主要关注四个文件：

app_adv.c

app_mesh_adv.c

app_task.c

app_scan.c

一、程序跑起来会默认进行 scan 和发送普通广播包和 mesh 广播包

1、 scan 的数据结果在 gapm_ext_adv_report_ind_handler 这个函数打印输出，
在此修改过滤收到的是 mesh 广播就可以传到 mesh 的 bearer 层

2、 默认发送一个固定的 mesh 广播，广播内容在 appm_mesh_build_adv_data 这个函数里面设置。从 bearer 层过来的 mesh 广播数据需要通过 appm_mesh_updata_adv_data 函数进行实时更新。每次更新广播完成事件在 gapm_cmp_evt_handler 这个函数收到。只有收到完成事件，才能做下次更新。

3、普通广播的数据内容设置在 appm_build_adv_data 这个函数。

4、广播间隔等信息在以下宏定义设置

```
/// Advertising channel map - 37, 38, 39
#define APP_ADV_CHMAP           (0x07)
/// Advertising minimum interval - 40ms (64*0.625ms)
#define APP_ADV_INT_MIN         (160 )
/// Advertising maximum interval - 40ms (64*0.625ms)
#define APP_ADV_INT_MAX         (160)
/// Fast advertising interval
#define APP_ADV_FAST_INT        (32)

/// Advertising channel map - 37, 38, 39
#define APP_MESH_ADV_CHMAP          (0x07)
/// Advertising minimum interval - 40ms (64*0.625ms)
#define APP_MESH_ADV_INT_MIN        (80 )
/// Advertising maximum interval - 40ms (64*0.625ms)
#define APP_MESH_ADV_INT_MAX        (80)
/// Fast advertising interval
#define APP_MESH_ADV_FAST_INT       (32)
```

5、关于 scan\advertising 各种不同模式的设置

```
static void appm_start_scaning(void)
{
    uart_printf("func %s,scan_state:%d\r\n",__func__,app_env.scan_state);
    if (app_env.scan_state == APP_SCAN_STATE_CREATED)
    {
        // Prepare the GAPM_ACTIVITY_START_CMD message
        struct gapm_activity_start_cmd *p_cmd = KE_MSG_ALLOC(GAPM_ACTIVITY_START_CMD,
                                                             TASK_GAPM, TASK_APP,
                                                             gapm_activity_start_cmd);

        p_cmd->operation = GAPM_START_ACTIVITY;
        p_cmd->actv_idx = app_env.scan_actv_idx;

        p_cmd->u_param.scan_param.type = GAPM_SCAN_TYPE_OBSERVER;//GAPM_SCAN_TYPE_GEN_DISC;//GAPM_SCAN_TYPE_OBSERVER;//;

        p_cmd->u_param.scan_param.prop = GAPM_SCAN_PROP_PHY_1M_BIT ;//| GAPM_SCAN_PROP_ACTIVE_1M_BIT;  设置不同的SCAN模式

        //p_cmd->u_param.scan_param.scan_param_1m.scan_intv = app_env.scan_intv;
        //p_cmd->u_param.scan_param.scan_param_1m.scan_wd = app_env.scan_wd;
```

```
static void appm_mesh_create_advertising(void)
{

    if (app_env.mesh_adv_state == APP_MESH_ADV_STATE_IDLE)
    {
        // Prepare the GAPM_ACTIVITY_CREATE_CMD message
        struct gapm_activity_create_adv_cmd *p_cmd = KE_MSG_ALLOC(GAPM_ACTIVITY_CREATE_CMD,
                                                                  TASK_GAPM, TASK_APP,
                                                                  gapm_activity_create_adv_cmd);

        // Set operation code
        p_cmd->operation = GAPM_CREATE_ADV_ACTIVITY;

        // Fill the allocated kernel message
        p_cmd->own_addr_type = GAPM_STATIC_ADDR;
        p_cmd->adv_param.type = GAPM_ADV_TYPE_LEGACY;//GAPM_ADV_TYPE_EXTENDED;//;
        p_cmd->adv_param.prop = GAPM_ADV_PROP_NON_CONN_NON_SCAN_MASK ;//| GAPM_ADV_PROP_SCAN_REQ_NTF_EN_BIT;//GAPM_ADV_PROP_BROADCAST_NON_SCAN_MASK
        p_cmd->adv_param.filter_pol = ADV_ALLOW_SCAN_ANY_CON_ANY;
        p_cmd->adv_param.prim_cfg.chnl_map = APP_MESH_ADV_CHMAP;
        p_cmd->adv_param.prim_cfg.phy = GAP_PHY_LE_1MBPS;


        p_cmd->adv_param.disc_mode = GAPM_ADV_MODE_GEN_DISC;
        p_cmd->adv_param.prim_cfg.adv_intv_min = APP_MESH_ADV_INT_MIN;
        p_cmd->adv_param.prim_cfg.adv_intv_max = APP_MESH_ADV_INT_MAX;


        // Send the message
        ke_msg_send(p_cmd);

        // Keep the current operation
        app_env.mesh_adv_state = APP_MESH_ADV_STATE_CREATING;
        // And the next expected operation code for the command completed event
        app_env.mesh_adv_op = GAPM_CREATE_ADV_ACTIVITY;
    }
}
```

设置不同的广播模式