Simple-Pair
用户指南

版本 1.0

关于本手册

本文介绍了乐鑫自主研发的 Simple-Pair 技术，说明了使用方式并提供了示例解决方案。

本手册结构如下：

章	标题	内容
第 1 章	Simple-Pair 技术简介	介绍 Simple-Pair 的技术原理。
第 2 章	Simple-Pair 使用方式	说明 Simple-Pair 的使用流程。
第 3 章	解决方案	提供 Simple-Pair 的示例解决方案。
第 4 章	示例代码	提供 Simple-Pair 的示例代码。

发布说明

日期	版本	发布说明
2016.07	V1.0	首次发布。

目录

1. Simple-Pair 技术简介

1.1. 概述

Simple-Pair 隶属于 ESP IE 的一部分，用于两个设备间快速协商或交换 Key（或其他信息，一般用于交换 Key）。

Simple-Pair 实现了数个步骤就能完成交换 Key。目前，Simple-Pair 仅提供 Station 与 AP 间交换 Key（限定 16 字节），最终需要交换的 Key 是从 AP 端发往 Station 端。为了保护最终的交换步骤，Simple-Pair 包含了一个被称为 TempKey 的 Key。这个 Key 一般存在 Station 端，用户在 Simple-Pair 之前，需要通过某些手段让 AP 和 Station 双方设备都知晓这个 TempKey。

Simple-Pair 设计初衷是用于快速交换某些功能要使用的 Key，能够使两个从未配对过的设备进行配对操作。在整个配对过程中，建议双方设备都暂时禁止休眠，以免无法正常收包。Simple-Pair 需要交换的 Key 存放在 AP 端，经过配对端步骤之后，交换的 Key 将会从 AP 端发到 Station 端，此时双方设备已经完成了 Simple-Pair 的步骤。对应用层来说，配对的过程还未结束，必须使用交换完的 Key 进行一次尝试，尝试成功方可认为 Simple-Pair 是正确的。

> 📖 **说明：**
> 可以用智能手机参与控制 *Simple-Pair* 其中的部分过程。关于如何使用智能手机，请参考第 *3* 章。

2. Simple-Pair 使用方式

在 Simple-Pair 的概念里，设备对应用层只有三个步骤：Set Peer Info，AP ANNOUNCE（STA SCAN）和 START Negotiate。所有的状态都通过回调函数来实现。

用户需要向 Simple-Pair 来注册一个状态回调函数，在 Simple-Pair 运行的过程中，诸如配对完成、协商请求、超时、错误等等都会通过状态回调函数来指示。如 AP 端的状态回调函数传入状态指示有 Station 发来协商请求，则 AP 端需要决定是否同意请求；如状态回调函数传入错误的状态值，则需要根据状态值的不同做不同的处理（详细请见 Demo 程序）。

在进行配对协商之前，AP 要进入 Simple-Pair Announce 模式，Station 则要进入 Simple-Pair Scan 模式。进入这些模式之后，Station 需要向 AP 发送 IEEE802.11 Probe Request（即进行普通的扫描 AP 的操作），这样 Station 才知道哪些 AP 已经准备好 Simple-Pair 了；AP 才知道哪些 Station 是准备好 Simple-Pair 了。

Station 在扫描到 AP 之后，需要从 BSS_INFO 里选择已经准备好 Simple-Pair 的 AP，设置好 AP 的 MAC 地址以及 TempKey，然后发送 STA Negotiate Start。

AP 收到请求之后，需要（或联合手机）判断是否同意请求。若同意，先设置好 Station 的 MAC 地址、TempKey 和交换的 Key，然后发送 AP Negotiate Start；若不同意，设置好 Station 的 MAC 地址和 TempKey，然后发送 AP Negotiate Refuse。

Station 收到 AP Negotiate Start，则会进行协商步骤；若接收到 Refuse，则通过状态回调函数告知应用层，应用层需要做相应的处理。

在协商完成之后，Station 端就获得了 AP 发送端交换的 Key。此时 Simple-Pair 端步骤已经结束。但是应用层需要做一次验证，以确定交换 Key 是正确的。

> ⚠ **注意：**
>
> 在协商开始之前，应用层需要检查 *ESP-NOW* 功能（以后或许会增加新功能）是否有对相同 *MAC* 地址设置过 *Key*，如设置过，请做相应功能的 *Delete* 操作，否则同一个 *MAC* 地址拥有两个 *Key*，可能会造成 *Simple-Pair* 使用错误的 *Key*。除 *Station* / *SoftAP* 的连接会设置密钥之外，系统中不允许其他功能对同一个 *MAC* 地址设置两次密钥。

3. 解决方案

以下以 Simple-Pair 交换 ESP-NOW 的 Key 为例说明一种解决方案。

> 📖 **说明：**
>
> *Simple-Pair* 可用于但不限于 *ESP-NOW* 的 *Key* 交换，也可用于其他功能的 *Key* 交换。

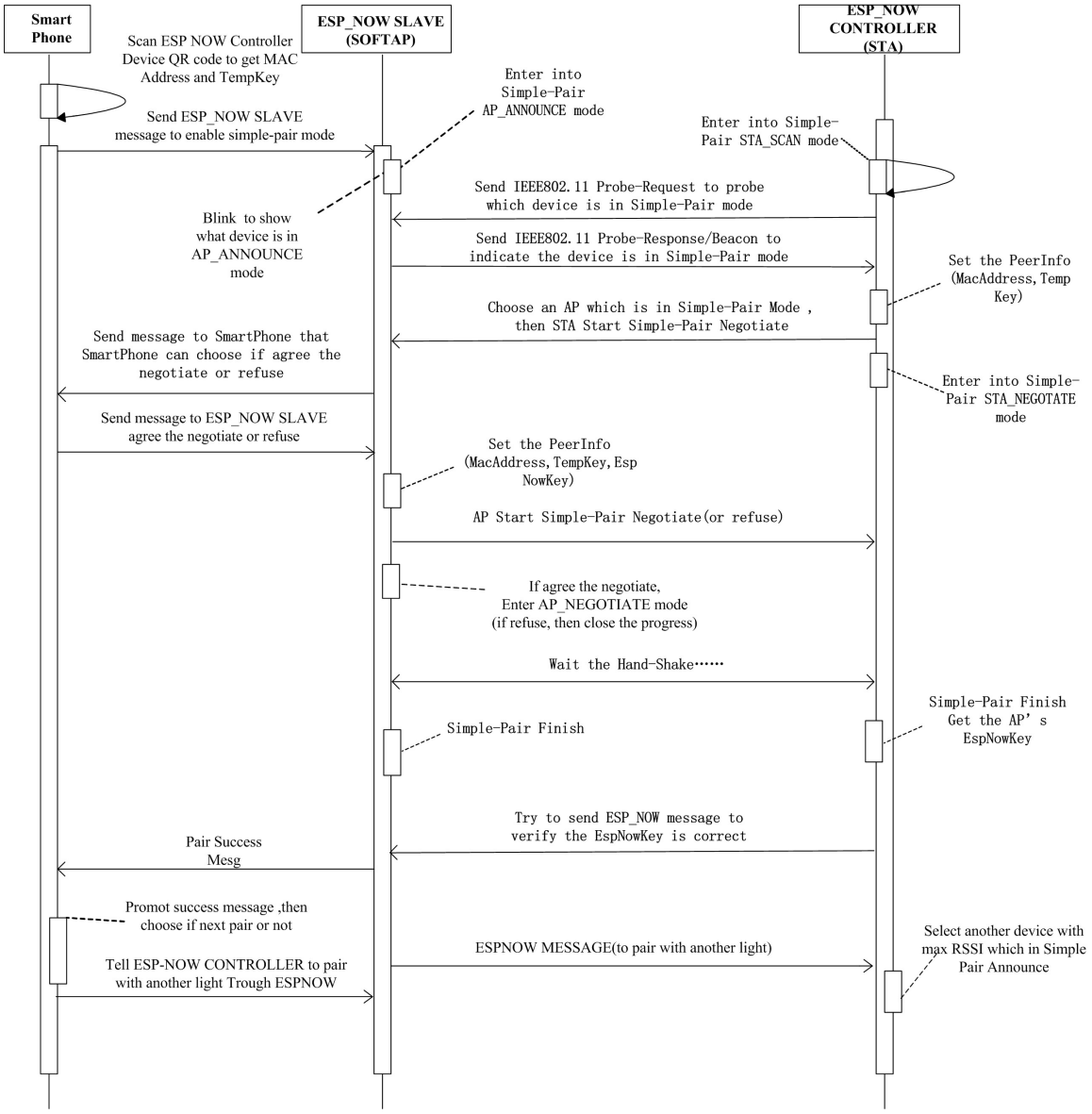

图 3-1. 示例解决方案

Simple-Pair 的步骤完成后，请务必使用交换完成的 Key 做一次验证，验证完成才能认为 Key 的交换是成功的。智能手机可控制其中部分步骤，在成功后，智能手机可控制 ESP-NOW SLAVE 设备是否需要进行下一个 CONTROLLER 的匹配操作。

4. 示例代码

> 📖 **说明：**
> 更多关于 *Simple-Pair API* 的信息，请参考 *ESP8266 Non-OS SDK API 参考*。

```
#include "osapi.h"
#include "user_interface.h"



#include "simple_pair.h"


/
**********************************************************************
**********
 * open AS_STA to compile sta test code
 * open AS_AP to compile ap test code
 * don't open both
 *
**********************************************************************
*******/
//#define AS_STA
#define AS_AP


/
**********************************************************************
*********
 * FunctionName : user_rf_cal_sector_set
 * Description  : SDK just reversed 4 sectors, used for rf init data
and paramters.
 *                We add this function to force users to set rf cal
sector, since
 *                we don't know which sector is free in user's
application.
 *                sector map for last several sectors : ABCCC
 *                A : rf cal
 *                B : rf init data
```

```
 *                C : sdk parameters
 * Parameters   : none
 * Returns      : rf cal sector
*******************************************************************
**********/
uint32 ICACHE_FLASH_ATTR
user_rf_cal_sector_set(void)
{
    enum flash_size_map size_map = system_get_flash_size_map();
    uint32 rf_cal_sec = 0;

    switch (size_map) {
        case FLASH_SIZE_4M_MAP_256_256:
            rf_cal_sec = 128 - 5;
            break;

        case FLASH_SIZE_8M_MAP_512_512:
            rf_cal_sec = 256 - 5;
            break;

        case FLASH_SIZE_16M_MAP_512_512:
        case FLASH_SIZE_16M_MAP_1024_1024:
            rf_cal_sec = 512 - 5;
            break;

        case FLASH_SIZE_32M_MAP_512_512:
        case FLASH_SIZE_32M_MAP_1024_1024:
            rf_cal_sec = 1024 - 5;
            break;

        default:
            rf_cal_sec = 0;
            break;
    }
```

```
    return rf_cal_sec;
}



void ICACHE_FLASH_ATTR
user_rf_pre_init(void)
{
}
/* STA & AP use the same tmpkey to encrypt Simple Pair communication
*/
static u8 tmpkey[16] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06,
0x07,
                  0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f};


#ifdef AS_STA
/* since the ex_key transfer from AP to STA, so STA's ex_key don't
care */
static u8 ex_key[16] = {0x00};
#endif /* AS_STA */


#ifdef AS_AP
/* since the ex_key transfer from AP to STA, so AP's ex_key must be
set */
static u8 ex_key[16] = {0xff, 0xee, 0xdd, 0xcc, 0xbb, 0xaa, 0x99,
0x88,
                  0x77, 0x66, 0x55, 0x44, 0x33, 0x22, 0x11, 0x00};
#endif /* AS_AP */


void ICACHE_FLASH_ATTR
show_key(u8 *buf, u8 len)
{
      u8 i;


      for (i = 0; i < len; i++)
            os_printf("%02x,%s", buf[i], (i%16 == 15?"\n":" "));
}
```

```
#ifdef AS_STA
static void ICACHE_FLASH_ATTR
scan_done(void *arg, STATUS status)
{
    int ret;

    if (status == OK) {

        struct bss_info *bss_link = (struct bss_info *)arg;

        while (bss_link != NULL) {
          if (bss_link->simple_pair) {
                os_printf("Simple Pair: bssid %02x:%02x:%02x:%02x:
%02x:%02x Ready!\n",
                          bss_link->bssid[0], bss_link->bssid[1],
bss_link->bssid[2],
                          bss_link->bssid[3], bss_link->bssid[4],
bss_link->bssid[5]);
            simple_pair_set_peer_ref(bss_link->bssid, tmpkey, NULL);
            ret = simple_pair_sta_start_negotiate();
            if (ret)
                  os_printf("Simple Pair: STA start NEG Failed\n");
            else
                  os_printf("Simple Pair: STA start NEG OK\n");
            break;
          }
            bss_link = bss_link->next.stqe_next;
        }
    } else {
        os_printf("err, scan status %d\n", status);
    }

}
#endif
```

```
void ICACHE_FLASH_ATTR
sp_status(u8 *sa, u8 status)
{
#ifdef AS_STA
      switch (status) {
      case  SP_ST_STA_FINISH:
            simple_pair_get_peer_ref(NULL, NULL, ex_key);
            os_printf("Simple Pair: STA FINISH, Ex_key ");
            show_key(ex_key, 16);
            /* TODO: Try to use the ex-key communicate with AP, for
example use ESP-NOW */

            /* if test ok , deinit simple pair */
            simple_pair_deinit();
            break;
      case SP_ST_STA_AP_REFUSE_NEG:
            /* AP refuse , so try simple pair again  or scan other
ap*/
            os_printf("Simple Pair: Recv AP Refuse\n");
            simple_pair_state_reset();
            simple_pair_sta_enter_scan_mode();
            wifi_station_scan(NULL, scan_done);
            break;
      case SP_ST_WAIT_TIMEOUT:
            /* In negotiate, timeout , so try simple pair again */
            os_printf("Simple Pair: Neg Timeout\n");
            simple_pair_state_reset();
            simple_pair_sta_enter_scan_mode();
            wifi_station_scan(NULL, scan_done);
            break;
      case SP_ST_SEND_ERROR:
            os_printf("Simple Pair: Send Error\n");
```

```
                /* maybe the simple_pair_set_peer_ref() haven't called,
        it send to a wrong mac address */


                break;
            case SP_ST_KEY_INSTALL_ERR:
                os_printf("Simple Pair: Key Install Error\n");
                /* 1. maybe something argument error.
                   2. maybe the key number is full in system*/


                /* TODO: Check other modules which use lots of keys
                            Example: ESPNOW and STA/AP use lots of keys
        */
                break;
            case SP_ST_KEY_OVERLAP_ERR:
                os_printf("Simple Pair: Key Overlap Error\n");
                /* 1. maybe something argument error.
                   2. maybe the MAC Address is already use in ESP-NOW or
        other module
                      the same MAC Address has multi key*/


                /* TODO: Check if the same MAC Address used already,
                            Example: del MAC item of ESPNOW or other
        module */
                break;
            case SP_ST_OP_ERROR:
                os_printf("Simple Pair: Operation Order Error\n");
                /* 1. maybe the function call order has something wrong
        */


                /* TODO: Adjust your function call order */
                break;
            default:
                os_printf("Simple Pair: Unknown Error\n");
                break;
            }
```

```
#endif /* AS_STA */

#ifdef AS_AP
      switch (status) {
      case  SP_ST_AP_FINISH:
            simple_pair_get_peer_ref(NULL, NULL, ex_key);
            os_printf("Simple Pair: AP FINISH\n");

            /* TODO: Wait STA use the ex-key communicate with AP, for
example use ESP-NOW */

            /* if test ok , deinit simple pair */
            simple_pair_deinit();
            break;
      case SP_ST_AP_RECV_NEG:
            /* AP recv a STA's negotiate request */
            os_printf("Simple Pair: Recv STA Negotiate Request\n");

            /* set peer must be called, because the simple pair need
to know what peer mac is */
            simple_pair_set_peer_ref(sa, tmpkey, ex_key);

            /* TODO:In this phase, the AP can interaction with Smart
Phone,
                  if the Phone agree, call start_neg or refuse */
            simple_pair_ap_start_negotiate();
            //simple_pair_ap_refuse_negotiate();
            /* TODO:if refuse, maybe call simple_pair_deinit() to
ending the simple pair */

            break;
      case SP_ST_WAIT_TIMEOUT:
            /* In negotiate, timeout , so re-enter in to announce
mode*/
            os_printf("Simple Pair: Neg Timeout\n");
            simple_pair_state_reset();
```

```
                simple_pair_ap_enter_announce_mode();
                break;
        case SP_ST_SEND_ERROR:
                os_printf("Simple Pair: Send Error\n");
                /* maybe the simple_pair_set_peer_ref() haven't called,
it send to a wrong mac address */

                break;
        case SP_ST_KEY_INSTALL_ERR:
                os_printf("Simple Pair: Key Install Error\n");
                /* 1. maybe something argument error.
                   2. maybe the key number is full in system*/

                /* TODO: Check other modules which use lots of keys
                            Example: ESPNOW and STA/AP use lots of keys
*/
                break;
        case SP_ST_KEY_OVERLAP_ERR:
                os_printf("Simple Pair: Key Overlap Error\n");
                /* 1. maybe something argument error.
                   2. maybe the MAC Address is already use in ESP-NOW or
other module
                      the same MAC Address has multi key*/

                /* TODO: Check if the same MAC Address used already,
                            Example: del MAC item of ESPNOW or other
module */
                break;
        case SP_ST_OP_ERROR:
                os_printf("Simple Pair: Operation Order Error\n");
                /* 1. maybe the function call order has something wrong
*/

                /* TODO: Adjust your function call order */
                break;
        default:
```

```
                os_printf("Simple Pair: Unknown Error\n");
                break;
        }

#endif /* AS_AP */
}

void ICACHE_FLASH_ATTR
init_done(void)
{
        int ret;

#ifdef AS_STA
        wifi_set_opmode(STATION_MODE);

        /* init simple pair */
        ret = simple_pair_init();
        if (ret) {
                os_printf("Simple Pair: init error, %d\n", ret);
                return;
        }
        /* register simple pair status callback function */
        ret = register_simple_pair_status_cb(sp_status);
        if (ret) {
                os_printf("Simple Pair: register status cb error, %d\n",
ret);
                return;
        }

        os_printf("Simple Pair: STA Enter Scan Mode ...\n");
        ret = simple_pair_sta_enter_scan_mode();
        if (ret) {
                os_printf("Simple Pair: STA Enter Scan Mode Error, %d\n",
ret);
                return;
```

```
        }
        /* scan ap to searh which ap is ready to simple pair */
        os_printf("Simple Pair: STA Scan AP ...\n");
          wifi_station_scan(NULL,scan_done);
#endif
#ifdef AS_AP
        wifi_set_opmode(SOFTAP_MODE);

        /* init simple pair */
        ret = simple_pair_init();
        if (ret) {
              os_printf("Simple Pair: init error, %d\n", ret);
              return;
        }
        /* register simple pair status callback function */
        ret = register_simple_pair_status_cb(sp_status);
        if (ret) {
              os_printf("Simple Pair: register status cb error, %d\n",
ret);
              return;
        }

        os_printf("Simple Pair: AP Enter Announce Mode ...\n");
        /* ap must enter announce mode , so the sta can know which ap
is ready to simple pair */
        ret = simple_pair_ap_enter_announce_mode();
        if (ret) {
              os_printf("Simple Pair: AP Enter Announce Mode Error, %d
\n", ret);
              return;
        }

#endif

}
```

```
void ICACHE_FLASH_ATTR
user_init(void)
{
      system_init_done_cb(init_done);
}
```

乐鑫 IOT 团队

www.espressif.com