Bluetooth-garagedeurbediening met geringe latentie

met korte BLE-berichten vanaf een smartphone

Stephan Lück (Duitsland)

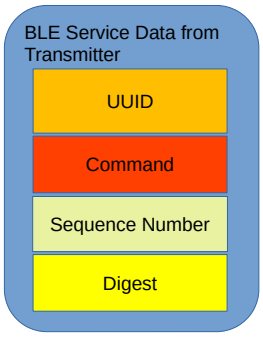
Vaak wil ik de garagedeur openen, maar heb de 433MHz afstandsbediening vergeten. Daarom heb ik een kleine Bluetooth Low Energy-ontvanger (BLE) gebouwd die wordt bediend met een Android-app op mijn smartphone.

Een bijzonderheid van dit project is dat BLE advertising Protocol Data Units (PDU's) worden gebruikt, waardoor vertraging bij het opzetten van de verbinding wordt vermeden. Het systeem reageert dus even snel op een commando van een afstandsbediening als een klassieke hardware-zender/ontvangercombinatie zou doen. Er is uitsluitend éénrichtingscommunicatie, van de smartphone naar de ontvanger, en die is beveiligd met HMAC [1]. Het idee was om alles eenvoudig te houden, zowel de hardware als de software.
Ik deel dit project in de hoop dat u het nuttig vindt of als basis kunt gebruiken voor soortgelijke projecten. De broncode is beschikbaar op GitHub [2].

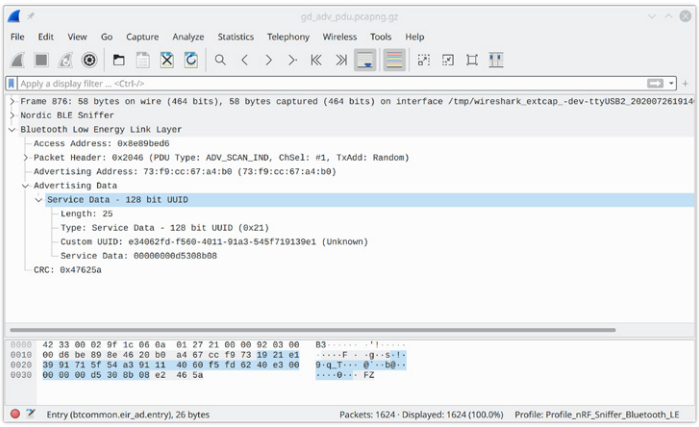
Het protocol

Voor het verzenden van een opdracht worden BLE advertising PDU's gebruikt, die worden beschreven in Volume 6, Part B, Section 2.3 van de Bluetooth-specificatie [3]. Om compatibiliteitsredenen worden de korte (legacy) advertising PDU's gebruikt die een *AdvData*-veld van ten hoogste 31 bytes bevatten. Het *AdvData*-veld bestaat uit een opeenvolging van AD-elementen (Advertising Data) [4] (zie Volume 3, Part C, Section 11 van de Bluetooth-specificatie). Een mogelijk AD-element is het *Service Data* AD-element (zie Bluetooth Core Specification Supplement, Part A, Section 1). Het *Service Data* AD-element bevat een *Service UUID* gevolgd door een willekeurig aantal bytes.
Deze *Service Data* AD-structuur wordt gebruikt om een afstandsbedienings-commando te verzenden. De *Service Universal Unique Identifier* (*UUID*) van 128 bits correspondeert met een unieke zender-ID. Met andere woorden, elke zender definieert een specifieke BLE-service. De na de UUID volgende data zijn volgt gestructureerd, en zijn vereenvoudigd voorgesteld in **figuur 1**:

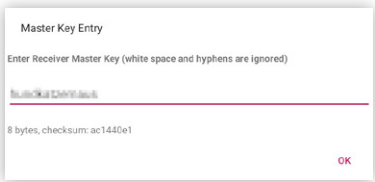
```
typedef struct {
uint8_t cmd; /* command (currently
always 0) */
uint8_t seq_no[3]; /* sequence number
(big endian) */
uint8_t digest[4]; /* first four
octets of HMAC-SHA256 */
} gd_message_t;
```



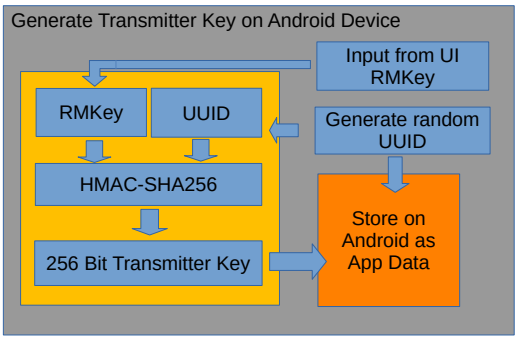
Figuur 1. Vereenvoudigde datastructuur.



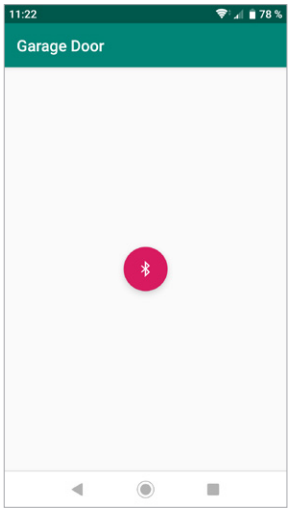
Figuur 2. PDU in Wireshark.



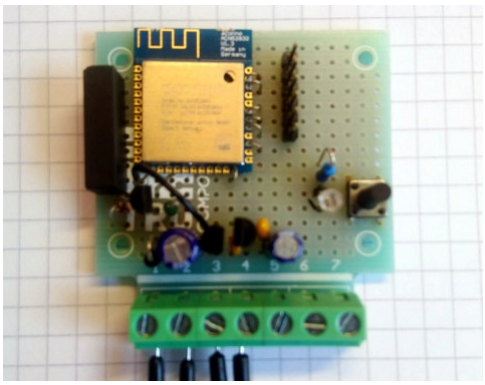
Figuur 3. Start van de app, en genereren van de sleutel.



Figuur 4. Stroomdiagram voor de berekening van de zendersleutel.



Figuur 5. De app is klaar voor gebruik.



Figuur 6. Het opgebouwde prototype.

De "digest" authenticeert het bericht. Het volgnummer maakt het mogelijk duplicaten van PDU's te detecteren en voorkomt replay-attacks. Aangezien BLE advertising PDU's niet versleuteld zijn, is het mogelijk dat anderen zowel de UUID als de erop volgende data ontvangen.
De digest wordt berekend op basis van een zendersleutel. De zendersleutel wordt afgeleid van een hoofdsleutel voor de ontvanger (RMkey) die in de ontvanger is opgeslagen, op basis van de UUID van de zender.

```
transmitter_key = HMAC-SHA256(RMkey,
transmitter_UUID)
```

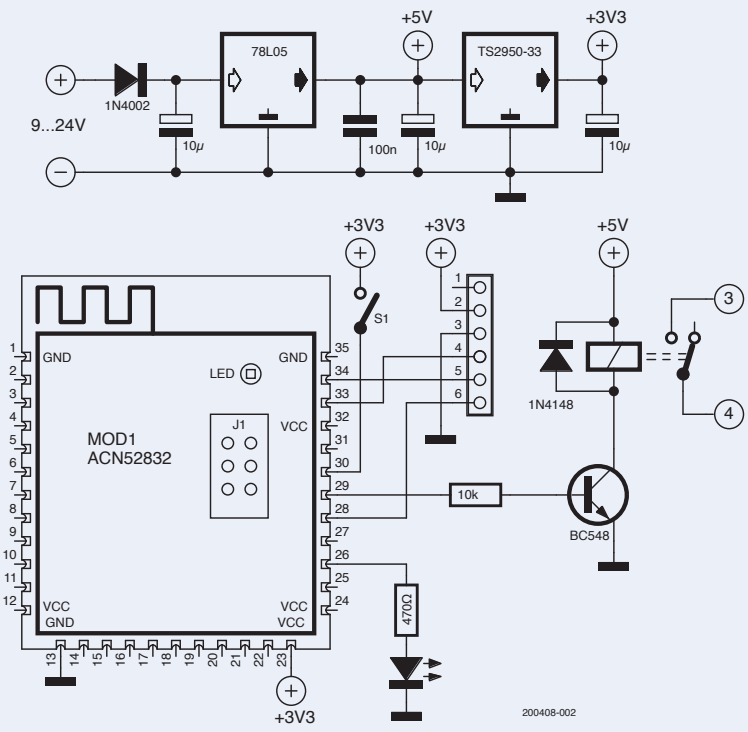
Een voorbeeld van een advertising PDU is te zien in het Wireshark-screenshot (**figuur 2**).

De zender

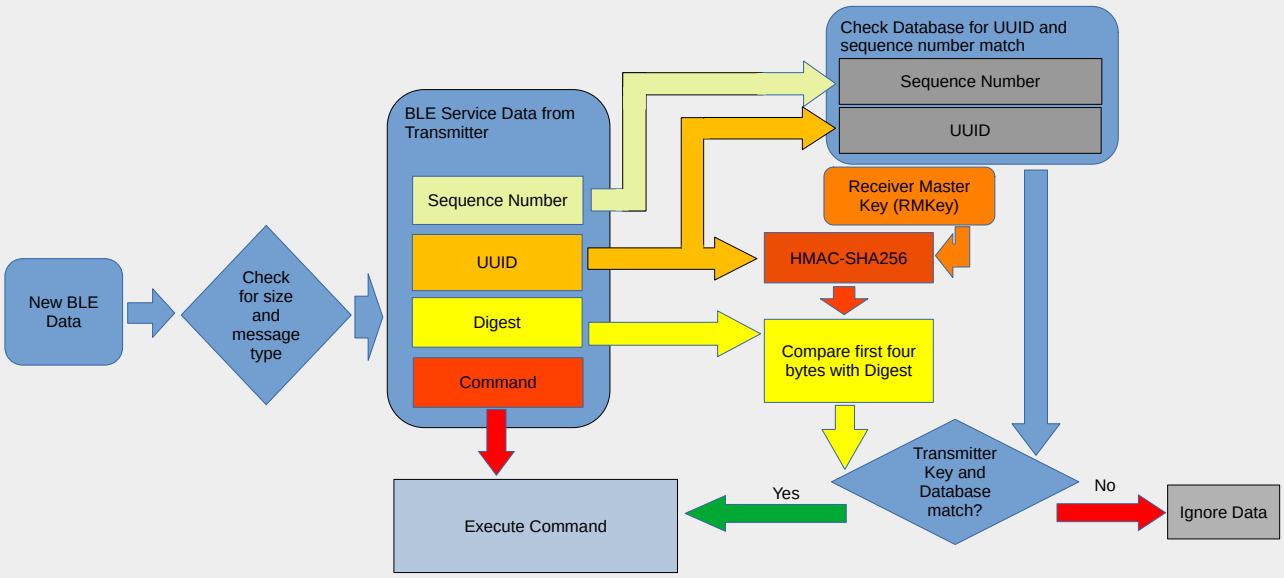
De zender is een eenvoudige Android-app die API niveau 21 vereist (overeenkomend met Android 5 "Lollipop"). Bij de eerste keer opstarten berekent de app een willekeurige UUID en toont een setup-dialoog om de RMkey in te voeren als een base32 gecodeerde string (zie **figuur 3**). Het setup-venster toont de lengte van de sleutel en een CRC32-checksum om de gebruiker te helpen typefouten te voorkomen bij het invoeren van de sleutel. De app berekent de zendersleutel uit de UUID en de RMkey en gooit dan de RMkey weg. De tuple (UUID, zendersleutel), die overeenkomt met een zender-identiteit, wordt dan blijvend opgeslagen. De zender-identiteit gaat verloren wanneer de app wordt gewist of wanneer applicatiegegevens handmatig worden gewist.
De berekeningswijze is te zien in **figuur 4**.
Na de setup toont de app een rode knop (zie **figuur 5**). Wanneer de knop wordt aangeklikt, wordt een BLE advertising procedure geconfigureerd om de hierboven beschreven *Service Data* AD-structuur meerdere malen uit te zenden gedurende een periode van enkele seconden.

De ontvanger

De ontvanger is op gaatjesprint opgebouwd op een prototyping board rond de aconno ACN52832 BLE-module [5] (**figuur 6**). Deze module bevat een Nordic Semiconductor NRF52382-microcontroller, een drukknop, een LED en een relais dat moet worden aangesloten op de motor van de garagedeur. De ontvanger heeft een 9...24 VDC-voeding nodig. Eventueel kan de voedingsspanning van de garagedeur-aandrijving worden afgetakt, maar anders kan ook een eenvoudige netadapter worden gebruikt. Het schema is te zien in **figuur 7**.
Bij het bouwen van de software voor de microcontroller, maakt een Python-script het bestand *rxm_key.bin* aan dat de 20 byte grote



Figuur 7. Schema van de ontvanger.



Figuur 8. Stroomdiagram voor het ontvangen van BLE-data.

RMkey bevat. Bovendien wordt een tekstuele representatie van de sleutel gegenereerd en opgeslagen in *_build/rxm_key.txt*. Dit laatste bevat de informatie die moet worden ingevoerd in de zender-app bij de eerste start. De ontvanger houdt een database bij van UUID's en volgnummers van zenders. Om een nieuwe zender toe te voegen, moet de knop worden ingedrukt en vervolgens moet de toe te voegen zender worden geactiveerd. Wanneer de knop langer dan vijf seconden wordt ingedrukt, wordt de database gewist en zal de ontvanger op geen enkele zender meer reageren.

Wanneer de ontvanger een advertising PDU ontvangt die de hierboven beschreven *Service Data* AD-structuur bevat, controleert hij de "digest" of hij de UUID van de zender kent en of het volgnummer juist is (**figuur 8**). Indien alle controles slagen, wordt het relais gedurende één seconde geactiveerd.

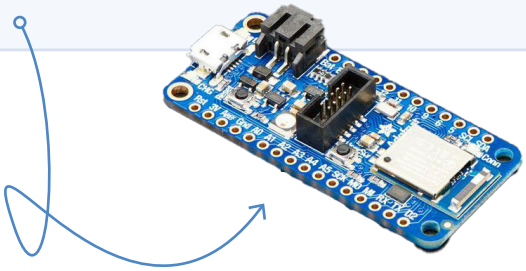
200408-01

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Neem contact op met de auteur via GitHub [2] of stuur een e-mail naar de redactie van Elektor via redactie@elektor.com.

GERELATEERDE PRODUCTEN

- **Adafruit CLUE – nRF52840 Express with Bluetooth LE (SKU 19512)**
  www.elektor.nl/19512
- **Dogan Ibrahim, Android App Development for Electronics Designers (SKU 18687)**
  www.elektor.nl/18687
- **makerdiary nRF52840 MDK USB Dongle with Case (SKU 19252)**
  www.elektor.nl/19252
- **Adafruit Feather nRF52840 Express (SKU 20114)**
  www.elektor.nl/20114



WEBLINKS

[1] HMAC (Wikipedia): https://en.wikipedia.org/wiki/HMAC
[2] GitHub-repository: https://github.com/kiffie/ble-garage-door
[3] Bluetooth SIG: www.bluetooth.com/
[4] Bluetooth Advertising Data Basics, Silicon Labs: https://bit.ly/silabs-bluetooth-ad
[5] ACN52832 van aconno: https://aconno.de/products/acn52832/