LORAWAN PROJECT

LEDOUX Alexis – MARCHAL Baptiste –
MEROLLA Mathis – NIVESSE Charles

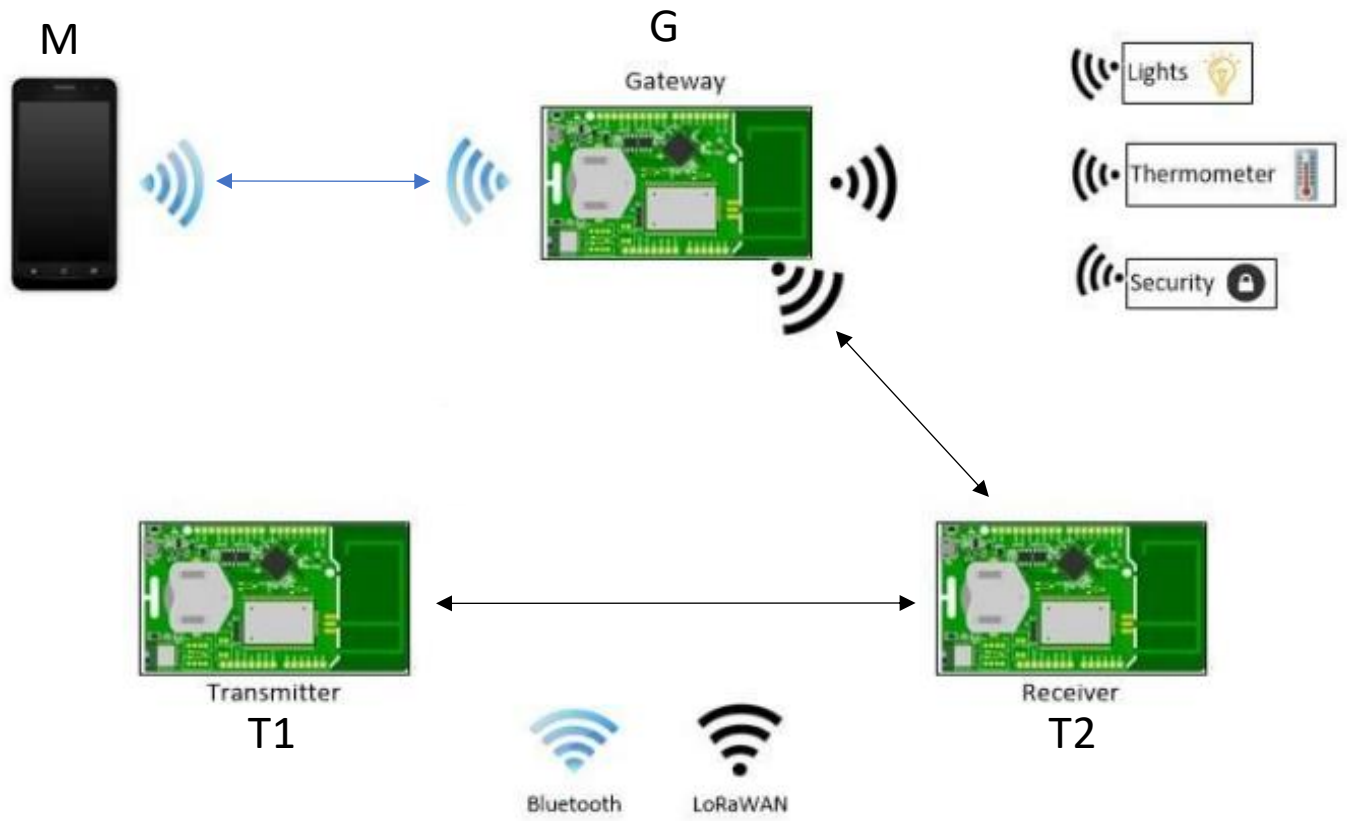
IOS 2

Sommaire

INTRODUCTION

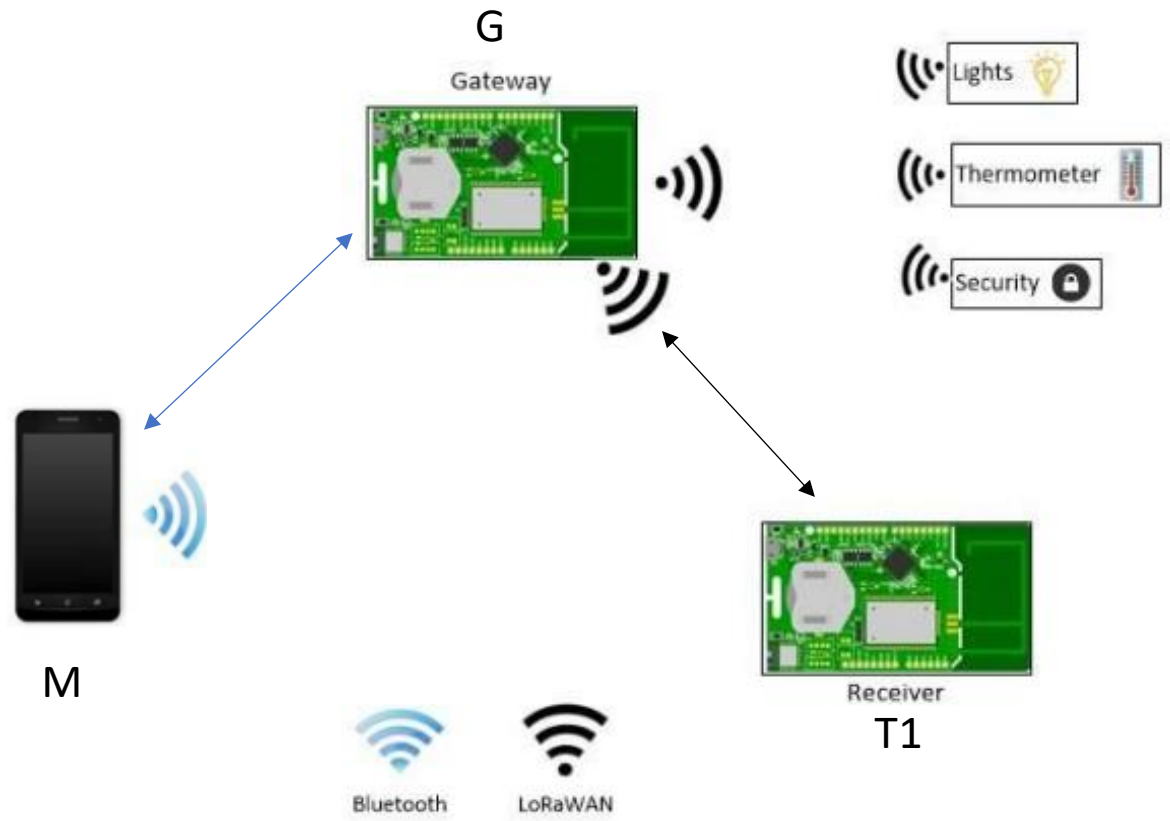
Scenario 1

- Button used on APP M
- G ask T2
- T2 ask T1
- T1 send θ1 to T2
- T2 send θ1 & θ2 to G
- G send θ1 & θ2 to the APP



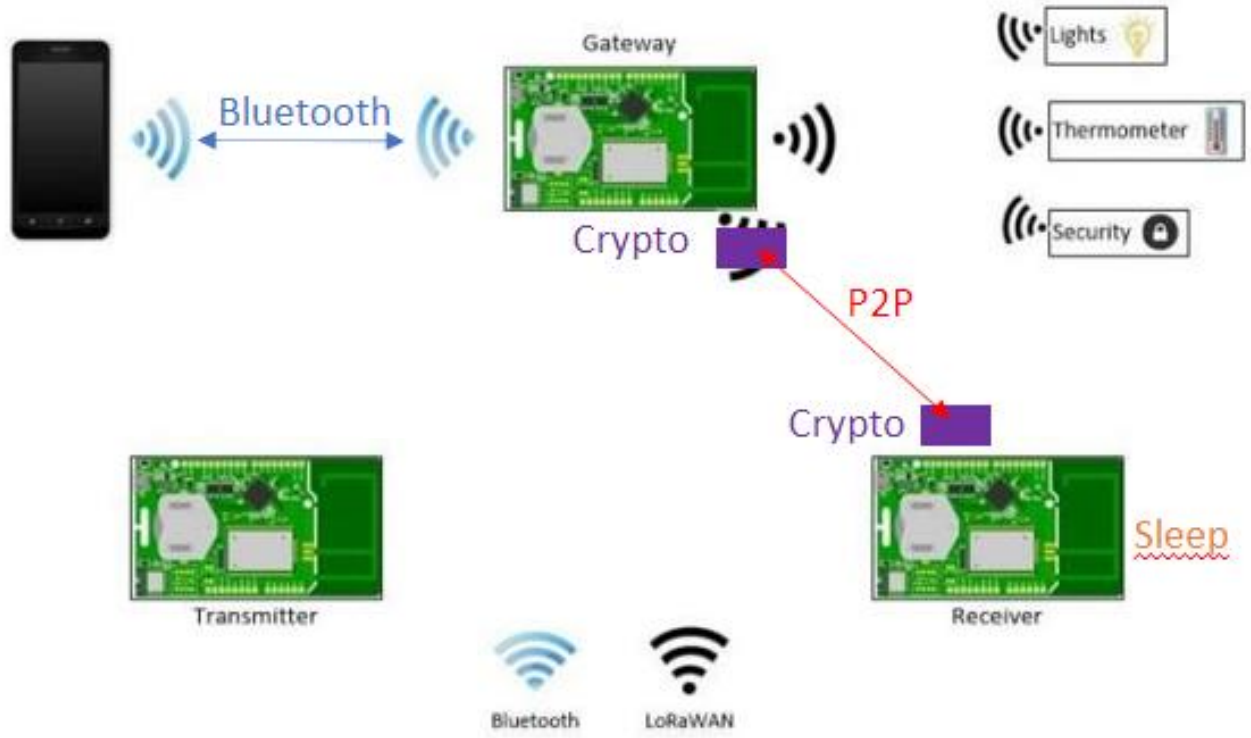
Scenario 2

- Button APP M
- G ask T1
- T1 send θ1 to G
- G send θ1 to the APP



List of locks

- Bluetooth
- Crypto
- P2P
- Sleep



SOLUTIONS

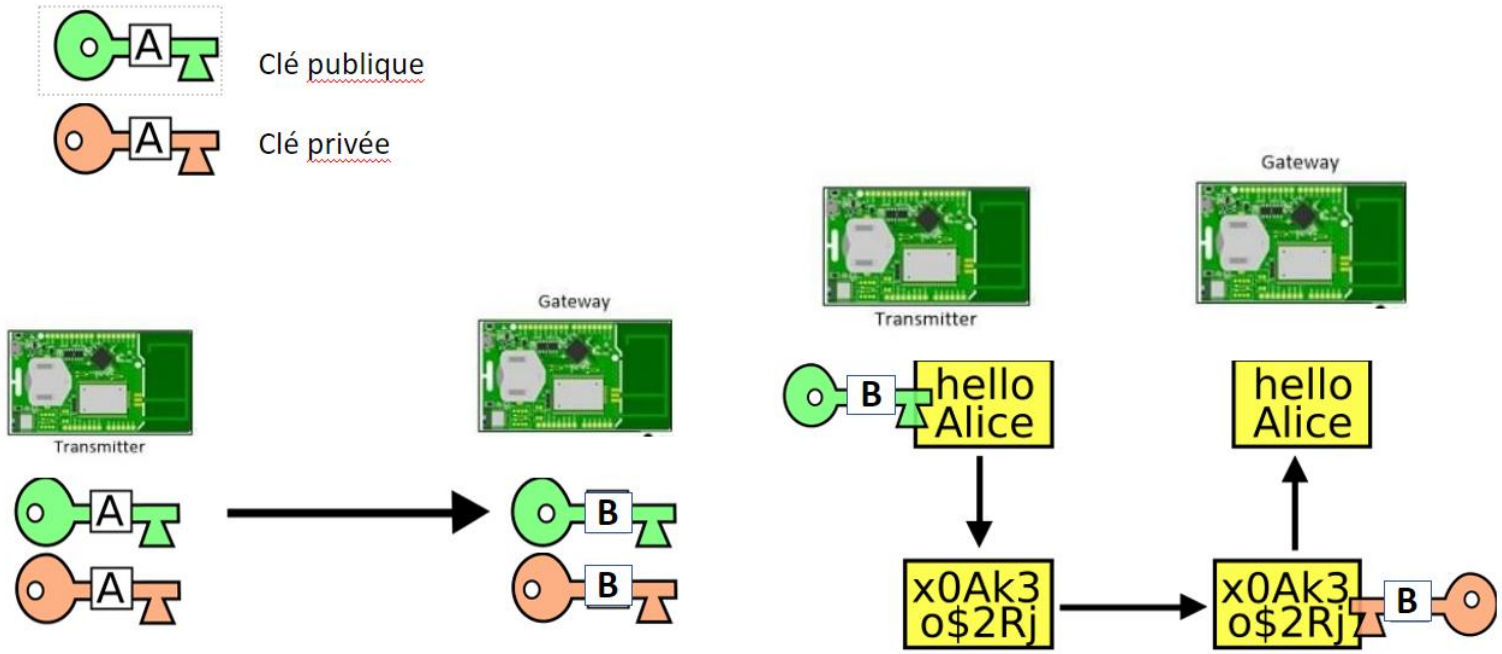
P2P - Code


```
// Configuring the RN2483 for P2P
void LoraP2P_Setup()
{

  Serial2.print("sys reset\r\n");
  delay(200);
  Serial2.print("radio set pwr ");
  Serial2.print(trPower);
  Serial2.print("\r\n");
  delay(100);
  Serial2.print("radio set sf ");
  Serial2.print(SprFactor);
  Serial2.print("\r\n");
  delay(100);
  Serial2.print("radio set wdt ");
  Serial2.print(readDelay);
  Serial2.print("\r\n");
  delay(100);
  Serial2.print("mac pause\r\n");
  delay(100);

  FlushSerialBufferIn();
}
```

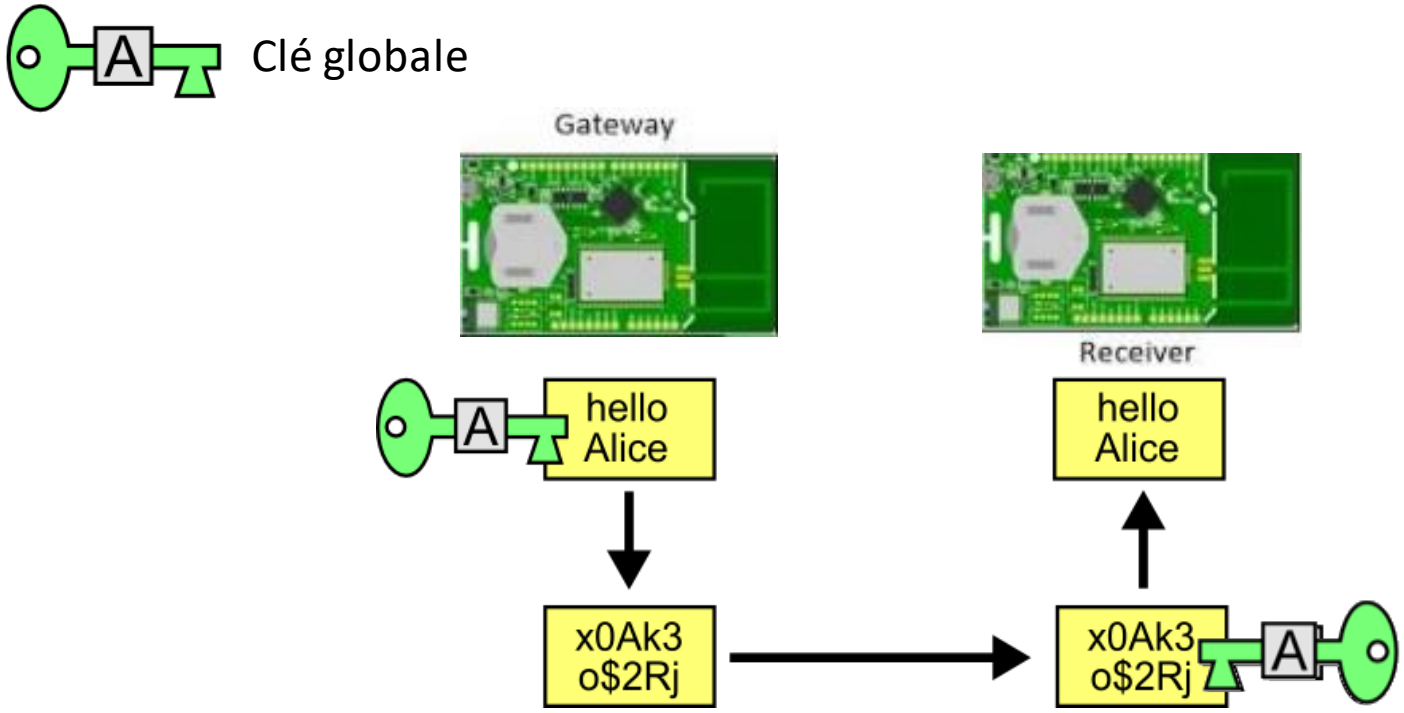
```
// Send Data array (in HEX)
void LORA_Write(char* Data)
{
  Serial2.print("radio tx ");
  Serial2.print(Data);
  Serial2.print("\r\n");
  Serial2.flush();

  waitTillMessageGone();

}
```

```
int LORA_Read(char* Data)
{
  int messageFlag = 0;
  String dataStr = "radio_rx  ";
  String errorStr = "radio_err";
  String Buffer = "";
  StartLoraRead();
  while (messageFlag == 0) // As long as there is no message
  {
    while (!Serial2.available());
    delay(50);  // Some time for the buffer to fill
    // Read message from RN2483 LORA chip
    while (Serial2.available() > 0 && Serial2.peek() != LF)
    {
      Buffer += (char)Serial2.read();
    }
    // If there is an incoming message
    if (Buffer.startsWith(dataStr, 0)) // if there is a message in the buffer
    {
      int i = 10;  // Incoming data starts at the 11th character
      // Seperate message from string till end of datastring
      while (Buffer[i] != CR && i - 10 < max_dataSize)
      {
        Data[i - 10] = Buffer[i];
        i++;
      }
      messageFlag = 1; // Message received
    }
    else if (Buffer.startsWith(errorStr, 0))
    {
      messageFlag = 2; // Read error or Watchdogtimer timeout
    }
  }

#ifdef DEBUG
  SerialUSB.println(Buffer);
#endif
  return (messageFlag);
}
```

Crypto – Theory : Cryptographie asymétrique



Crypto – Theory : Cryptographie symétrique



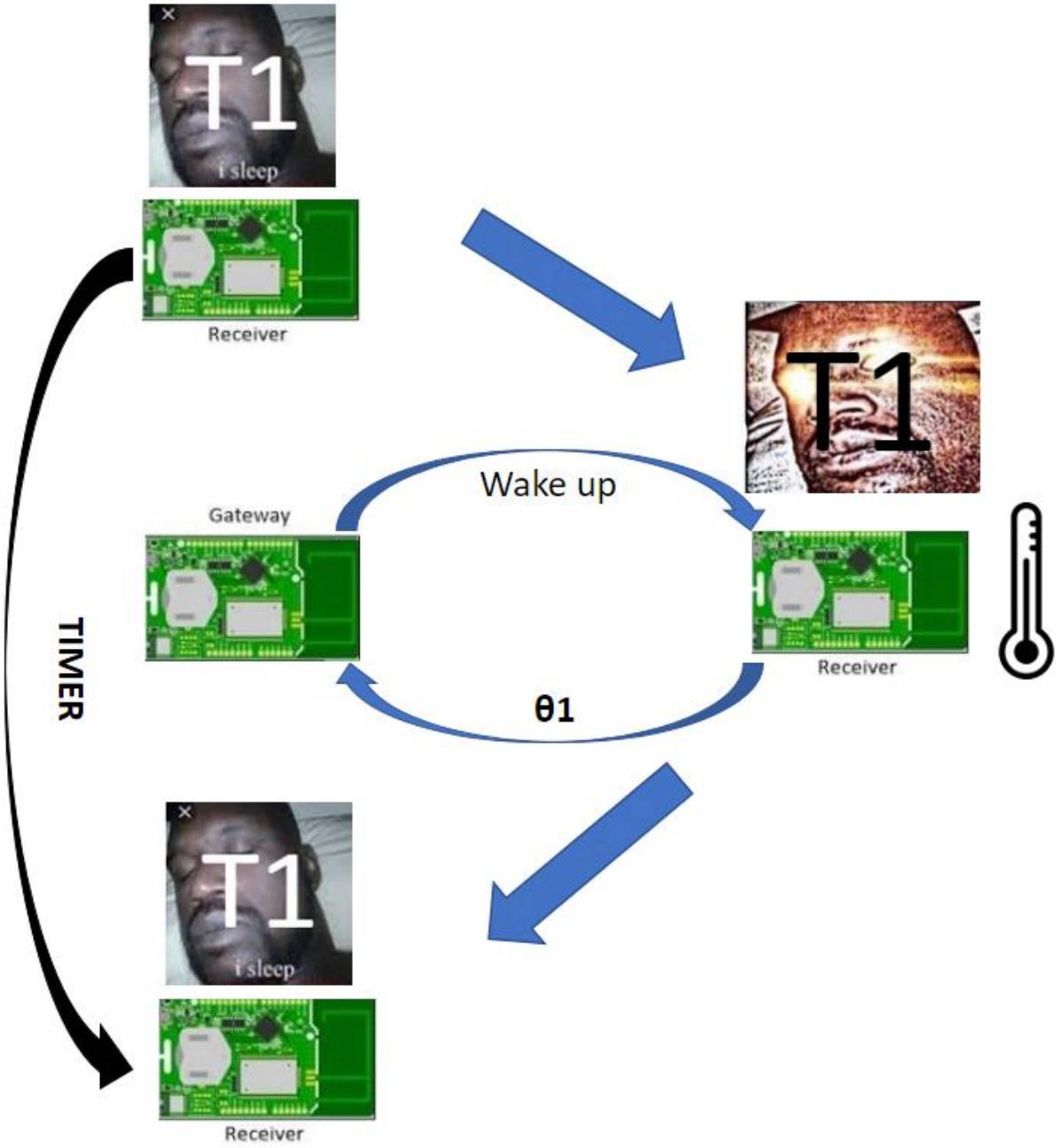
Crypto - Code : Cryptographie symétrique AES

- Librarie : #include <AES.h>
- Key_setting : clef de cryptage

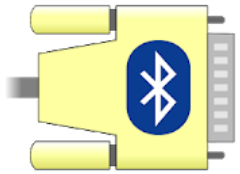
```
void key_setting()
{
  aes.set_key (key, 256);
}
void cryptage ()
{
  aes.encrypt (plain, cipher);
}
void decryptage()
{
  aes.decrypt (cipher, check);
}
void affichage()
{
  for (byte ph = 0 ; ph < 3 ; ph++)
  {
    Serial.println(ph);
    for (byte i = 0 ; i < sizeof(plain)/sizeof(plain[0]) ; i++)
    {
      byte val = ph == 0 ? plain[i] : ph == 1 ? cipher[i] : ph == 2 ? check[i] : 0;
      Serial.print (val >> 4, HEX) ; Serial.print (val & 15, HEX) ; Serial.print (" ")
    }
    Serial.println () ;
  }
}
```

Sleep - Theory

- G awake T1
- T1 send θ1 to G
- T1 sleep
- G send θ1 to the APP



Bluetooth - Theory



Serial Bluetooth Terminal

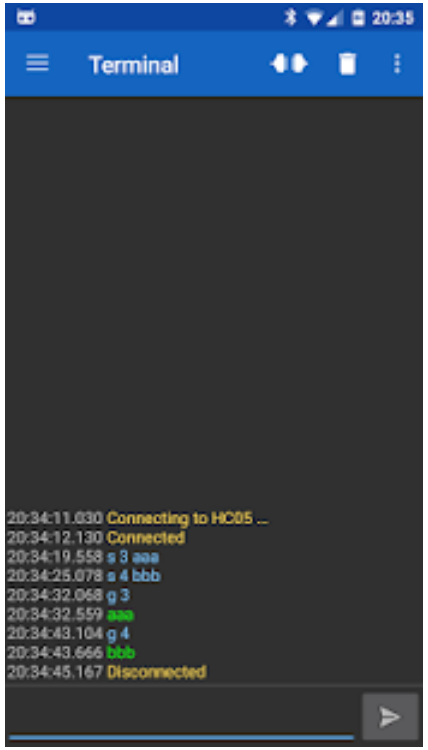
Kai Morich Outils ★★★★★ 1 747

PEGI 3

Achats via l'application proposés

Cette application est disponible pour votre appareil

Ajouter à la liste de souhaits Installer



Bluetooth - Code

```
// Initialize the BLE hardware
rn487xBle.hwInit() ;
// Open the communication pipe with the BLE module
bleSerial.begin(rn487xBle.getDefaultBaudRate()) ;
// Assign the BLE serial port to the BLE library
rn487xBle.initBleStream(&bleSerial) ;
// Finalize the init. process
rn487xBle.swInit();

// Fist, enter into command mode
rn487xBle.enterCommandMode() ;
// Stop advertising before starting the demo
rn487xBle.stopAdvertising() ;
// Set the advertising output power (range: min = 5, max = 0)
rn487xBle.setAdvPower(3) ;
// Set the serialized device name, i.e. device n,ame + 2 last bytes from MAC address.
rn487xBle.setSerializedName(myDeviceName) ;
rn487xBle.clearAllServices() ;
rn487xBle.reboot() ;
rn487xBle.enterCommandMode() ;
// Set a private service ...
rn487xBle.setServiceUUID(myPrivateServiceUUID) ;
```

● ● ●

```
void loop() {
  if (rn487xBle.getConnectionStatus())
  {
```

● ● ●

```
aes.decrypt (out, check);

rn487xBle.writeLocalCharacteristic(temperatureHandle, check)
```