C3-Wireless, LLC
624 Atlantis Rd
Melbourne, FL 32904
(800) 769-1132
Fax: (321) 248-0355
www.C3Wireless.com

C3-Wireless BTLE Access Control Guide v0.1

September 20, 2016

C3-WIRELESS

C3-Wireless BTLE Access Control Guide

Table of Contents

Beacons

Beacons will continue to be passive beacons and not support any kind of connection with any device. The beacons cannot be iBeacon compatible under this proposal.

The beacons can be emulated on other hardware (phones), but the protocol is designed around the hardware capabilities of the NRF51822 chip. Initial research indicates that an implementation on Android would be trivial; it's less clear if an iOS is practical. It seems that advertising under iOS is limited to 28 bytes and is split into different arrays for 'name' and 'service uuids'. It's probable that by limiting the protocol to 28 bytes and padding with specific bytes that an iOS device could broadcast a compliant packet.

It's also important to note that if interaction *only* with cell phones is desired (no hardware beacons); there are more appropriate protocols (TOTP; Google Authenticator) that could be adapted to BTLE.

Beacon packets can be generated periodically, or only upon button-press (or a combination of both); depending upon battery life/privacy concerns.

Common Requirements: Beacon

Conceptually, the beacons will act like wireless OTP (one-time password) generators. The protocol is similar to Yubikey; but their products are USB and/or NFC.

The protocol requires the following fields:

Beacon Secret Key Each beacon must have a unique 128-bit key that is known only to the beacon and the authentication server.

Unique Beacon ID Each beacon must have a unique non-secret ID. It is used as an index to lookup the beacons secret key on the server.

Packet Count The beacon must maintain an accurate count of the number of packets sent. This number should never repeat.

Monotonic Clock Each beacon will maintain a clock that increments every second after power on. Starting at a random value and wrapping are acceptable.

Usage Hash The beacon must maintain a value that represents recent button usage. Initially seconds since last usage; other options possible.

Button Flags A bitfield representing the buttons triggered since the last transmission

Option #1: Authentication Only

In this option, all information is transmitted in the clear. A message contains the following:

- Unique Beacon Identifier (4 bytes)
- Packet Counter (4 bytes)
- Monotonic Clock (4 bytes)
- Button Flags (1 byte)
- Usage Hash (4 bytes)
- TxPower@ 1m (1byte, optional)

A 4 byte CMAC tag is added to each broadcast that allows the authentication server to verify that the sender possesses the correct beacon key *and* that the message is unchanged. The probability of guessing the valid MAC tag without the secret key is less than 1 in 4.2 billion.

Option #2: Encryption

In this option, only the Unique Beacon Identifier is transmitted in the clear. The message is composed of three parts, their use and encryptions is as defined as in the EAX mode (using the AES128 block cipher):

Nonce *Number only used once*; used to initialize encryption algorithms, not secret

Header (Authenticated) The header contains the Unique Beacon ID, not secret but authenticated

Payload (Encrypted/Authenticated) The payload contains -Packet Counter (4 bytes) -Monotonic Clock (4 bytes) -Button Flags (1 byte) Usage Hash(4bytes) -TxPower@ 1m(1byte, optional)

Security Analysis

An attacker who can read/write wireless traffic **CAN**:

• Identify users and track their location. Every beacon will announce its unique id; this can be trivially correlated to an actual user.
• Jam all transmissions on the BLE advertising channels

By placing a transmitter at the listener sites and running constant carrier, reception of all beacon emissions can be inhibited, but **CANNOT**:

• Generate a valid beacon packet. Odds are less than 1 in 4.2 billion that an attacker can guess the correct authentication tag; assuming that the key is secret and generated randomly for each beacon.
• Replay a valid packet that was intercepted. If an attacker copies a valid packet and tries to send it again one of two things happen: If the beacon has continued to send packets received by one or more listeners, they will have higher packet counter fields, the authentication server will discard any packet with a packet counter <= the last successfully authenticated packet.

Or

If the attacker copied a packet when the beacon was not in the range of a listener; the server is able to correlate packet broadcast time to wall time by the monotonic clock field in each packet. If the packet isn't received within a few seconds of the expected time, it is discarded.

An attacker who has access physical access to a beacon **CAN**:

• Operate the beacon as if they were the legitimate user. Similar to a garage door opener. Analysis of usage patterns may allow users to be notified of abnormal traffic.
• Clone a beacon

Due to hardware flaws in the NRF51822, the protected areas holding the application and secret key can be dumped for analysis. The dumping process would take a practiced attacker about five minutes. This may be mitigated somewhat by using anti-tampering potting compound to seal the IC and case.

but **CANNOT**:

• Generate a valid packet for any other beacon device All beacon keys are unique and the compromise of one does not reduce the security of the remaining beacons. It also means that users cannot meaningfully subvert each other.
• Cloned devices can be detected. Using potted IC and case; tampering is quite obvious. If the beacon is missing or shows unexpected signs of damage, it's expected that the user would report this and the old beacon would be decommissioned.

Assuming the beacon could be cloned and returned without attracting any notice. The usage hash values of the two devices would diverge (as the devices were used differently). The authentication server can detect and flag a situation where "single beacon" is sending inconsistent data.

C3-Wireless BTLE Access Control Guide

Security Summary

Cloning is possible, but the window of exposure is small. An attacker would have to steal, clone and return a beacon.

Cloning would likely involve milling through the pcb and using needle probes to attach to unexposed pads on the IC package (BGA), or milling through the potting compound an using needle probes on a no-lead package (QFN). If the attacker doesn't write a custom replacement firmware for the original device, its battery would rapidly drain (2-3 weeks, due to another hardware bug).

Without constantly monitoring the original devices wireless traffic, cloned devices would only remain undetected until the next use of the original device. In general, the failure modes of the beacon access control system are most similar to an automatic garage door; clones possible, but fall out of sync upon use and breaking one doesn't compromise any others.

Encryption/Authentication vs. Authentication Only

Encryption obfuscates the details of the protocol, but if you know how the system works it wouldn't be difficult to guess what fields exist (even if you couldn't read/verify them).

Added value from encryption:

- Obscures which button was pressed by the user
- Could increase error into attacker location calculations

By randomly varying the tx power and encrypting the current power in the message only the server could decrypt the current power level, attackers would have to guess.

Problems with encryption:

- More complex implementation
- Worse battery life

The authentication and encryption both use the on-die AES hardware in the NRF51822; but the encryption routine would use this hardware three times per packet whereas the authentication-only setup uses the hardware only once per packet.

Manufacturing / Design Notes

ICs for the beacon shall be programmed at the factory before they are soldered to the board. No traces shall be routed to the debug interface pins.

Upon initial startup, the beacons will run the reinit command and go to sleep.

Optional Hardening

- Guard traces routed near/under the debug pads; breaks could be detected; RAM/firmware wiped.
- Potting compound could seal the IC and/or case
- Have a sliding switch to disable/enable periodic beacon packets (without button press)

This minimizes tracking concerns for end users and may increase difficulty of keeping clones in sync with original state.

Client Provisioning

The client will have access to a device/software that will connect to each beacon via external exposed contacts through the case. Via TWI/Serial connection, the programmer will make an unauthenticated request to the beacon. The beacon will support two instructions: Reinitialize and confirm.

When reinit is triggered, the beacon will wipe the secret key and Unique ID from memory and generate new ones using the RNG hardware. The unique ID and key will be output to the programmer via serial connection. If the ID is unique, the programmer will write the confirm command; and the beacon will begin functioning. Otherwise the beacon will sleep waiting for commands (as it came from the factory).

An attacker can create new unique ids and secret keys, but since the matching data isn't in the authentication database, the beacon is useless.

Provisioning of cell phone beacons would be via QR code, the programming station would randomly generate unique id and key. The phone/tablet would scan the QR code into the beacon application (and the programming station would send the key to the authentication server).

Considerations

Who will host the authentication database? A database provided by C3, or will it be hosted with the main access control application.

If the latter, how will the programming/provisioning station insert new keys and relate them to user records in the access control system. It must also validate individual packets received by the listeners.

If C3-Wireless provides the authentication database, we could provide an API; but then we would need support from access control vendors.