### nl80211\_start\_ap

*/\*\**

*\* nl80211\_start\_ap - 启动Wi-Fi接入点或P2P组所有者*

*\* @skb: 包含命令的网络缓冲区*

*\* @info: 泛型网络层信息结构*

*\**

*\* 此函数负责通过从网络层消息解析必要的参数并调用驱动程序的start\_ap操作来启动Wi-Fi接入点（AP）或P2P组所有者（GO）。*

*\* 它验证输入参数，设置AP设置结构，并根据驱动程序和设备能力处理各种可选设置。*

*\**

*\* 返回值：成功时返回0，失败时返回负错误码。*

*\*/*

static int nl80211\_start\_ap(struct sk\_buff \*skb, struct genl\_info \*info)

1. 设备检查

- 确认`net\_device`是否为AP或P2P GO类型。

- 确认注册设备`cfg80211`是否支持启动AP。

2. 参数验证

- 检查Beacon间隔、DTIM周期和Beacon头部是否已提供。

- 解析Beacon信息，包括速率、信号强度等。

- 验证Beacon间隔是否有效。

- 解析并验证其他可选属性，如SSID、隐藏SSID标志、隐私设置、认证类型、加密设置、不活动超时、P2P CTWindow、P2P OPPPS、工作频率、TX速率、SMPS模式、PBSS标志、ACL策略、TWT Responder标志、HE OBSS PD、HE BSS Color、FILS Discovery、未请求的广播探针响应。

3. 信道配置

- 如果提供了工作频率，则解析频道定义。

- 如果设备预设了频道，则使用预设的频道定义。

- 如果没有指定频道，尝试获取AP的默认频道。

4. TX速率配置

- 如果提供了TX速率，则解析并验证速率是否在当前频道下有效。

5. AP启动

- 使用收集到的所有参数调用`rdev\_start\_ap`来实际启动AP。

- 更新无线设备的状态信息，如SSID、Beacon间隔、频道定义等。

6. 错误处理

- 在任何步骤中遇到错误，函数会返回相应的错误码。

- 清理分配的资源，如ACL数据结构。

总结

`nl80211\_start\_ap`函数是一个复杂的网络管理函数，用于初始化和配置无线AP或P2P GO。它处理大量的输入参数，确保它们符合预期的格式和值域，然后使用这些参数来配置无线设备，使其能够作为AP运行。此外，它还负责处理各种可能的错误情况，以确保系统的稳定性和安全性。

之后被声明成一个结构体数组，不知道被谁调用

### nl80211\_add\_commands\_unsplit

*/\*\**

*\* nl80211\_add\_commands\_unsplit - 向nl80211消息中添加不拆分的命令*

*\* @rdev: 注册的802.11设备结构体*

*\* @msg: SKB消息缓冲区*

*\**

*\* 此函数通过在给定的SKB消息缓冲区中添加nl80211命令，来通知用户空间关于可用的*

*\* 802.11操作。它静态链接，意味着它只在该模块内部使用。*

*\**

*\* 返回值: 添加的命令数量，或在失败时返回-ENOBUFS。*

*\*/*

static int nl80211\_add\_commands\_unsplit(struct cfg80211\_registered\_device \*rdev,

struct sk\_buff \*msg)

### nl80211\_send\_wiphy

*/\*\**

*\* nl80211\_send\_wiphy - 发送nl80211消息来描述一个wiphy设备*

*\* @rdev: 注册的802.11设备结构体*

*\* @cmd: nl80211命令类型*

*\* @msg: 包含nl80211消息的SKB*

*\* @portid: netlink端口ID*

*\* @seq: netlink消息序列号*

*\* @flags: netlink消息标志*

*\* @state: 用于控制消息分片和状态跟踪的结构体*

*\**

*\* 返回值: 成功返回0，否则返回负错误码*

*\**

*\* 这个函数被用于构造和发送描述wiphy设备属性的nl80211 netlink消息。它根据不同的*

*\* nl80211命令和状态，动态地添加各种属性到消息中。对于新wiphy设备的描述，它会包含*

*\* 基本的设备属性，如设备索引、名称、配置生成号等。此外，根据设备的支持特性，它还*

*\* 可能包含重传参数、频段、频率、命令支持等详细信息。*

*\*/*

static int nl80211\_send\_wiphy(struct cfg80211\_registered\_device \*rdev,

enum nl80211\_commands cmd,

struct sk\_buff \*msg, u32 portid, u32 seq,

int flags, struct nl80211\_dump\_wiphy\_state \*state)

### wiphy\_new\_nm

*/\*\**

*\* @brief 创建并初始化一个新的无线物理层（wiphy）设备实例*

*\**

*\* 此函数负责根据提供的操作集、私有数据大小以及请求的设备名来创建和初始化wiphy设备。*

*\* 它会分配必要的内存空间，设置wiphy的属性，并确保所有必需的初始化步骤得以执行。*

*\**

*\* @param ops 无线物理层设备的操作集，包含一系列用于设备交互的函数指针。*

*\* @param sizeof\_priv 私有数据区域的大小，用于设备特定的数据存储。*

*\* @param requested\_name 请求的设备名称，如果未提供则会使用默认名称。*

*\**

*\* @return 返回指向新创建的wiphy设备的指针，如果创建失败则返回NULL。*

*\*/*

struct wiphy \*wiphy\_new\_nm(const struct cfg80211\_ops \*ops, int sizeof\_priv,

const char \*requested\_name)

### nl80211\_parse\_fils\_discovery

*/\*\**

*\* nl80211\_parse\_fils\_discovery - 解析FILS发现相关的Netlink属性*

*\* @rdev: 注册的802.11设备结构体*

*\* @attrs: 包含FILS发现属性的Netlink属性数组*

*\* @params: 用于存储FILS发现参数的结构体*

*\**

*\* 此函数用于解析来自Netlink消息的FILS发现相关属性，并将它们存储在*

*\* cfg80211\_ap\_settings结构体中。它首先检查wiphy扩展特性是否支持FILS发现，*

*\* 然后解析属性数组，提取并存储模板长度、最小间隔、最大间隔和模板数据。*

*\**

*\* 返回值:*

*\* 如果解析成功，返回0；如果缺少必要属性，返回-EINVAL。*

*\*/*

static int nl80211\_parse\_fils\_discovery(struct cfg80211\_registered\_device \*rdev,

struct nlattr \*attrs,

struct cfg80211\_ap\_settings \*params)

### ieee80211\_start\_ap

*/\*\**

*\* ieee80211\_start\_ap - 初始化AP接口*

*\* @wiphy: 无线物理层设备结构体*

*\* @dev: 网络设备结构体*

*\* @params: AP设置参数*

*\**

*\* 该函数用于初始化一个802.11接入点（AP）接口。它配置了AP的各种参数，如SSID、信道、 Beacon间隔等。*

*\* 在成功初始化后，AP将准备好接受客户端的连接请求。*

*\**

*\* 返回值:*

*\* 0: AP初始化成功*

*\* -EALREADY: AP已初始化*

*\* -ENOTSUPP: 不支持的SMPS模式*

*\* 其他错误码: 配置AP参数时发生错误*

*\*/*

static int ieee80211\_start\_ap(struct wiphy \*wiphy, struct net\_device \*dev,

struct cfg80211\_ap\_settings \*params)

### rdev\_start\_ap

*/\*\**

*\* rdev\_start\_ap - 用于启动一个接入点（AP）模式的网络设备*

*\* @rdev: 注册的802.11设备结构体，表示无线设备的注册信息*

*\* @dev: 网络设备结构体，表示需要启动AP模式的网络设备*

*\* @settings: AP的配置设置，包括SSID、密码等信息*

*\**

*\* 此函数通过调用注册设备的操作集合中的start\_ap函数来启动一个接入点（AP）。*

*\* 在调用前，它会通过trace\_rdev\_start\_ap记录调试信息。调用后，通过*

*\* trace\_rdev\_return\_int记录返回值。这有助于调试和理解AP启动过程中的*

*\* 数据流动和执行流程。*

*\**

*\* 返回值: 返回启动AP操作的结果，通常为0表示成功，非0表示失败。*

*\*/*

static inline int rdev\_start\_ap(struct cfg80211\_registered\_device \*rdev,

                struct net\_device \*dev,

                struct cfg80211\_ap\_settings \*settings)

/\* 调用设备操作集合中的start\_ap函数来启动AP \*/

    ret = rdev->ops->start\_ap(&rdev->wiphy, dev, settings);

### ieee80211\_stop\_ap

*/\*\**

*\* ieee80211\_stop\_ap - 停止AP接口操作*

*\* @wiphy: 无线物理层设备结构体*

*\* @dev: 网络设备结构体*

*\**

*\* 该函数用于停止一个IEEE 802.11接入点（AP）接口的操作。它涉及到取消广播帧的发送、*

*\* 释放频道资源、停止信标和探测响应的发送等操作。此外，它还处理了与频道切换、*

*\* 信道测量相关的资源清理。*

*\**

*\* 返回值:*

*\* 返回0，表示成功停止AP接口；如果在取消广播帧发送过程中遇到错误，则返回相应的错误码。*

*\*/*

static int ieee80211\_stop\_ap(struct wiphy \*wiphy, struct net\_device \*dev)