Giới thiệu

- ESP-NOW là một giao thức được phát triển bởi Espressif cho phép nhiều thiết bị có thể kết nối với nhau mà không cần dùng Wifi. Các thiết bị sau khi kết nối được với nhau sẽ là peer-to-peer, và không yêu cầu bắt tay.
- ESP-NOW có thể được xem là một công nghệ được sử dụng cho ESP8266 để truyền các gói dữ liệu với tốc độ cao, nó được ứng dụng trong các thiết bị chiếu sáng thông minh, điều khiển từ xa các cảm biến,...
- Trong ESP-NOW dùng chuẩn IEEE802.11 cùng các hàm IE và mã hóa CCMP đảm bảo được độ tin cậy.

Các chức năng chính

- ESP-NOW có hỗ trợ các đặc điểm sau:
  - Mã hóa và giải mã gói tin unicast
  - Mã hóa trộn và giải mã với peer devices
  - Playload có thể lên tới 250 byte
  - Có hàm callback để xác định được việc truyền dữ liệu thành công hay thất bại
- Tuy nhiên cũng có một số hạn chế:
  - Không hỗ trợ broadcast
  - Giới hạn các peer được mã hóa. Khoảng 10 peer được mã hóa trong mode Station, 6 peer trong SoftAP hoặc SoftAP + mode Station. Peer không mã hóa có thể có số lượng nhiều hơn nhưng phải nhỏ hơn 20.
  - Playload bị giới hạn 250 byte.

Mô tả

Thông tin sẽ bao gồm

- Trong local device
  - PMW
  - Role
- Trong peer
  - Key

    - Địa chỉ MAC
    - Role
    - Channel
- Bảng mô tả

Device	Thông tin	Giá trị/ Độ dài	Mô tả	Ghi chú
Local device	PMK	Độ dài 16 byte	Primary Master Key, ví dụ như KOK trongAPI, dùng để mã hóa Key của peer.	Hệ thống sẽ tạo PMK mặc định, không cần phải config
	Role	IDLE CONTROLLER SLAVE COMBO	Đây là vai trò(role) của device IDLE: chưa phân role CONTROLLER:controller SLAVE: slave COMBO: đảm nhiệm 2 role là controller slave	Role của local device được định nghĩa thông qua SoftAP hoặc Station của ESP-NOW IDLE: việc truyền nhận sẽ không được cho phép CONTROLLER: ưu tiên cho Sation interface SLAVE: ưu tiên cho SoftAP interface COMBO: ưu tiên cho cả SoftAP interface và Station interface
Peer	Key	Độ dài 16 byte	Sử dụng để mã hóa paload key trong quá trình giao tiếp với các peer	
	MacAddress	Độ dài 6 byte	Địa chỉ MAC của peer	Địa chỉ MAC phải cùng với địa chỉ gửi. Ví dụ như nếu gói tin được gửi từ Station interface thì địa chỉ MAC phải giống với địa chỉ của Station
	Role	IDLE CONTROLLER SLAVE COMBO	Đây là vai trò(role) của device IDLE: chưa phân role CONTROLLER:controller SLAVE: slave COMBO: đảm nhiệm 2 role là controller slave	
	Channel	Giá trị từ 0 - 255	Kênh để local device và peer kết nối với nhau	
# Hoạt động				

1. Thiết lập hàm sending callback Việc sử dụng hàm sending callback để thông báo việc truyền nhận là thành công hay thất bại, cần phải thực hiện theo các bước sau nếu sử dụng các hàm này
2. Với kiểu unicast

- Nếu lớp application không nhận được gói tin nhưng hàm callback trả về success thì nguyên nhân có thể do:
    - Tấn công từ các thiết bị khác

- Thiết lập mã hóa Key bị lỗi
- Gói tin bị mất trong lớp application

Lưu ý

Việc bắt tay để đảm bảo tăng tỉ lệ thành công của việc truyền nhận

- Nếu lớp ứng dụng nhận được gói tin nhưng hàm callback trả về failure thì nguyên nhân:
  - Channel đang bận hoặc không nhận được ACK.

- Với kiểu multicast (bao gồm cả broadcast)
  - Nếu hàm callback trả về success nghĩa là gói tin đã được gửi đi thành công
  - Nếu hàm callback trả về failure, nghĩa là gói tin chưa gửi được
- Thiết lập hàm receiving callback Hàm reciecing callback được sử dụng để thông báo tới lớp ứng dụng là gói tin gửi đi đã được nhận bởi peer Hàm này sẽ trả lại địa chỉa MAC của peer và payload của gói tin
- Nếu Key được mã hóa thì API cấu hình PMK(KOK) sẽ được gọi để cấu hình Nếu PMK chưa được cấu hình thì sẽ sử dụng PMK mặc định.
- Lựa chọn giao thức cho device Thông thường Station interface được cài đặt cho CONTROLLER, SoftAP interface cho SLAVE và COMBO.
- Lựa chọn Key cho device. Gọi hàm add peer.
- Gọi hàm gửi để trả về payload.

Lưu ý

Tham khảo thêm các thông tin về API của ESP-NOW tại SDK API Guide

Chương trình

```
void ICACHE_FLASH_ATTR simple_cb(u8 *macaddr, u8 *data, u8 len)
{
    int i;
    u8 ack_buf[16];
    u8 recv_buf[17];
    os_printf("now from[");
    for (i = 0; i < 6; i++)
        os_printf("%02X, ", macaddr[i]);
    os_printf(" len: %d]:", len);
    os_bzero(recv_buf, 17);
    os_memcpy(recv_buf, data, len < 17 ? len : 16);
    if (os_strncmp(data, "ACK", 3) == 0)
        return;
    os_sprintf(ack_buf, "ACK[%08x]", ack_count++);
    esp_now_send(macaddr, ack_buf, os_strlen(ack_buf));
}
void user_init(void)
{
    u8 key[16] = {0x33, 0x44, 0x33, 0x44, 0x33, 0x44, 0x33, 0x44,
                  0x33, 0x44, 0x33, 0x44, 0x33, 0x44, 0x33, 0x44
                 };
    u8 da1[6] = {0x18, 0xfe, 0x34, 0x97, 0xd5, 0xb1};
    u8 da2[6] = {0x1a, 0xfe, 0x34, 0x97, 0xd5, 0xb1};
    if (esp_now_init() == 0) {
        os_printf("esp_now init ok\n");
        esp_now_register_recv_cb(simple_cb);
        esp_now_set_self_role(1);
        esp_now_add_peer(da1, 1, key, 16);
        esp_now_add_peer(da2, 2, key, 16)
    } else {
        os_printf("esp_now init failed\n");
    }
}
void ICACHE_FLASH_ATTR demo_send(u8 *mac_addr, u8 *data, u8 len)
{
    esp_now_send(NULL, data, len);/* Vi du nay se gui toi 2 thiet bi duoc them vao boi ham esp_
//esp_now_send(mac_addr, data, len); /* gui toi dia chi mac mac_addr */
}
```

Tham khảo

- Có thể xem thêm các tài liệu mô tả về ESP-NOW của Espressif