BLE SECURITY Cryptography Options

Abstract—*This paper provides an overview of security threats for a Bluetooth low energy (BLE) enabled embedded device and best practices to address risks. Key elements of secure communication are authentication, key establishment, and message encryption. There are several security functions provided by embedded processors that can be utilized for efficient implementation of security.*

I. INTRODUCTION

Security of an embedded application is addressed by protecting the major attack vectors susceptible to the device. The highest risk attack would be an imposter mobile app taking control of the embedded device and corrupting or activating it. A secondary risk is knock-off device imitating the original device. The most significant attack vector is an over the air Bluetooth communication. A best practice is to not store private keys on the device. The device should randomly generate a private security key and then perform a secure key exchange to the mobile application. Once a shared private key has been established a symmetrical encrypted standard can be used to secure Bluetooth messages between devices.

Several embedded microcontroller unit (MCU) vendors provide security capabilities within their firmware. For the purpose of this paper an STM32 ARM M4 is used as an example. The ARM M4 provides hardware security capability for random number generation and AES encryption that efficiently enables mobile security best practices.

CALL US
360.693.2600

US HEADQUARTERS
1714 Broadway, Vancouver, WA 98663

WEBSITE
www.sigmadzn.com

SIGMADESIGN

An ideal interface for the IOT is Bluetooth Low Energy (BLE) Established in 2010 as part of the Bluetooth 4.0 standard. BLE devices with integrated radios use minimal power and are available for as little as $2.00 [3]. This makes BLE advantageous over competing protocols such as Z-Wave and ZigBee. BLE uses the industry standard Advanced Encryption Standard (AES) encryption protocol but "Bluetooth: With Low Energy Comes Low Security", a whitepaper by Researcher Mike Ryan [2] describes the security hole. BLE divides its 2.4 GHz spectrum into 40 channels, 3 are used to establish connections and the remaining 37 are used for data transmission. BLE complicates eavesdropping by periodically jumping channels but a third party eavesdropper can capture the establishment, follow the channel jumps and use brute force to break the initial key exchange since there are only 100,000 combinations to try. Once the initial key is broken the symmetric AES key can be discovered and the eavesdropper can listen to the entire conversation. To be secure an independent layer must be added on top of BLE for authentication, key establishment, and encryption.

II. AUTHENTICATION

It is possible for an attack to imitate an embedded device or imitate the mobile application. Most important is preventing an imitation mobile application from taking control of the device. This is best accomplished by a certificate.

In cryptography, a public key certificate (also known as a digital certificate or identity certificate) is an electronic document used to prove the ownership of a public key. These can be unmanaged or managed certificates issued by a certification authority (CA). A simple authentication would be for the mobile application to send the device a 128byte number that is compared with a version stored in its ROM. One method that bypasses the security risk of having a key stored in ROM is an ECDSA (Elliptic Curve Digital Signature Algorithm). The best choice for authentication depends on several factors and is a topic that needs future investigation.

III. ENCRYPTION

The Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST). This secure encryption is supported by a ST's STM32 hardware-acceleration and library (X-CUBE-CRYPTOLIB).

The library software has an export classified by US Bureau of Industry and Security as ECCN 5D002. Many of the library algorithm implementations (including AES-3971) are validated by the US Cryptographic Algorithm Validation Program (CAVP).

CALL US
360.693.2600

US HEADQUARTERS
1714 Broadway, Vancouver, WA 98663

WEBSITE
www.sigmadzn.com

SIGMADESIGN

The AES library supports AES-128, AES-192, AES-256 bit with the following modes: ECB, CBC, CTR, CFB, OFB, CCM, GCM, CMAC, KEY WRAP and XTS.

The AES cipher algorithm can process data blocks of 128 bit, using a key length of 128, 192 or 256 bit. The STM32 Crypto Library User Manual UM1924 details the interface to the AES library methods. The library utilizes the STM32F4 hardware acceleration which includes a DMA interface.

AES is based on a symmetrical key encryption algorithm. Both parties use a common shared private key to encrypt or decrypt messages. The mobile application also implements AES using the private key established with the device.

IV. KEY ESTABLISHMENT

AA simple method for implementing a private key is to store a hard-coded private key in the device's ROM. This simple implementation has security risks. Once the key is discovered it can be used to learn the communication protocol and impersonate the application.

A secure key would not be hardcoded on the device and based on a random number. The Diffie Hellman (DH) key exchange is a secure algorithm used to establish a shared secret between two parties. The keys are randomly generated so even if a key is discovered it cannot be used for a different session. DH is primarily used as a method of exchanging cryptography keys in symmetric encryption algorithms like AES. The U.S. National Institute of Standards and Technology (NIST) has endorsed elliptic curve cryptography in its Suite B set of recommended algorithms, specifically elliptic curve Diffie–Hellman (ECDH).

The ECDH provides several advantages over RSA algorithm for encryption [4]. With an equivalent level of security the ECDH provides significant power consumption, computational overhead, delay time advantages along with and smaller device area required if implemented in hardware.

The ECDH library implementation can run on all STM32 microcontrollers. The STM32 Crypto Library User Manual UM1924 details the process for ECC key generation. The mobile application can also implement ECDH to establish a shared private key for secured channel communication over AES.
The STM32 contains a hardware random number generator. The hardware uses an analog source to seed a 32bit random number generation.

V. CONCLUSION

The STM32 hardware and associated library are well equipped to implement secure communication. Since the library is CAVP validated it can be used for authentication, key exchange and encryption without risk that it contains security holes because of poor implementation. The AES encryption hardware minimizes the memory and processor overhead and the library reduces the cost to implement security on a target applications.

CALL US
360.693.2600
US HEADQUARTERS
1714 Broadway, Vancouver, WA 98663
WEBSITE
www.sigmadzn.com
SIGMADESIGN

REFERENCES

[1] UM1924 - User manual - STM32 crypto library

[2] Mike Ryan, "Bluetooth: With Low Energy Comes Low Security," @inproceedings, 179196, Presented as part of the 7th USENIX Workshop on Offensive Technologies, 2013, Washington, D.C., {https://www.usenix.org/conference/woot13/workshop-program/presentation/Ryan}, USENIX

[3] Nitesh Dhanjani, "Abusing the Internet of Things," isbn, 9781491902332, See: http://oreilly.com/catalog/errata.csp?isbn=9781491902332 for release details, Copyright © 2015 Nitesh Dhanjani. All rights reserved, Printed in the United States of America, Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[4] Lightweight security algorithm for low power IoT devices, http://ieeexplore.ieee.org/document/7732296/?reload=true

CALL US
360.693.2600

US HEADQUARTERS
1714 Broadway, Vancouver, WA 98663

WEBSITE
www.sigmadzn.com