdego klienta. Inne mogą mieć dostęp tylko dla klienta uwierzytelnionego. Charakterystyki, które wymagają uwierzytelniania,

Słownik określeń stosowanych przy transmisji BLE

- Parowanie (pairing) – proces wymiany kluczy.
- Uwierzytelnianie (authentication) – Proces parowania przeprowadzony z zastosowaniem techniki MIMT.
- Szyfrowanie (encryption) – Dane są szyfrowane po parowaniu lub po ponownym wznowieniu połączenia, gdy klucze są pobrane z pamięci nieulotnej.
- Wiązanie (bonding) – Zapisywanie kluczy w pamięci nieulotnej, w celu zastosowania po ponownym wznowieniu połączenia.
- MITM (Man in the Middle protection) – Zabezpieczenie przed atakiem poprzez podsłuchanie transmisji radiowej.
- Dołączanie (Commissioning) – Dodawanie nowego węzła (urządzenia) do sieci bezprzewodowej lub zestawianie połączenia pomiędzy dwoma urządzeniami.
- Autoryzacja (Authorization) – Dodatkowy poziom wymiany klucza wykonywany w aplikacji, po wykonaniu uwierzytelniania.

Poprzednie części kursu i dodatkowe materiały dostępne są na stronie www.media.avt.pl

są dostępne dopiero wtedy gdy klient przejdzie przez proces parowania z uwierzytelnianiem. Ta weryfikacja jest wykonywana przez stos i nie wymaga obsługi przez aplikację. Jedyne wymaganie dla charakterystyki to poprawne zarejestrowanie w serwerze GATT.

Autoryzacja jest dodatkową warstwą bezpieczeństwa, oprócz mechanizmów zaimplementowanych przez stos BLE. Aplikacja definiuje własne wymagania na autoryzację. Autoryzacja jest wykonywana w ścisłej współpracy ze stosem BLE.

Menedżer wiązania

Menedżer wiązania (GAP Bond Manager) jest modułem, który realizuje mechanizmy bezpieczeństwa [12].

Ogólnie biorąc menedżer wiązania wykonuje następujące kroki:

1. Parowanie wymienia klucze z zastosowaniem jednego z kilku trybów.
2. Szyfrowanie transmisji z zastosowaniem kluczy z kroku 1.
3. Wiązanie zapisuje klucze w bezpiecznej pamięci Flash.
4. Zastosowanie kluczy zapamiętanych w pamięci do szyfrowania transmisji po ponownym wznowieniu połączenia. Nie jest wtedy wymagane ponowne przeprowadzenie parowania z uwierzytelnianiem.

Nie wszystkie kroki musza być wykonywane. np. może być wykonywane parowanie ale bez wiązania.

Typy parowania

Specyfikacja BLE 4.2 wprowadziła nowy tryb parowania *Secure Connections* [13]. W tym trybie stosowana jest negocjacja kluczy Elliptic Curve Diffie-Hellman. Poprzednie sposoby parowania definiowane przez BLE 4.0 i BLE 4.1 są dalej dostępne i określane jako *LE Legacy*. Nie stosują one szyfrowania z algorytmami na krzywych eliptycznych.

Są dostępne cztery typy parowania:

- **Just Works (*Secure Connections* lub *LE Legacy*)**. Metoda parowania, w której klucze są przesyłane przez transmisję radiową, bez stosowania zabezpieczenia MITM. Nie wymaga żadnych sposobów interakcji z użytkownikiem. Podatna na atak MITM.
- **Passkey Entry (*Secure Connections* lub *LE Legacy*)**. W tej metodzie jedno urządzenie wyświetla 6-cyfrowe hasło a na drugim urządzeniu to hasło jest wprowadzane. Od udostępnianych przez urządzenia możliwości interakcji zależy, które z nich pełni którą rolę. Hasło jest generowane losowo. Passkey Entry jest typem parowania z uwierzytelnianiem, który zapobiega atakowi MITM.
- **Numeric Comparison (*Secure Connections*)**. W tej metodzie oba urządzenia pokazują 6-cyfrowe hasło. Po porównaniu obu haseł trzeba obu urządzeniom zasygnalizować rezultat: Tak/Nie. Typowo są do tego używane przyciski. Numeric Comparison jest typem parowania z uwierzytelnianiem, który zapobiega atakowi MITM.
- **Out of Band (*Secure Connections* lub *LE Legacy*)**. W tej metodzie klucz nie jest wymieniany poprzez transmisję BT. Raczej stosowane są inne sposoby, jak port szeregowy lub NFC (też transmisję radiową, ale o bardzo małym zasięgu)

Stosowane do komunikacji tryby i typy parowania zależą od wyposażenia obu komunikujących się urządzeń (inicjatora połączenia i odpowiadającego). Na przykład, czy posiada klawiaturę, wyświetlacz lub żadnej możliwości interakcji. W dokumentacji Bluetooth Core Specification Version 5.0 są zamieszczone dokładne ich opisy w rozdziale *Selecting Key Generation Method section* ([Vol 3], Part H, Section 2.3.5.1) [14].

Są jeszcze inne metody poprawienia bezpieczeństwa dostarczane przez specyfikację BLE. Na przykład w metodzie *LE Privacy* w trakcie połączenia zmieniany jest często adres urządzenia. Zmniejsza to możliwość śledzenia urządzenia [13].

Próba odczytu przez klienta bez uwierzytelnienia jest pokazana na rysunku w nagłówku artykułu [13]. Żądanie zostało automatycznie odrzucone przez stos BLE z zasygnalizowaniem błędu.

Dokumentacja

Dla rodziny CC13x2 opis dotyczący bezpieczeństwa komunikacji BLE ukryty jest w portalu TIREX [10]. Dla procesora CC1352R1F3 opis znajduje się w pakiecie *SimpleLink CC13x2 Software Development Kit* [4]. Trzeba nawigować do ścieżki *Documents → Documentation Overview* oraz *TI BLE 5-Stack BTool User's Guide*. Wtedy otwierany jest document BTool Guide 5.00.00 [12].

W pakiecie *SimpleLink Academy 2.20.03 for SimpleLink CC13x2 SDK 2.20* [7] jest bardzo ciekawe ćwiczenie laboratoryjne *Bluetotth 5 Fundamentals* [8]. Zawiera ono też wariantowy opis użycia aplikacji BTool. Wykorzystywane są w nim przykładowe projekty z pakietu CC13x2 SDK [4] udostępnione w ścieżce *Examples → Development Tools → CC1352R LaunchPad → ble5stack*. Są to przede wszystkim projekty: host_test, project_zero i inne. Każdy projekt zawiera plik *README.html* w którym jest zamieszczony opis aplikacji projektu. Niestety, nie jest on widoczny w portalu TIREX ani w środowisku CCS. Jest za to umieszczony w lokalnym folderze przykładu pakietu CC13x2 SDK [5, 6].Są to całkiem spore opisy i zagadką jest, dlaczego jest taki kiepski do nich dostęp.

Opis nawigowania po serwisach GATT został zamieszczony w poprzednim odcinku kursu „Bluetooth Low Energy" [S7].

W ramach nowych pakietów programowych SDK dla układów rodziny CC12x2R/CC26x2R platformy SimpleLink, firma Texas Instruments udostępniła aplikację BTool (Bluetooth Low Energy Application) [12]. Aplikacja BTool pracuje na komputerze PC i komunikuje się z modułem sprzętowym CC1352R1 LaunchPad (lub CC26x2R LaunchPad) poprzez łącze UART (COM) dostarczane przez emulator XDS110 tego modułu. Moduł LaunchPad musi mieć wpisany kod programu Host Test i pracuje jako procesor sieciowy (network processor). Aplikacja BTool komunikuje się łączem UART z modułem LaunchPad przy zastosowaniu poleceń HCI. Dołączony moduł LaunchPad pracuje jako „central device" w sieci BLE. Taki zestaw umożliwia komunikację z układami pracującymi z protokołem BLE 5 oraz BLE 4.2.

Przygotowanie do pracy

Dokładny opis instalowania oprogramowania jest zamieszczony w poprzednim artykule „Oprogramowanie narzędziowe dla układów CC26xx i CC13xx platformy SimpleLink" [S12]. Opis zestawu startowego CC1352R1 LaunchPad jest zamieszczony w poprzednim artykule „Zestaw CC1352R1 LaunchPad" [S15].

Do wykonania zamieszczonych dalej zadań są potrzebne:

- Dwa zestawy startowe CC1352R1 LaunchPad [3] (lub CC2652R1 LaunchPad).
- Aplikacja CCS 8.1 z zainstalowaną obsługą układów CC13xx/CC26xx
- Zainstalowany pakiet programowy CC13x2 SDK [4] (lub CC26x2 SDK w przypadku stosowania zestawu startowego CC2652R1 LaunchPad)
- Zainstalowana aplikacja UniFlash 4 [9].
- Aplikacja BTool – dostarczana w ramach pakietu CC13x2 SDK (lub CC26x2 SDK)

Zadania dalej opisane są wykonywane z zastosowaniem zestawu startowego CC1352R1 LaunchPad [3]. Można je również wykonać z zastosowaniem zestawu startowego CC2652R1 LaunchPad [S13]. Trzeba wtedy jedynie zastosować dedykowany dla niego pakiet programowy CC26x2 SDK.

Zadanie 1 – Zaprogramowanie aplikacji HostTest

W dokumencie *README.html* jest zamieszczony opis aplikacji HostTest. Można go jednak otworzyć tylko w lokalnym folderze przykładu

pakietu CC13x2 SDK [4]. W systemie MS Windows dla projektu HostTest jest to ścieżka [6].

A1. Dołącz zestaw startowy CC1352R1 LaunchPad do komputera PC używając kabla USB.
A2. W oknie *Menadżer Urządzeń* sprawdź numer portu **COMx1** dla kanału *Aplication/User UART*. Zanotuj ten numer.
A3. Wystartuj aplikację *UniFlash 4*. Zostanie automatycznie wyszukany moduł LaunchPad dołączony do komputera PC.
A4. Opis pracy aplikacji *UniFlash 4* jest zamieszczony w [S15].
A5. Kliknij *OK* lub *Start*. Kliknij *Program*.
A6. W sekcji *Flash image(s)*, kliknij na przycisk *Browse*.
A7. Nawiguj do foldera plików ładowalnych C:\ti\simplelink_cc13x2_sdk_2_20_00_71\examples\rtos\CC1352R1_LAUNCHXL\ble5stack\hexfiles\cc13x2r1
A8. Zaznacz plik *ble5_host_test_cc13x2r1lp_app_FlashROM_Release.hex*
A9. Kliknij *Otwórz*. Kliknij przycisk *Load Image*.
A10. Czekaj na informację o poprawnie zakończonej operacji.
A11. Zamknij aplikację *UniFlash 4*.
A12. Odłącz zestaw startowy CC1352R1 LaunchPad do komputera PC.
A13. Oznacz moduł jako: ***HostTest***

Zadanie 2 – Zaprogramowanie aplikacji BLE ProjectZero

B1. Dołącz drugi zestaw startowy CC1352R1 LaunchPad do komputera używając kabla USB.
B2. Wystartuj środowisko CCS.
B3. Poczekaj na zakończenie sprawdzania aktualizacji środowiska. Jedyny sposób to obserwowanie informacji o postępie sprawdzania wyświetlanych na pasku stanu.

Importuj projekt aplikacji ProjectZero

B4. Kliknij na przycisk *Browse Examples*.
W oknie *Resource Explorer* rozwiń ścieżkę:
SimpleLink CC13x2 SDK – v. 2.20.00.71 → Examples → Development Tools → CC1352R LaunchPad → ble5stack → project_zero → TI-RTOS → CCS Compiler → project_zero_app
B5. W prawym oknie kliknij na przycisk *Import to IDE* 
B6. W wyświetlanym oknie kliknij na *I Have Read And Agree*.
B7. Poczekaj na zakończenie pobierania trzech projektów. To może chwilę trwać. Należy obserwować pasek stanu na dole okna. Szczególnie trzeba poczekać aż znikną znaczki wykrzyknika nałożonego w oknie *Project Explorer* na ikonki folderów projektów.
Aplikacja składa się z trzech projektów: app (aplikacja), library (biblioteka stosu) oraz bim (bootloader).

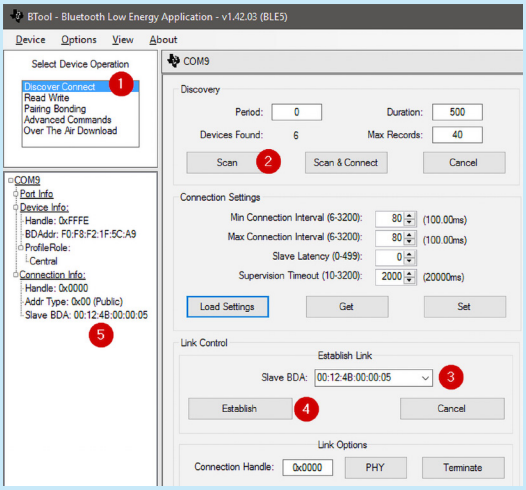
W dokumencie *README.html* jest zamieszczony opis aplikacji BLE ProjectZero. Można go jednak otworzyć tylko w lokalnym folderze przykładu pakietu CC13x2 SDK [4]. W systemie MS Windows dla projektu BLE ProjectZero jest to ścieżka [5].

Wykonaj budowanie wszystkich projektów

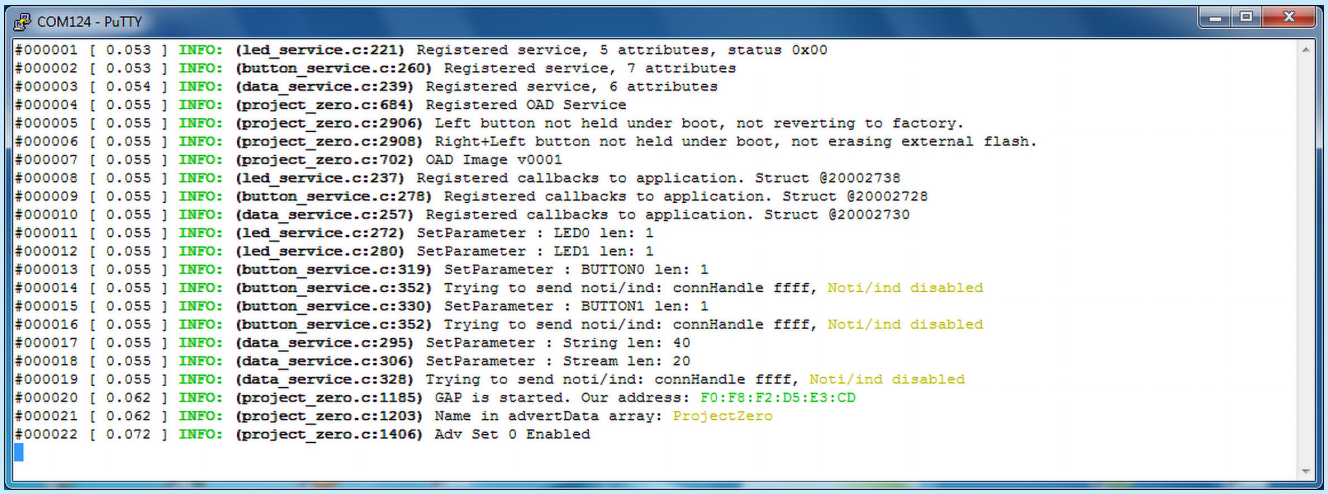
B8. W oknie *Project Explorer* kliknij na linię projektu ***ble5_project_zero_cc13x2r1lp_app*** (wybierz go).
B9. Wybierz z menu *Project → Clean*. W oknie *Clean* kliknij na przycisk *Clean*. Spowoduje to wykonanie budowania wszystkich projektów.
B10. Czekaj na pełne zakończenie budowania wszystkich projektów. Na pasku stanu jest pokazywany postęp w procentach. Budowanie jest wieloprzebiegowe. W oknie *Console* pokazywane są informacje o postępach budowania. Istotną dla zakończenia budowania jest informacja o wygenerowaniu pliku *.out.

Zaprogramuj projekt BIM

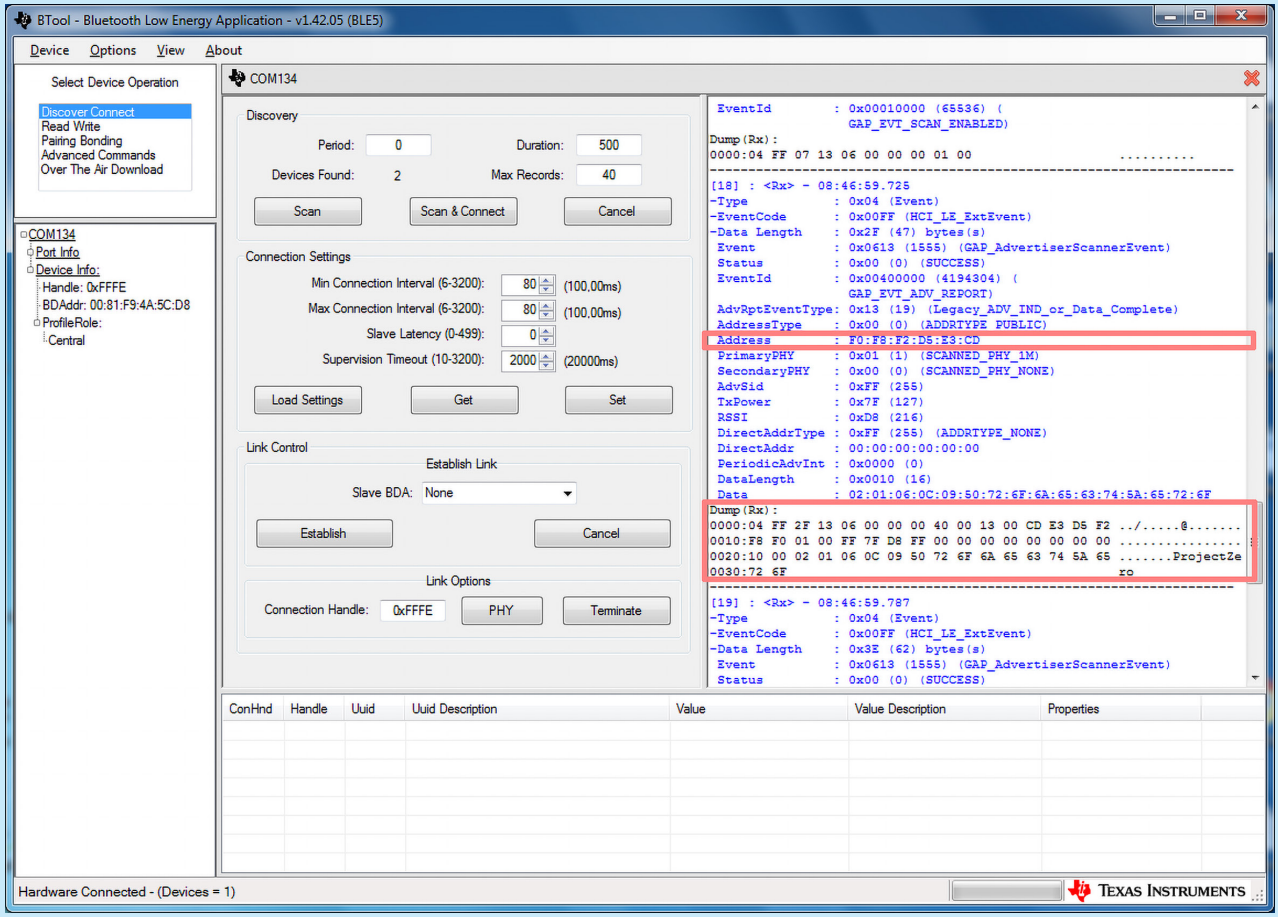
BIM (Boot Image Manager) to bootloader, który ładuje obraz kodu aplikacji do pamięci rdzeni procesora. Kod projektu BIM musi być wpisany do pamięci Flash procesora jako pierwszy. Opis BIM jest zamieszczony w dokumencie BLE 5-Stack User's Guide [11].
B11. W oknie *Project Explorer* kliknij na linię projektu ***cc13x2r1lp_bim_offchip*** (wybierz go).
Uwaga: To musi być projekt BIM. Linia wybranego projektu jest wyświetlana z pogrubieniem i uzupełniona przez informację [Active – Debug].



Rysunek 2. Kroki pracy z aplikacją BTool [8]

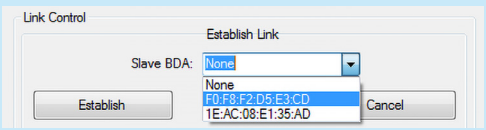


Rysunek 1. Okno aplikacji PuTTy dołączonej do modułu z aplikacją *ProjectZero*



Rysunek 3. Okno aplikacji BTool po wykryciu urządzenia z aplikacją ProjectZero

B12. W oknie *Project Explorer* kliknij prawym klawiszem myszki na linię projektu **cc13x2r1lp_bim_offchip** i wybierz *Debug As → Code Composer Debug Session*.



Rysunek 4. Wybór urządzenia z adresem uzyskanym z okna PuTTY

B13. Czekaj na zatrzymanie programu w pierwszej instrukcji funkcji *main()* w pliku *bim_main.c*.

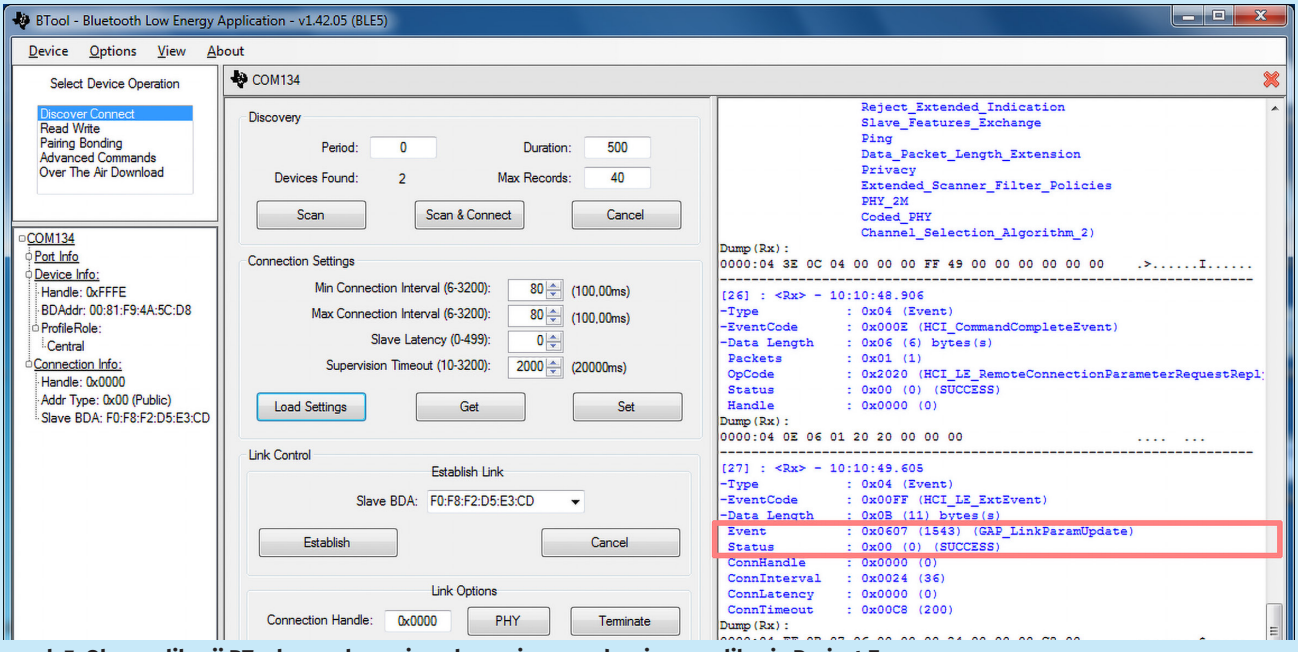
B14. W perspektywie CCS Debug zakończ sesję debugową. Kliknij na Terminate ■.

B15. Czekaj aż CCS przełączy widok na perspektywę CCS Edit.

Zaprogramuj projekt ProjectZero

B16. W oknie *Project Explorer* kliknij na linię projektu *ble5_project_zero_cc13x2r1lp_app*.

Uwaga: To musi być projekt ProjectZero.



Rysunek 5. Okno aplikacji BTool po wykonaniu połączenia z urządzeniem z aplikacją Project Zero

```
#000023 [ 5484.529 ] INFO: (project_zero.c:1411) Adv Set 0 Disabled
#000024 [ 5484.529 ] INFO: (project_zero.c:1431) Adv Set 0 disabled after conn 0
#000025 [ 5484.529 ] INFO: (project_zero.c:1240) Link establish event, status 0x00. Num Conns: 1
#000026 [ 5484.530 ] INFO: (project_zero.c:1253) Connected. Peer address: 00:81:F9:4A:5C:D8
#000027 [ 5484.585 ] INFO: (project_zero.c:1406) Adv Set 0 Enabled
#000028 [ 5491.379 ] INFO: (project_zero.c:1694) Updated params for 00:81:F9:4A:5C:D8, interval: 36 ms, latency: 0, timeout: 200 ms
```

Rysunek 6. Informacje aplikacji *ProjectZero* o wykonaniu połączenia

B17. Kliknij na przycisk „*Debug*".

B18. Czekaj na zatrzymanie programu w pierwszej instrukcji funkcji *main()* w pliku *main.c*. Nie w poprzednim pliku *bim_main.c*. Musi być widoczna strzałka po lewej stronie kodu.

Dołącz program terminala

B19. W oknie *Menadżer Urządzeń* sprawdź numer portu **COMx2** dla kanału *Aplication/User UART*. Zanotuj ten numer.

B20. Wystartuj aplikację terminala *PuTTY* (dokładny opis w [S15]).

B21. Kliknij przycisk *Serial*. W polu *Serial line* wpisz numer portu szeregowego COMx2. W polu *Speed* wpisz prędkość transmisji 115200.

B22. W oknie *Category* kliknij na *Colours*. Zaznacz *Use system colours*. Kliknij *OK*.

B23. Rozciągnij okno aplikacji PuTTy wszerz (ważne).

Uwaga: Nie odłączaj modułu CC1352R1 LaunchPad od komputera jeśli program terminala PuTTY jest z nim połączony.

B24. W perspektywie *CCS Debug* aplikacji CCS kliknij na przycisk „*Resume*".

B25. W oknie aplikacji PuTTY Sprawdź adres urządzenia: „Our address: F0:F8:F2:D5:E3:CD" (**rysunek 1**).

Z informacji w oknie PuTTy można zobaczyć, że aplikacja inicjalizuje trzy serwisy: LED, Button i Data. Następnie ustawia ich początkowe wartości oraz sygnalizuje, że urządzenie jest gotowe i rozpoczęło rozgłaszanie.

Zadanie 3 – Wykonaj połączenie z aplikacją BTool

Czas na zastosowanie aplikacji BTool.

C1. Dołącz pierwszy zestaw startowy CC1352R1 LaunchPad z zaprogramowana aplikacją HostTest do komputera używając kabla USB.

C2. W oknie *Menadżer Urządzeń* sprawdź numer nowego portu COMx1 dla kanału *Aplication/User UART*. Powinien być on zgodny z numerem uzyskanym w poleceniu A2.

Wystartuj aplikację BTool

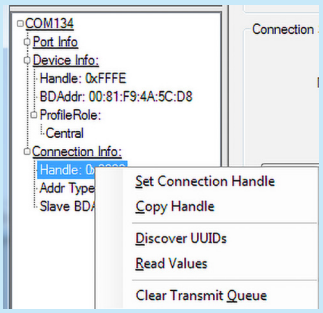
Pliki aplikacji BTool są zamieszczone w folderze pakietu programowego SDK. Należy zastosować wersję aplikacji dołączoną do pakietu programowego z którego była pobrana aplikacja HostTest. W tym przypadku jest to ścieżka *C:\ti\simplelink_cc13x2_sdk_2_20_00_71\tools\ble5stack\btool*.

C3. Wystartuj aplikację BTool. Dukliknij na plik ***btool.exe***.

C4. W oknie W oknie *Serial Port Settings* wybierz port COMx1 (dla aplikacji HostTest).

C5. Kliknij OK.

Aplikacja BTool po pomyślnym połączeniu poprzez łącze UART z modułem LaunchPad otwiera okno podzielone na trzy kolumny (rysunek 3). W lewej kolumnie są pokazywane informacje dotyczące statusu dołączonych urządzeń. W prawej kolumnie jest pokazywany log wiadomości przesyłanych przez aplikację BTool do modułu LaunchPad



Rysunek 7. Wybór operacji z dołączonym urządzeniem

ConHnd	Handle	Uuid	Uuid Description	Value	Value Description	Properties
0x0000	0x0021	0x2800	GATT Primary Service Declaration	00:00:00:00:00:00:00:B0:00:40:51:0...	Project Zero LED Service Declaration	
0x0000	0x0022	0x2803	GATT Characteristic Declaration	0E:23:00:00:00:00:00:00:00:00:B0:0...		Rd Wwr Wr 0x0E
0x0000	0x0023	0xF00011104514000B00...	LED0 State	00	Send 00 to turn off, higher values to turn on	Rd Wwr Wr 0x0E
0x0000	0x0024	0x2803	GATT Characteristic Declaration	0E:25:00:00:00:00:00:00:00:00:B0:0...		Rd Wwr Wr 0x0E
0x0000	0x0025	0xF00011204514000B00...	LED1 State	00	Send 00 to turn off, higher values to turn on	Rd Wwr Wr 0x0E
0x0000	0x0026	0x2800	GATT Primary Service Declaration	00:00:00:00:00:00:00:B0:00:40:51:0...	Project Zero Button Service Declaration	
0x0000	0x0027	0x2803	GATT Characteristic Declaration	12:28:00:00:00:00:00:00:00:00:B0:0...		Rd Nfy 0x12
0x0000	0x0028	0xF00011204514000B00...	BUTTON0 State	00	Read or enable notification to get button state	Rd Nfy 0x12
0x0000	0x0029	0x2902	Client Characteristic Configuration	00:00	Write 01:00(Notifications) 02:00(Indications)	
0x0000	0x002A	0x2803	GATT Characteristic Declaration	12:2B:00:00:00:00:00:00:00:00:B0:0...		Rd Nfy 0x12
0x0000	0x002B	0xF00011220451400B00...	BUTTON1 State	00	Read or enable notification to get button state	Rd Nfy 0x12
0x0000	0x002C	0x2902	Client Characteristic Configuration	00:00	Write 01:00(Notifications) 02:00(Indications)	

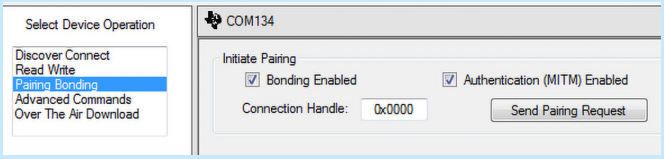
Rysunek 8. Tablica GAT w knie aplikacji BTool po wykonaniu operacji *Read Values*

ConHnd	Handle	Uuid	Uuid Description	Value	Value Description	Properties
0x0000	0x0021	0x2800	GATT Primary Service Declaration	00:00:00:00:00:00:00:B0:00:40:51:0...	Project Zero LED Service Declaration	
0x0000	0x0022	0x2803	GATT Characteristic Declaration	0E:23:00:00:00:00:00:00:00:00:B0:0...		Rd Wwr Wr 0x0E
0x0000	0x0023	0xF00011104514000B00...	LED0 State	01	Send 00 to turn off, higher values to turn on	Rd Wwr Wr 0x0E
0x0000	0x0024	0x2803	GATT Characteristic Declaration	0E:25:00:00:00:00:00:00:00:00:B0:0...		Rd Wwr Wr 0x0E
0x0000	0x0025	0xF00011204514000B00...	LED1 State	00	Send 00 to turn off, higher values to turn on	Rd Wwr Wr 0x0E
0x0000	0x0026	0x2800	GATT Primary Service Declaration	00:00:00:00:00:00:00:B0:00:40:51:0...	Project Zero Button Service Declaration	
0x0000	0x0027	0x2803	GATT Characteristic Declaration	12:28:00:00:00:00:00:00:00:00:B0:0...		Rd Nfy 0x12
0x0000	0x0028	0xF00011204514000B00...	BUTTON0 State	00	Read or enable notification to get button state	Rd Nfy 0x12
0x0000	0x0029	0x2902	Client Characteristic Configuration	00:00	Write 01:00(Notifications) 02:00(Indications)	
0x0000	0x002A	0x2803	GATT Characteristic Declaration	12:2B:00:00:00:00:00:00:00:00:B0:0...		Rd Nfy 0x12
0x0000	0x002B	0xF00011220451400B00...	BUTTON1 State	00	Read or enable notification to get button state	Rd Nfy 0x12
0x0000	0x002C	0x2902	Client Characteristic Configuration	00:00	Write 01:00(Notifications) 02:00(Indications)	

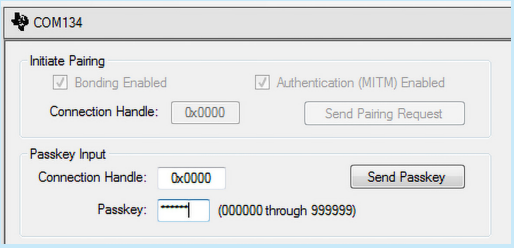
Rysunek 9. Fragment tablicy GATT po zmianie wartości dla serwisu LED0

```
#000029 [ 1129.616 ] INFO: (led_service.c:419) ReadAttrCB : LED0 connHandle: 0 offset: 0 method: 0x0a
#000030 [ 1129.976 ] INFO: (led_service.c:430) ReadAttrCB : LED1 connHandle: 0 offset: 0 method: 0x0a
#000031 [ 1130.471 ] INFO: (button_service.c:473) ReadAttrCB : BUTTON0 connHandle: 0 offset: 0 method: 0x0a
#000032 [ 1130.966 ] INFO: (button_service.c:484) ReadAttrCB : BUTTON1 connHandle: 0 offset: 0 method: 0x0a
#000033 [ 1131.687 ] INFO: (data_service.c:463) ReadAttrCB : String connHandle: 0 offset: 0 method: 0x0a
#000034 [ 1786.533 ] INFO: (led_service.c:495) WriteAttrCB : LED0 connHandle(0) len(1) offset(0) method(0x12)
#000035 [ 1786.533 ] INFO: (project_zero.c:2389) (CB) LED Svc Characteristic value change: paramID(0). Sending msg to app.
#000036 [ 1786.534 ] INFO: (project_zero.c:2060) Value Change msg: LED Service LED0: 01
#000037 [ 1786.534 ] INFO: (project_zero.c:2068) Turning LED0 on
```

Rysunek 10. Informacje aplikacji *ProjectZero* o wykonaniu zmian wartości dla serwisu LED0



Rysunek 11. Wybór operacji parowania



Rysunek 12. Wprowadzanie hasła

i odbieranych od niego. W środkowym oknie znajduje się GUI do sterowania pracą. W dolnym oknie jest pokazywana odczytana tablica GAT.

Skanuj układy BLE

Teraz moduł LaunchPad z aplikacją HostTest pracuje jako Central Device w sieci BLE i jest gotowy na wykrycie układu który rozgłasza („advertising"). Podstawowy tryb pracy z aplikacją BTool jest wykonywany w pięciu krokach (**rysunek 2**).

Krok 1

C6. W polu *Select Device Option* wybierz *Discover Connect*.

Krok 2

C7. W polu Discovery kliknij przycisk *Scan*.

Po dziesięciu sekundach skanowanie jest zatrzymywane. Kiedy rozgłaszające urządzenia zostaną znalezione to log wiadomości pokazuje rozpoznane urządzenia. Przewiń go do góry aż zobaczysz napis ProjectZero. Pokazywany jest adres (linia Address: F0:F8:F2:D5:E3:CD) oraz można odczytać string nazwy (Dump (Rx): ProjectZero) (**rysunek 3**).

Krok 3

C8. W polu *Slave BDA* (**rysunek 4**) rozwiń listę i wybierz adres modułu LaunchPad z aplikacją ProjectZero odczytany w oknie terminala PuTTY do niego dołączonego (rysunek 4).

Krok 4

C9. Kliknij na przycisk *Establish*.

Log wiadomości pokazuje informacje o zestawieniu połączenia z sukcesem (**rysunek 5**).

Krok 5

Po zestawieniu połączenia w lewej kolumnie pokazywane są parametry: adres urządzenia i uchwyt (rysunek 5).

W oknie terminala PuTTy dołączonego do modułu LaunchPad z aplikacją ProjectZero pojawiają się dodatkowe informacje – linie 23-28 (**rysunek 6**).

Pokazywany w linii 26 *Peer address* jest adresem modułu LaunchPad pracującego jako *Central Device*.

Zadanie 4 – Nawiguj po tablicy atrybutów

Teraz można obejrzeć charakterystyki i serwisy dostarczane przez urządzenie z aplikacją Project Zero.

D1. Na liście *Connection Info* kliknij prawym klawiszem myszki na *Handle*.

D2. Teraz kliknij lewym klawiszem myszki na *Discover UUIDs* (**rysunek 7**).

Sekcja poniżej GUI jest wypełniana informacją o serwisach i charakterystykach odczytywaną z modułu LaunchPad z aplikacją ProjectZero.

D3. Na liście *Connection Info* kliknij prawym klawiszem myszki na *Handle*.

D4. Teraz kliknij lewym klawiszem myszki na *Read Values*.

Odczytywana jest cała tablica GAT dostarczana przez aplikację ProjectZero zawierająca własności i zezwolenia charakterystyk (**rysunek 8**).

Interakcja

GATT (Generic Attribute Profile) organizuje dane przechowywane i przesyłane przez BLE i określa format danych przechowywanych na serwerze GATT. Przesyłane atrybuty są w GATT formowane w serwisy i charakterystyki. Każdy serwis może zawierać jedną lub kilka charakterystyk. Z kolei każda charakterystyka zawiera pojedynczą wartość oraz dowolną liczbę deskryptorów opisujących tę wartość. Zestaw serwisów, określających minimalny zakres przypadków użycia danego urządzenia pozwalających wypełnić jego funkcje, tworzy profil urządzenia (szerszy opis w [S7]).

Jedyne informacje przesyłane poprzez łącze radiowe to:

- Uchwyt (Handle) – dynamiczna forma adresowania atrybutu
- Wartość (Value) – zwiera daną
- Typ (Type) – mówi jak interpretować wartość

Każdy wiersz tabeli GATT na rysunku 8 jest atrybutem (Attribute). Typ (Type) narzuca hierarchię w tabeli GATT identyfikowaną kolorami:

- Czerwony – 0x2800 – Deklaracja serwisu – Wartość zawiera UUID dla serwisu
- Żółty – 0x2803 – Deklaracja charakterystyki – Wartość zawiera wartość atrybutu charakterystyki
- Biały – 0xXXXX(-XXX...) – Wartość atrybutu charakterystyki – Wartość zawiera aktualną daną
- Biały – 0x29xx – Opis charakterystyki – Wartość jest informacją o wartość atrybutu charakterystyki

Niektóre atrybuty w tabeli nie posiadają ustawionej wartości. Można odczytać wszystkie wartości lub manualnie odczytywać pojedyncze. Wystarczy kliknąć na pole wartości aby wykonać operację odczytu wartości.

W kolumnie deskryptorów (Value Description) jest informacja jakie akcje są dozwolone. W celu zapisu do charakterystyki trzeba dwukliknąć na pole wartości.

D5. Dwukliknij na pole *Value* dla *LED0 State* (rysunek 8).

D6. W polu *Value* wpisz wartość wyższą niż 00, np.01.

D7. Kliknij na przycisk *Write Value*.

D8. Zamknij okno *Attribute Data Item*.

D9. Zauważ zmianę wartości w polu *Value* dla atrybutu *LED0 State* (**rysunek 9**).

D10. Na płytce modułu LaunchPad z aplikacją ProjectZero powinna zapalić się czerwona dioda LED.

W oknie terminala PuTTy dołączonego do modułu LaunchPad z aplikacją ProjectZero pojawiają się kolejne informacje – linie 29-37 (**rysunek 10**).

W taki sam sposób można pracować z serwisami Button i Data.

Zadanie 5 – Używanie bezpiecznych połączeń z BLE 5

Aplikacja *BTool* pozwala na zastosowanie własności bezpieczeństwa dostarczanych przez BLE 5, takich jak szyfrowanie (encryption), uwierzytelninie (authentication) oraz wiązanie (bonding)

Szyfrowanie połączenia

W celu zastosowania szyfrowania połączenia trzeba zainicjować proces parowania (pairing).

E1. W polu *Select Device Operation* wybierz „*Pairing Bonding*".

E2. W polu „*Initiate Pairing*" zaznacz „*Bonding Enabled*" oraz „*Authentication (MITM) Enabled*".

E3. Kliknij na przycisk „*Send Pairing Request*" (**rysunek 11**).

Spowoduje to wysłanie żądanie sparowania do urządzenia peryferyjnego.

Urządzenie peryferyjne wysyła odpowiedź, co wymaga aby użytkownik wprowadził sześciocyfrowe hasło. Typowo to hasło jest używane przez urządzenie peryferyjne wyposażone w wyświetlacz. Poprzez wyświetlanie tego hasła na wyświetlaczu urządzenia peryferyjnego i wymaganiu aby użytkownik wprowadził go poprzez interfejs urządzenia centralnego połączenie zostaje uwierzytelnione. Czyli zostaje zweryfikowane, że połączenie nie zostało „zhakowane" techniką man-in-the-middle (MITM).

Przesyłanie hasła

Hasło musi zostać wysłane w ciągu 30 sekund po odbiorze przez urządzenie centralne informacji o odbiorze żądania sparowania przez urządzenie peryferyjne. W przeciwnym wypadku proces parowania zostaje uznany za błędny. Wymagane jest wtedy ponowne wysłanie żądania sparowania do urządzenia peryferyjnego.

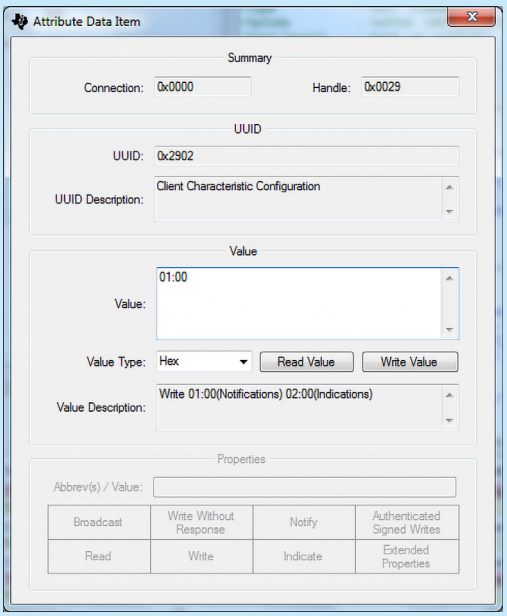
E4. W polu „*Passkey Input*" należy w oknie „*Passkey*" wpisać hasło: '123456'.

E5. Kliknij przycisk „*Send Passkey*" (**rysunek 12**).

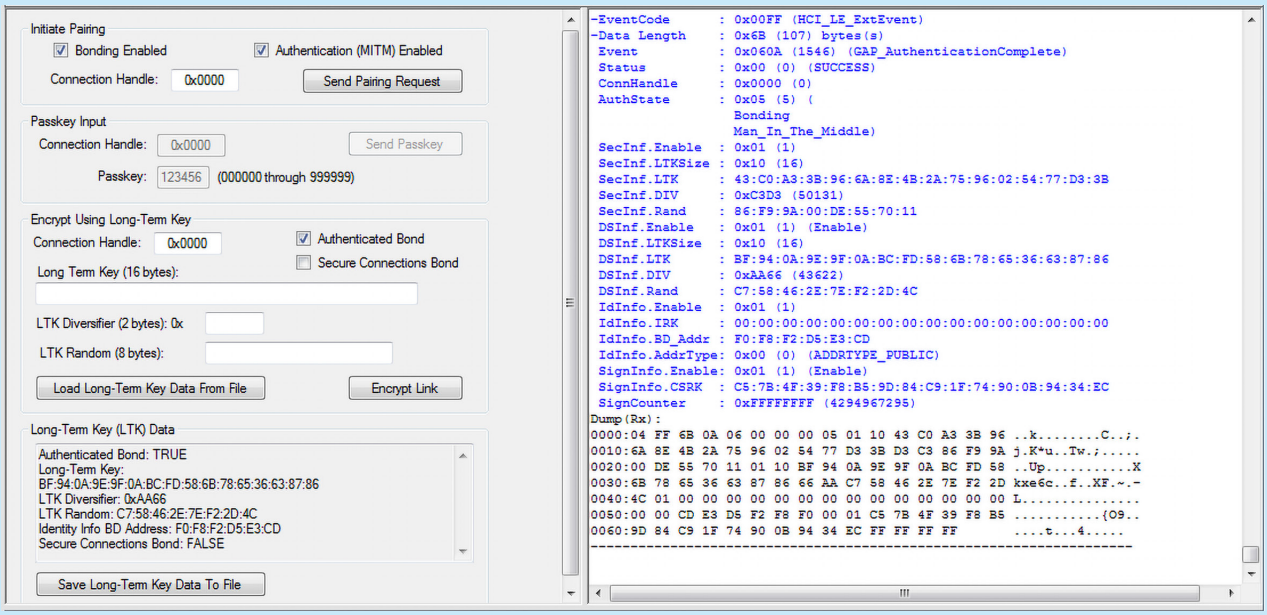
Kiedy parowanie zakończy się sukcesem w oknie loga wiadomości jest wyświetlane zdarzenie „*GAP_AuthenticationComplete*" ze statusem „*Success*" (**rysunek 13**). Połączenie jest teraz szyfrowane. W oknie terminala PuTTy dołączonego do modułu LaunchPad z aplikacją ProjectZero pojawiają się kolejne informacje – linie 38-41 (**rysunek 14**).

Zastosowanie powiadomień

Do sygnalizowania zmiany stanu, np przycisku służy mechanizm notyfikacji. Pozwala to na uniknięcie stosowania przepytywania. GATT serwer, po zmianie wartości, wysyła wiadomość ATT Handle Value Notification. Wcześniej klient musi zasubskrybować powiadomienie.



Rysunek 15. Okno zmiany wartości notyfikacji dla serwisu *BUTTON0*



Rysunek 13. Okno aplikacji BTool po wykonaniu operacji parowania z uwierzytelnianiem

```
#000038 [ 2092.358 ] INFO: (project_zero.c:1472) Pairing started
#000039 [ 2092.358 ] INFO: (project_zero.c:1472) Pairing started
#000040 [ 2092.358 ] INFO: (project_zero.c:1524) BondMgr Requested passcode. We are Displaying passcode 123456
#000041 [ 2100.780 ] INFO: (project_zero.c:1478) Pairing success
```

Rysunek 14. Informacje aplikacji *ProjectZero* po wykonaniu operacji parowania z uwierzytelnianiem

ConHnd	Handle	Uuid	Uuid Description	Value	Value Description	Properties
0x0000	0x0021	0x2800	GATT Primary Service Declaration	00:00:00:00:00:00:00:B0:00:40:51:0...	Project Zero LED Service Declaration	
0x0000	0x0022	0x2803	GATT Characteristic Declaration	0E:23:00:00:00:00:00:00:00:00:B0:0...		Rd Wwr Wr 0x0E
0x0000	0x0023	0xF00011110451400B00...	LED0 State	01	Send 00 to turn off, higher values to turn on	Rd Wwr Wr 0x0E
0x0000	0x0024	0x2803	GATT Characteristic Declaration	0E:25:00:00:00:00:00:00:00:00:B0:0...		Rd Wwr Wr 0x0E
0x0000	0x0025	0xF00011120451400B00...	LED1 State	00	Send 00 to turn off, higher values to turn on	Rd Wwr Wr 0x0E
0x0000	0x0026	0x2800	GATT Primary Service Declaration	00:00:00:00:00:00:00:B0:00:40:51:0...	Project Zero Button Service Declaration	
0x0000	0x0027	0x2803	GATT Characteristic Declaration	12:28:00:00:00:00:00:00:00:00:B0:0...		Rd Nfy 0x12
0x0000	0x0028	0xF00011210451400B00...	BUTTON0 State	00	Read or enable notification to get button state	Rd Nfy 0x12
0x0000	0x0029	0x2902	Client Characteristic Configuration	01:00	Write 01:00(Notifications) 02:00(Indications)	
0x0000	0x002A	0x2803	GATT Characteristic Declaration	12:2B:00:00:00:00:00:00:00:00:B0:0...		Rd Nfy 0x12
0x0000	0x002B	0xF00011220451400B00...	BUTTON1 State	00	Read or enable notification to get button state	Rd Nfy 0x12
0x0000	0x002C	0x2902	Client Characteristic Configuration	00:00	Write 01:00(Notifications) 02:00(Indications)	

Rysunek 16. Fragment tablicy GATT dla serwisu *BUTTON0* po przyciśnięciu przycisku

Dla obsługi przycisków w aplikacji *ProjectZero* możliwa jest notyfikacja ponieważ:

- Własności każdej charakterystyki *BUTTONx State* zawierają znacznik GATT_PROP_NOTIFY
- Każda charakterystyka *BUTTONx State* zawiera atrybut *Client Characteristic Configuration Descriptor* (CCCD). Wpis do niego umożliwia włączenie/wyłączenie notyfikacji.
- Aplikacja, w zależności od ustawienia CCCD, wysyła notyfikację po zmianie stanu wartości charakterystyki.
- Dla charakterystyki *BUTTON0 State* na rysunku 9 w polu własności (Properies) jest widoczna notyfikacja (Nfy). Ma ona również atrybut *Client Characteristic Configuration*. Którego nie ma charakterystyka LED0.

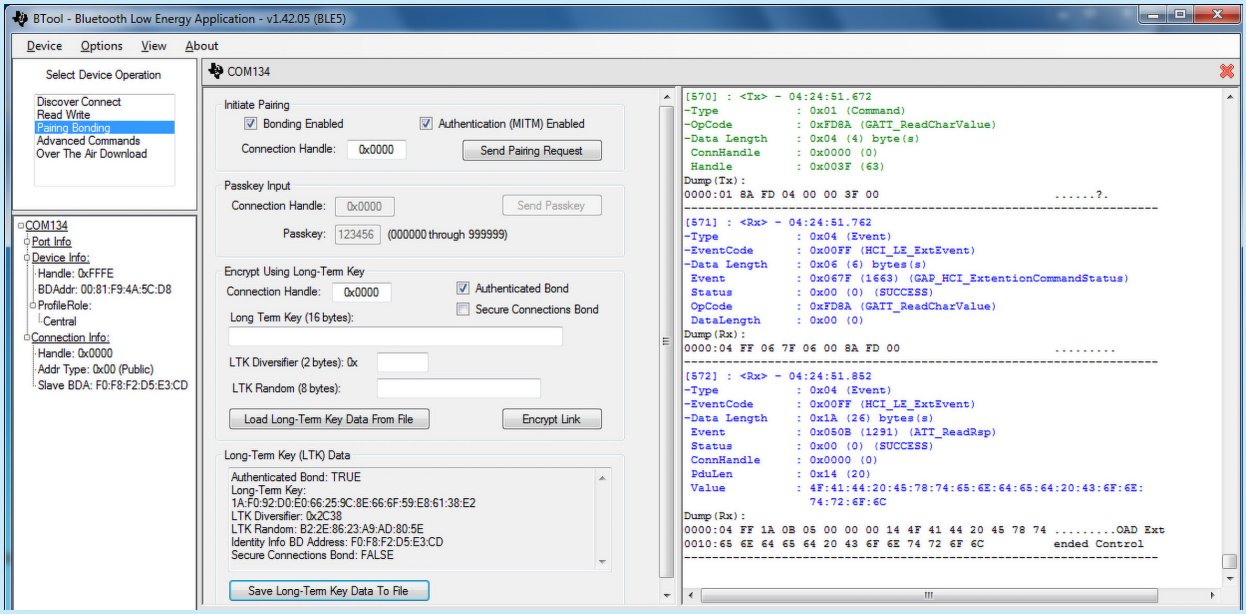
E6. Dla serwisu „*BUTTON0*" dwukliknij na pole *Value* atrybutu *Client Characteristic Configuration*.
E7. W polu *Value* wpisz wartość 01:00 (**rysunek 15**).
E8. Kliknij na przycisk *Write Value*.
E9. Zamknij okno *Attribute Data Item*.
E10. Na płytce modułu LaunchPad z aplikacją ProjectZero przyciśnij i trzymaj przycisk lewy (BUTTON0).
E11. Zauważ zmianę informacji w profilu ProjectZero w linii 0x0028 handle line „BUTTON0 State" (**rysunek 16**).
E12. W oknie terminala PuTTy dołączonego do modułu LaunchPad z aplikacją ProjectZero pojawiają się kolejne informacje – linie 42-44 (**rysunek 17**).

Używanie wiązania i klucza długoterminowego

Wiązanie jest cechą, która umożliwia, po wykonaniu parowania z innym urządzeniem, zapisu specyficznych informacji o tym urządzeniu. Szczególnie, zapisywane jest lokalnie hasło generowane podczas operacji parowania. Jeśli połączenie jest zakończone i ponownie wznowione, to klucze są pobrane z pamięci nieulotnej, bez konieczności wykonywania pełnej procedury parowania. Pamiętane są również ustawienia notyfikacji.

```
#000042 [ 2387.380 ] INFO: (button_service.c:539) WriteAttrCB (CCCD): param: 0 connHandle: 0 - OTA write
#000043 [ 2387.380 ] INFO: (project_zero.c:2459) (CB) Button Svc Char config change paramID(0). Sending msg to app.
#000044 [ 2387.382 ] INFO: (project_zero.c:2129) CCCD Change msg: Button Service BUTTON0: Notifications enabled
```

Rysunek 17. Informacje aplikacji *ProjectZero* po wykonaniu zmian wartości notyfikacji dla serwisu *BUTTON0*



Rysunek 18. Okno aplikacji BTool po ponownym wznowieniu połączenia

Literatura

1. CC1352R (PREVIEW) SimpleLink Multi-Band CC1352R Wireless MCU, ***http://bit.ly/2vJkmqm***
2. Układy scalone z obsługą Bluetooth 5, Henryk A. Kowalski, „Elektronika Praktyczna", 5/2018
3. SimpleLink Multi-Band CC1352R Wireless MCU LaunchPad Development Kit, LAUNCHXL-CC1352R1, ***http://bit.ly/2I0oP99***
4. SimpleLink CC13x2 Software Development Kit, SIMPLELINK-C-C13X2-SDK, Ver. 2.20.00.71, 09-Jul-2018, ***http://bit.ly/2M4Anub***
5. ProjectZero example project for the CC1352R1 LaunchPad , C:/ti/simplelink_cc13x2_sdk_2_20_00_71/examples/rtos/CC1352R1_LAUNCHXL/ble5stack/project_zero/README.html
6. HostTest example project for the CC1352R1 LaunchPad C:/ti/simplelink_cc13x2_sdk_2_20_00_71/examples/rtos/CC1352R1_LAUNCHXL/ble5stack/host_test/README.html
7. SimpleLink Academy 2.20.03 for SimpleLink CC13x2 SDK 2.20, ***http://bit.ly/2vSOzDm***
8. Bluetooth low energy Fundamentals lab, ***http://bit.ly/2RdCS0Y***
9. Category:CCS UniFlash (Version: 4.4.9.1922), ***http://bit.ly/2r8q8xc***
10. TI Resource Explorer (TIREX) , Texas Instruments, ***http://bit.ly/2HHDqqo***
11. BLE 5-Stack User's Guide, ***http://bit.ly/2DGONBF***
12. BTool Guide 5.00.00, ***http://bit.ly/2RdaVWP***
13. CC26x0 SimpleLink Bluetooth® low energy Software Stack 2.2.x Developer's Guide (CC2640/CC2650 Bluetooth low energy Software Developer's Guide), SWRU393E, 14 Mar 2018, ***http://bit.ly/2QeTt2Q***
14. Bluetooth Core Specification Version 5.0, Bluetooth Special Interest Group (SIG), ***http://bit.ly/2DFXnR2***
15. SimpleLink CC1352R Sub-1 GHz + Bluetooth low energy concurrency example, Michelle Tate, February 27, 2018, 09:47, Texas Instruments, ***http://bit.ly/2MB4z1f***

Wybrane pozostałe artykuły kursu „Systemy dla Internetu Rzeczy"

S7 Bluetooth Low Energy, „Elektronika Praktyczna", 6/2017
S12 Oprogramowanie narzędziowe dla układów CC26xx i CC13xx platformy SimpleLink, „Elektronika Praktyczna", 11/2017
S13 Zestaw CC26x2R1 LaunchPad, „Elektronika Praktyczna", 1/2018
S15 Zestaw CC1352R1 LaunchPad, „Elektronika Praktyczna", 5/2018
S17 Jednoczesna komunikacja radiowa z użyciem dwóch protokołów i w dwóch pasmach, 8/2018
S18 Praca z jednoczesną komunikacją radiową z użyciem dwóch protokołów i w dwóch pasmach, 9/2018

Sekcja „*Long-term Key (LTK) Data*” wyświetla nowe informacje, gdy skończy się pomyślenie parowanie z włączonym zezwoleniem na wiązanie (Bonding). Są one pobierane z pakietu danych zdarzenia „*GAP_AuthenticationComplete*”. Dane te są potrzebne w przypadku wznawiania połączenia szyfrowanego.

E13. Kliknij na przycisk „*Save Long-term Key Data to File*”.

E14. Wpisz nazwę pliku, np. „Key1”. Spowoduje to zapis informacji do pliku w formacie tekstowym CSV.

E15. Zapamiętaj (zapisz) ścieżkę w której plik został zapisany.

Aplikacja ProjectZero zawiera menadżera wiązania, który zapisuje długoterminowy klucz generowany w trakcie wiązania. Jest on zapisywany w bezpiecznej pamięci nieulotnej (Flash) układu scalonego CC1352R1. Więcej informacji o menadżerze wiązania jest zamieszczone w dokumencie „Bluetooth low energy Software Developer's Guide” (SWRU393) [13].

Weryfikacja

W celu zweryfikowania działania wiązania zamknij połączenie.

E16. W oknie aplikacji BTool w polu „*Select Device Operation*” kliknij na pozycję „*Discover/Connect*”.

E17. Kliknij na przycisk „*Terminate*”.

Spowoduje to odłączenie urządzenia z aplikacją ProjectZero. W logu wiadomości zostanie wyświetlona informacja o zdarzeniu „*GAP_TerminateLink*” ze statusem „*Success*”. Także w lewej kolumnie zniknie informacja o połączeniu.

E18. Wykonaj ponowne połączenie zgodnie z wcześniejszym postępowaniem w krokach 1-5.

E19. W polu *Select Device Operation* wybierz „*Pairing Bonding*” (**rysunek 18**).

E20. Kliknij na przycisk „*Load Long-Term Key Data From File*”.

E21. Wybierz plik wcześniej zapisany (np. Key1.txt).

E22. W oknie *Select Connection* kliknij *OK*. Pola danych zostaną wypełnione informacją odczytaną z pliku.

E23. Kliknij na przycisk „*Encrypt Link*”.

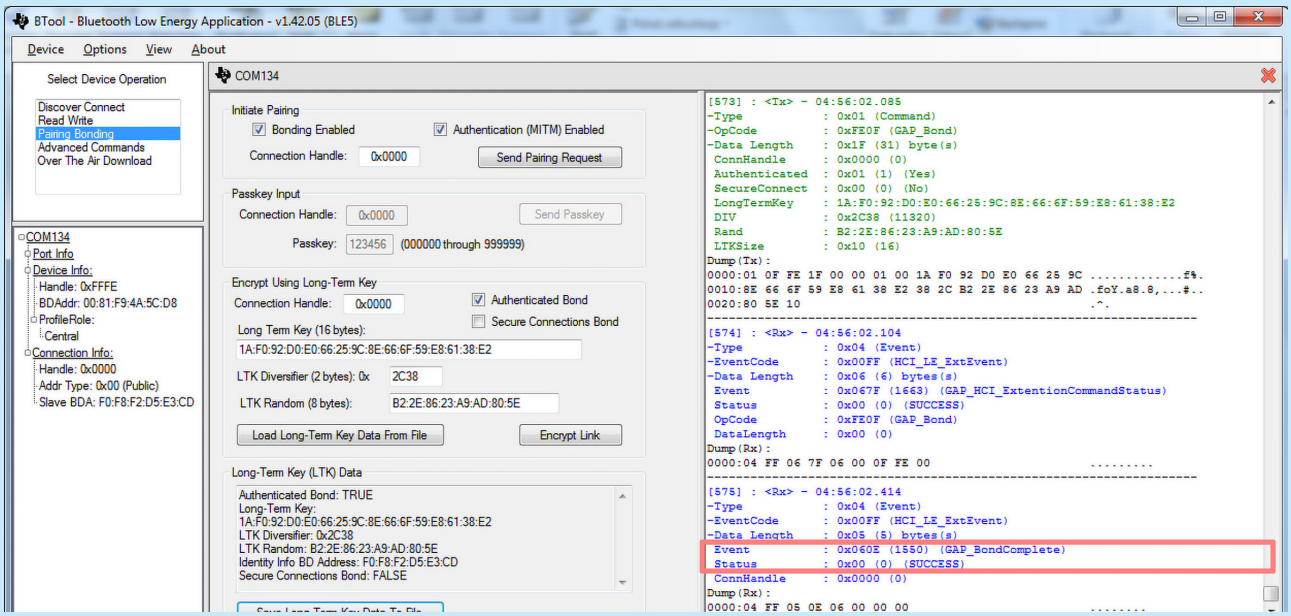
Spowoduje to ponowną inicjalizację szyfrowania oraz notyfikacji. Log zawiera informację o zdarzeniu „GAP_BondComplete” z informacją „Success” (**rysunek 19**). W oknie terminala PuTTy dołączonego do modułu LaunchPad z aplikacją ProjectZero pojawiają się kolejne informacje – linie 45-61 (**rysunek 20**).

Ponowne wznowienie połączenia ze wznowieniem szyfrowania zakończyło się sukcesem: Encription success.

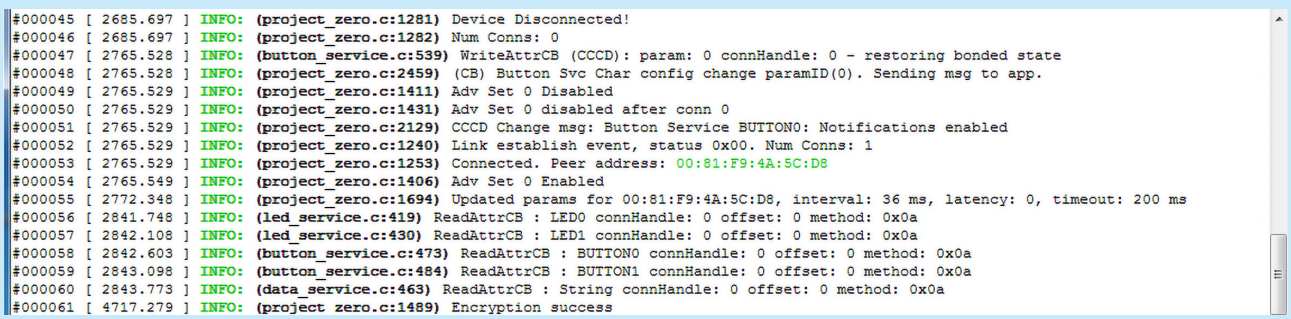
Podsumowanie

Przykład wykorzystania opisanej techniki bezpieczeństwa jest pokazany w prezentacji „SimpleLink CC1352R Sub-1 GHz + Bluetooth low energy concurrency example” [15]. W tym przypadku została zdefiniowana sieć SimpleLink w pasmie 868 MHz. W węzłach końcowych (czujnikowych) w tej sieci zostały zastosowane procesory dwupasmowe CC1352R1. Na nich zostało uruchomione oprogramowanie do jednoczesnej obsługi sieci SimmpleLink oraz drugiej sieci z obsługą pełnego standardu BLE 5. W poprzednim odcinku kursu „Praca z jednoczesną komunikacją radiową z użyciem dwóch protokołów i w dwóch pasmach” [S18] jest to dokładnie opisane. W tej prezentacji zostało dodatkowo pokazane, jak, używając smartfona z obsługą standardu BLE, można, po wykonaniu uwierzytelnienia dostępu do węzła dwupasmowego, zmieniać istotne parametry pracy sieci SimpleLink. Omówione są też możliwości zastosowania takiego rozwiązania w automatyce domowej, np. w obsłudze drzwi hotelowych.

Henryk A. Kowalski
Instytut Informatyki
Politechnika Warszawska
kowalski@ii.pw.edu.pl



Rysunek 19. Okno aplikacji BTool po wznowieniu szyfrowania połączenia



Rysunek 20. Informacje aplikacji *ProjectZero* po wznowieniu szyfrowania połączenia