## 1.概述

### 1.1背景

设备接入商云后会将商云相关的配置功能清除并屏蔽设备端修改入口，包括无线配置、认证配置、射频配置等。但是目前商云上对这些功能没有完全实现，导致设备接入商云后部分配置无法正常使用，影响用户体验。同时设备绑定到商云时会将设备原有配置清除，导致设备配置丢失。

因此需要设计方案解决这个问题。

### 1.2商云配置现状

当前商云上无线相关配置都是基于项目的，其中SSID和认证基于绑定设备生效，认证用户、免认证策略全局生效，因此修改SSID条目会影响到所有已绑定该条目的设备，认证用户、免认证用户的操作会影响项目中所有支持该功能的设备。

## 2.现有上线配置同步流程

目前商云配置同步的流程有两种：

1.清除设备端旧配置，下发云端配置。此类配置同步时云端会先下发cloud\_reset命令将设备端已有配置清除，然后依次下发云端配置，相关配置项包括无线服务、免认证、web认证用户、Fat AP的射频配置。

2.拉取设备端配置。此类配置同步时，云端会拉取设备端的配置项，存储到云端，包括Fit AP的射频配置。

## 3.双向同步方案

### 3.1 方案目标

设备端不屏蔽相关配置页面，云端和设备端都能修改配置，修改后进行同步，云端修改下发到设备端，设备端修改后通知云端，云端再拉取配置进行同步。

### 3.2 fearture更新

设备端通过feature告知云端支持双向同步功能，feature暂定为x10000。

### 3.3 配置同步原则

当设备和云端配置版本不一致时，以设备端配置为准，将设备配置同步到云端。

### 3.4 模块配置版本号管理

①版本号的定义：设备端通过<String model,<String subModel,Long configId>>维护模块-配置版本号，目前云端的cloudConfigId是由DeviceConfig类存储的，DeviceConfig类的定义如下,暂定云端沿用<String model,Long configId>标识版本号。

|  |
| --- |
| Class DeviceConfig |
| Long Id |
| Long siteId |
| Long deviceIndex |
| String module |
| String content(configId) |
| String info |

②版本号的更新规则

设备端修改则版本号置为0，云端修改配置下发时沿用现在的修改逻辑。

③版本号下发规则

云端为每个设备的每个module维护一个是否同步的标志，缓存redis中，格式为

nms:device:moduleSync:${project\_id}:${mac}=true/false。用来表示当前模块设备和云端的配置是否是同步状态，如果某一次设备通知处理出现异常，那么该标志被设置为false，后续进行模块配置同步后不再更新云端moduleConfigId，以保持云端和设备端的ID存在差异，下一次设备上线时再次触发上线同步流程。

### 3.5配置绑定设备

免认证策略、WEB认证用户添加绑定设备的配置项让用户可以选择下发到指定设备，默认在所有设备上生效，用户可以取消选中指定设备。SSID配置、射频目前配置已经是基于设备的，无需调整。认证全局参数，需调整为列表模式。

### 数据库设计

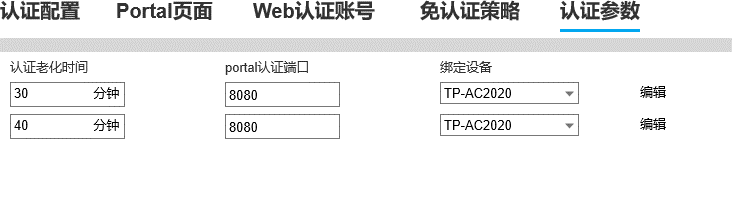
云端新增配置条目和设备绑定关系表，具体信息如下：

|  |  |  |
| --- | --- | --- |
| 字段名 | 字段类型 | 字段含义 |
| id | bigint | 绑定关系条目ID |
| project\_id | bigint | 项目ID |
| config\_module | varchar(32) | 配置模块名称，值定义如下  免认证：freestrategy  认证用户:auth\_user  认证全局参数：auth\_global |
| module\_entry\_id | bigint | 配置条目对应的ID |
| device\_entry\_id | varchar(64) | 设备端条目ID |
| device\_index | bigint | 绑定设备的ID |

## 4.同步方案设计

### 4.1 配置下发（以全局认证参数为例）

4.1.1 配置形式调整为列表操作



4.1.2 后台逻辑调整

云端目前的认证参数是基于项目下发的，需要改成基于设备下发，从wireless到config-server的下发消息定义如下：

|  |
| --- |
| Class SetConfigMsg |
| private Long siteId; |
| private Long deviceGroupId; |
| private List<Long> deviceIndexes; |
| private ConfigType configType; |
| private ConfigModule module; |
| private Object configTarget; |
| private ConfigTargetType configTargetType; |
| private Long moduleConfigSequence; |
| private ConfigContext configContext; |

①根据项目id和PortalGlobalId（存储在configTarget字段）拿到PortalGlobal发送到config-server。

②configTargetType为device，拿到targetDevices，对targetDevices进行排除fitAp操作（填充fitAp的belongDevice），更新configId。

所以云端的调整只需要1.传入用户选择的设备信息：deviceIndexes；2.ConfigTargetType设置为device。

### 4.2设备上线的配置同步

#### 4.2.1现状

设备上报上线消息的configId基于设备。私云使用redis-mysql存储的cloudConfigId也是基于设备，公云使用redis存储cloudConfigId也是基于设备。

#### 4.2.2 云端调整

将云端存储cloudConfigId改为基于模块。

#### 4.2.3 上线配置同步流程图



### 4.3用户触发同步的配置拉取

采用用户触发同步配置方案时，设备端配置发生变化时不需要通知云端具体的改动条目，只需通知改动的模块，云端收到通知后记录设备端模块配置已发生变化，在设备列表和设备详情接口返回中添加deviceConfigChanged=true/false字段表明设备是否需要同步配置，当为true时前端页面提供同步配置入口，点击后云端触发设备配置同步依次同步设备发生变化的模块配置。

#### 4.3.1 模块配置同步标志维护

云端在redis中为每台设备维护需要同步的模块信息，key初步定为nms:dev:config:syncModule:${project\_id}:${deviceIndex},value为Set，包含需要同步的模块名称。

#### 4.3.2 更改通知

①消息设计

|  |  |  |
| --- | --- | --- |
| noticeType  String | MODULE\_CONFIG\_SYNC\_NOTIFY | |
| attachment  Object | moduleName  String | 变更的模块名，对应TDCP接口的module |
| subModuleName  String | 子模块名称，比如auth\_user\_list等，如果不存在子模块则不传。 |

②更高通知流程



#### 4.3.3 接口调整

新增是否需要同步配置的接口。



获取设备列表和设备详情的时候，访问是否需要同步配置的接口，如果用户选择同步则进入同步流程，否则保持不同步的状态。

#### 4.3.4同步流程



模块配置同步子流程拉取设备端条目。

## 5. 模块配置同步

### 5.1认证参数

#### 5.1.1 配置下发流程

5.1.1.1 配置下发流程图



5.1.1.2云端需要调整的地方

1.配置增删改查时需要更新version字段。

2.配置下发时需要传入deviceIndexes，更改配置目标类型为device。

3.更新configId细化到模块粒度。

4.构造下发消息请求体时，

5.更新configId时，

#### 5.1.2 拉取设备端配置同步流程

5.1.2.1 TDCP接口

|  |
| --- |
| **request：**  {"method":"get","authentication":{"name":"portal\_para"}}  **response：**  {  "error\_code": 0,  "authentication": {  "portal\_para": {  ".name": "portal",  ".type": "global",  "default\_auth\_mode": "vlan",  "age\_time": "30",  "auth\_mode": "ssid",  "age\_enable": "on",  "cmcc\_portal\_port": "50100",  ".anonymous": false,  "portal\_port": "8080",  "cmcc\_port": "2000"  }  }  } |

5.1.2.2云端参数对应关系

|  |  |  |
| --- | --- | --- |
| **portal认证参数** | **设备** | **商云** |
|
| 全局参数ID |  | portalGlobalId |
|
| 场所ID |  | siteId |
|
| 老化时间 | age\_time | ageingTime |
|
| 认证端口 | portal\_port | portalPort |
|
| 认证老化开关 | age\_enable |  |
|
| cmcc\_port | cmcc\_port |  |
|
| cmcc\_portal\_port | cmcc\_portal\_port |  |
|
| 认证模式 | auth\_mode |  |
|

5.1.2.3配置同步流程

设备认证全局参数之前是单条目的，且未存储云端的portalGlobalId信息，因此同步时必然是modifyType=update，cloudEntryId=””的情形，具体流程如下。



### 5.2 WEB认证账号

#### 5.2.1 配置下发

#### 5.2.2 配置拉取

5.2.2.1 TDCP接口

|  |
| --- |
| **request：**  {"method":"get","authentication":{"table":"userlist","para":{"start":0,"end":9}}}  **response：**  {  "error\_code": 0,  "authentication": {  "count": {  "userlist": 1  },  "userlist": [{  "userlist\_1": {  "mac": "",  "allow\_auth\_time": "00%3a00-24%3a00",  "mac\_bind\_type": "none",  "user\_type": "formal",  "enable": "on",  "expire\_time": "2020%2f12%2f31",  "down\_limit": "0",  "auth\_user\_id": "",  "username": "admin",  "allow\_user": "1",  "name": "",  "entry\_id": "1",  "telephone": "",  "password": "123456",  "up\_limit": "0",  "note": ""  }  }]  }  } |

#### 5.2.2云端参数对应关系

|  |  |  |
| --- | --- | --- |
| **参数** | **设备** | **商云** |
|
| 用户ID | auth\_user\_id | authUserId |
|
| 用户类型 | user\_type | userType |
|
| 开启 | enable | enable |
|
| 备注 | note | notice |
|
| 用户名 | username | username |
|
| 密码 | password | password |
|
| 允许登录时间 | allow\_auth\_time | effectiveTimePeriod |
|
| 失效时间（截至） | expire\_time | expireDate |
|
| mac绑定 | mac\_bind\_type | macBindType |
|
| 静态绑定的mac | mac | mac |
|
| 上行带宽限制 | up\_limit | upstreamRate |
|
| 下行带宽限制 | down\_limit | downstreamRate |
|
| 姓名 | name | name |
|
| 电话 | telephone | phone |
|
| 条目唯一标识ID | entry\_id |  |
|
| 允许同时登陆的用户数 | allow\_user | maxSessions |
|
| 场所ID |  | siteId |
|

#### 5.2.3云端调整

1.云端条目可能会出现重名条目，编辑时如果条目名称有改动则进行重名校验，名称未改动时不做重名校验，添加时需要进行重名校验。

2.删除条目时需要同步删除对应的绑定关系；

3.解除绑定关系时，向配置服务器下发配置消息中需要带上解绑的deviceIndex信息。

配置下发流程

①add

wireless

> SetConfigMsg msg

1.

SetConfigMsg msg = new SetConfigMsg();  
msg.setSiteId(vo.getSiteId());  
msg.setConfigType(type);  
msg.setModule(ConfigModule.*authUser*);  
  
msg.setConfigTarget(vo.getAuthUserId());  
msg.setConfigTargetType(ConfigTargetType.*site*);  
msg.setConfigContext(new ConfigContext(context.getCloudUsername(), context.getToken()));  
  
MsgUtil.*initSetConfigMsgSequence*(vo.getSiteId(), msg);

2.tag=set

3.发送

config-server

ConfigMsgProcessor. handleDeviceConfigMsg -> handleSetConfigMsg

1. 获取目标配置设备：deviceParse.getConfigTargetDevices(setConfigMsg)，过滤不在线设备并将更新不在线的设备的configVersion。

2.目标设备列表过滤AP，拿到portalGlobal条目。

3.忽略子设备，构造

ConfigDetail（configContext,ConfigDevice,ConfigType,Module,siteId,deviceGroupId），并将configDetail和SetConfigMsg（带有云端条目ID）以及AuthUser一起下发。

②delete

①edit

wireless

-> SetConfigMsg msg

1.

msg.setSiteId(globalVO.getSiteId());  
msg.setConfigType(type);  
msg.setModule(ConfigModule.*globalAuth*);  
  
msg.setConfigTarget(globalVO.getPortalGlobalId());  
msg.setConfigTargetType(ConfigTargetType.*site*);  
msg.setConfigContext(new ConfigContext(context.getCloudUsername(), context.getToken()));  
  
MsgUtil.*initSetConfigMsgSequence*(globalVO.getSiteId(), msg);

2.tag=set

3.发送

config-server

ConfigMsgProcessor. handleDeviceConfigMsg -> handleSetConfigMsg

1. 获取目标配置设备：deviceParse.getConfigTargetDevices(setConfigMsg)，过滤不在线设备并将更新不在线的设备的configVersion。

2.目标设备列表过滤AP，拿到portalGlobal条目。

3.忽略子设备，构造

ConfigDetail（configContext,ConfigDevice,ConfigType,Module,siteId,deviceGroupId），并将configDetail和SetConfigMsg（带有云端条目ID）一起下发。

### 5.3 免认证策略

#### 5.3.1TDCP接口

|  |
| --- |
| **request：**  {"method":"get","authentication":{"table":"free\_auth\_list","para":{"start":0,"end":9}}}  **response：**  {  "error\_code": 0,  "authentication": {  "count": {  "free\_auth\_list": 1  },  "free\_auth\_list": [{  "free\_auth\_list\_1": {  ".name": "free\_auth\_list\_1",  ".anonymous": false,  "protocol": "udp",  "id": "1",  "src\_port": "68-68",  ".index": 4,  "name": "dhcp%20client",  "dst\_port": "67-67",  "enable": "on",  ".type": "free\_auth\_list",  "match\_type": "quintuple"  }  }]  }  } |

#### 5.3.2 云端参数对应关系

|  |  |  |
| --- | --- | --- |
| **免认证策略参数** | **设备** | **商云** |
|
| 是否启用 | enable | enable |
|
| 免认证策略类型 | match\_type | strategyType |
|
| 源MAC地址 | src\_mac | srcMac |
|
| 源IP地址范围 | src\_ip\_set | srcIpset |
|
| 目的IP地址范围 | dst\_ip\_set | dstIpset |
|
| 源端口范围 | src\_port | sport |
|
| 目的端口范围 | dst\_port | dport |
|
| 协议类型 | protocol | serviceProtocol |
|
| url | url | url |
|
| 备注 | note |  |
|
| 免认证策略id值 | free\_auth\_id | freeStrategyId |
|
| 源vlan值 | src\_vlan |  |
|
| 生效区段 | interface |  |
|
| 场所id |  | siteId |
|
| 免认证条目名称 | name | freeStrategyName |
|

#### 5.3.3 配置同步流程

免认证策略的同步流程和web认证账号一致，需要额外考虑设备默认条目的同步方式。

不同的设备有不同的默认条目，这些配置条目不允许编辑删除，因此云端不允许编辑删除。需要云端识别出来那些条目是系统默认的，识别规则由设备端提供。

#### 5.3.4 云端流程调整

和WEB认证用户一致。

### 5.4 SSID

#### 5.4.1 TDCP接口

FitAP SSID

|  |
| --- |
| **request：**  {"method":"get","apmng\_wserv":{"table":"wlan\_serv","para":{"start":0,"end":9}}}  **response：**  {  "apmng\_wserv": {  "wlan\_serv": [{  "wlan\_serv\_0": {  "ssidbrd": "1",  "key\_update\_intv": "86400",  "radius\_ip": "0.0.0.0",  "isolate": "0",  "key": "0",  "bw\_ctrl\_mode": "1",  "default\_bind\_freq": "32639",  "encryption": "0",  "ssid\_code\_type": "1",  "radius\_port": "0",  "up\_limit": "0",  "serv\_id": "0",  "priv\_key": "3",  "cipher": "0",  "bw\_ctrl\_enable": "0",  "down\_limit": "0",  "auto\_bind": "off",  "enable": "on",  "default\_bind\_vlan": "0",  "radius\_acct\_port": "0",  "ssid": "ddddd",  "radius\_pwd": "",  "desc": "",  "ssid\_id": "",  "auth": "3"  }  }],  "count": {  "wlan\_serv": 1  }  },  "error\_code": 0  } |

Fat AP SSID

|  |
| --- |
| **request：**  {"wireless":{"table":"wlan\_serv","filter":[{"radio\_id":"1"}]},"method":"get"}  **response：**  {  "wireless": {  "count": {  "wlan\_serv": 1  },  "wlan\_serv": [{  "wlan\_serv\_1": {  "radio\_id": "1",  "enable": "on",  "ssid": "cddfcfc",  "ssidbrd": "1",  "ssid\_code\_type": "1",  "encryption": "0",  "isolate": "0",  "network\_type": "1",  "wss\_support": "0",  "serv\_id": "1",  "apIdx": "0",  "vlan\_id": "0",  "bw\_ctrl\_enable": "0"  }  }]  },  "error\_code": 0  } |

#### 5.4.2 云端参数对应关系

|  |  |  |
| --- | --- | --- |
| **无线服务参数** | **设备** | **商云** |
|
| 无线服务ID | serv\_id |  |
|
| 无线网络名称 | ssid | ssidName |
|
| 内部隔离 | isolate | staIsolation |
|
| 无线服务描述 | desc | description |
|
| 带宽控制启用禁用 | bw\_ctrl\_enable | bwLimitEnable |
|
| 认证类型 | auth | authType |
|
| 加密算法 | cipher | encryptAlg |
|
| Radius服务器ip | radius\_ip | radiusIp |
|
| Radius服务器端口 | radius\_port | radiusAuthPort |
|
| Radius 计费端口 | radius\_acct\_port | radiusAcctPort |
|
| psk密码 | key | psk |
|
| Radius 密码 | radius\_pwd | radiusPwd |
|
| 组密钥更新周期 | key\_update\_intv | keyUpdateInterval |
|
| 带宽控制模式 | bw\_ctrl\_mode | limitMode |
|
| 最大上行带宽 | up\_limit | upstreamRate |
|
| 最大下行带宽 | down\_limit | downstreamRate |
|
| 当前条目是否启用自动绑定功能 | auto\_bind |  |
|
| 启用自动服务时，默认绑定的频率 | default\_bind\_freq |  |
|
| 启用自动绑定时，默认绑定的vlan | default\_bind\_vlan |  |
|
| SSID\_ID | ssid\_id | wirelessId |
|
| 是否启用 | enable | ssidEnable |
|
| 是否广播（隐藏）ssid | ssidbrd | hideWirelessNetwork |
|
| ssid的编码方式 | ssid\_code\_type | encode |
|
| 安全类型 | encryption | encryptionType |
|
| portal条目的ID |  | portalId |
|
| 是否为访客网络 |  | isGuestNetwork |
|
| 场所ID |  | siteId |
|

#### 5.4.3配置同步流程

整体流程和WEB认证基本一致，生成设备绑定关系时需要记录serv\_id到device\_entry\_id字段，供后续同步portal认证条目使用。

无线路由自身有默认的SSID条目，且默认条目不允许删除，因此云端需要区分出来，目前根据web页面的返回内容无法确认，需要设备端提供一个识别机制。

配置下发流程

①构造下发消息

SetConfigMsg msg = new SetConfigMsg();  
 msg.setSiteId(siteId);  
 msg.setConfigType(type);  
 msg.setModule(ConfigModule.*ssid\_portal*);  
  
 msg.setConfigTarget(target);  
// msg.setConfigTargetType(ConfigTargetType.deviceGroup);  
 msg.setConfigContext(new ConfigContext(context.getCloudUsername(), context.getToken()));  
  
 MsgUtil.*initSetConfigMsgSequence*(siteId, msg);  
 return mqSendClient.sendSetConfigMsg(msg);

### 5.5 Portal页面

#### 5.5.1 TDCP接口

获取portal页面列表

|  |
| --- |
| **request：**  {"method":"get","authentication":{"table":"jump\_page\_list","para":{"start":0,"end":9}}}  **response：**  {  "error\_code": 0,  "authentication": {  "jump\_page\_list": [{  "jump\_page\_list\_1": {  "jump\_page\_mode": "local",  "note": "",  "jump\_page\_name": "ffffffff",  "portalPageId": "00000001",  "reference": "2",  "entry\_id": "1"  }  }],  "count": {  "jump\_page\_list": 1  }  }  } |

获取portal页面详细内容

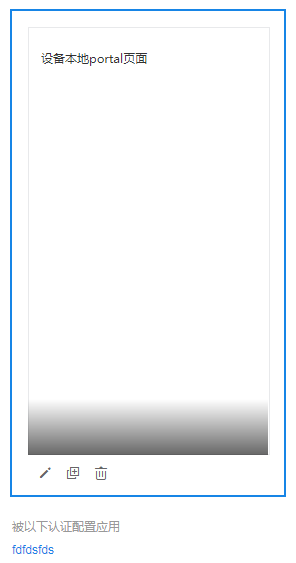
|  |
| --- |
| **request：**  {"method":"get","authentication":{"table":"portal\_page\_list","para":{"start":"0","end":"10"},"filter":[{"portalPageId":"00000001"}]}}  **response：**  {  "error\_code": 0,  "authentication": {  "portal\_page\_list": [{  "portal\_page\_list\_00000001": {  "templateId": 12345678,  "portalPageDesc": "",  "portalPageId": "00000001",  "portalPageName": "",  "config": "%7b%22img\_success\_logo\_img%22%3a%22%22%2c%22auth\_back\_img%22%3a%22background.jpg%22%2c%22confirm%22%3a%22%e4%bf%9d%e5%ad%98%22%2c%22operation%22%3a%22upload%22%2c%22auth\_copyright%22%3a%22Copyright%22%2c%22img\_auth\_back\_img%22%3a%22%22%2c%22auth\_logo\_img%22%3a%22logo.png%22%2c%22auth\_title%22%3a%22%e6%a0%87%e9%a2%98%22%2c%22success\_back\_img%22%3a%22background.jpg%22%2c%22success\_logo\_img%22%3a%22logo.png%22%2c%22auth\_welcome%22%3a%22%e5%85%8d%e8%b4%b9%e6%97%a0%e7%ba%bf%e4%b8%8a%e7%bd%91%22%2c%22success\_welcome%22%3a%22%e6%ac%a2%e8%bf%8e!%22%2c%22img\_auth\_logo\_img%22%3a%22%22%2c%22success\_title%22%3a%22%e8%ae%a4%e8%af%81%e6%88%90%e5%8a%9f%22%2c%22img\_success\_back\_img%22%3a%22%22%7d"  }  }],  "count": {  "portal\_page\_list": 0  }  }  } |

#### 5.5.2 云端参数对应关系

|  |  |  |
| --- | --- | --- |
| **portal页面参数** | **设备** | **商云** |
|
| portal页面的ID | portalPageId | portalPageId |
|
| portal条目名称 | portalPageName | portalPageName |
|
| 支持的认证类型 | portalPageDesc | portalPageDesc |
|
| 模板ID | templateId | templateId |
|
| 场所ID | site\_id | siteId |
|
| 配置信息 |  | config（JSON类型的String） |
|
| 认证标题 | auth\_title | auth\_title |
|
| 欢迎信息 | auth\_welcome | auth\_welcome |
|
| 版权声明 | auth\_copyright | auth\_copyright |
|
| 背景图片名称 | auth\_back\_img | auth\_back\_img |
|
| 认证成功页面标题 | success\_title | success\_title |
|
| 认证成功欢迎信息 | success\_welcome | success\_welcome |
|
| 认证成功logo名称 | success\_logo\_img | success\_logo\_img |
|
| 认证logo图片名称 | auth\_logo\_img | auth\_logo\_img |
|
| 认证成功背景图片 | success\_back\_img | success\_back\_img |
|
| 认证logo图片保存路径 | img\_auth\_logo\_img |  |
|
| 认证成功背景图片保存路径 | img\_success\_back\_img |  |
|
| 确认按钮文字 | confirm |  |
|
| 认证成功logo图片保存路径 | img\_success\_logo\_img |  |
|
| 操作 | operation |  |
|
| 背景图片保存路径 | img\_auth\_back\_img |  |
|
| 是否启用超链接 |  | auth\_hyperlink\_text |
|
| 超链接显示文案 |  | auth\_hyperlink\_address |
|
| 超链接跳转地址 |  | auth\_hyperlink\_enable |
|

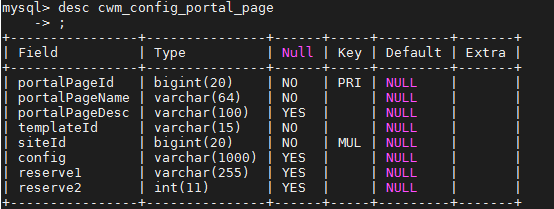
#### 5.5.3 配置同步方案

云端同步portal页面时只同步页面基本信息，不同步详细配置信息。云端展示时表明该页面是设备端页面，基本效果如下。生成绑定关系时需要将jump\_page\_name值保存到device\_entry\_id中供后续同步portal认证使用。



**本地条目识别**

云端针对本地portal页面条目将条目的templateId值固定设置为”-1”，云端根据该字段来判断是否为设备本地条目。云端getPortalPageList和getPortalList接口添加返回deviceLocalPortalPage=true/false字段，以便前端进行判断。



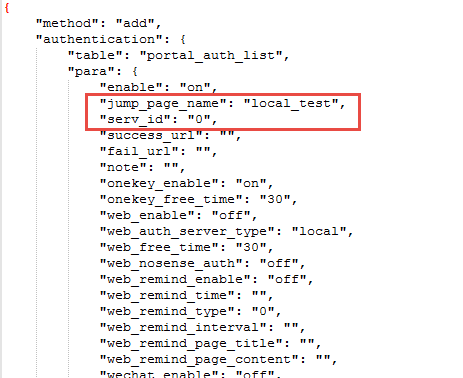
**使用限制**

1. 本地portal页面名称固定为$(mac)\_本地页面\_${name}, 预览方式和云端条目不同，固定为展示静态页面；
2. 本地portal页面不支持编辑、复制。删除操作和云端页面约束条件一致，有portal条目使用时不允许删除；

3. 如果SSID绑定的认证条目使用的是本地portal页面，那么这个SSID不允许绑定到其他设备下的AP射频。绑定时后台自动将其他设备的射频条目过滤掉

### 5.6 Portal认证

设备端的逻辑是Portal认证条目绑定SSID和portal页面，因此云端同步时需要先同步portal页面和SSID配置，最后同步portal认证条目。获取portal认证条目返回的内容中jum\_page\_name对应portal页面条目，serv\_id对应SSID条目ID



#### 5.6.1 TDCP接口

获取portal认证条目

|  |
| --- |
| **request：**  {  "method": "get",  "authentication": {  "table": "portal\_auth\_list",  "para": {  "start": 0,  "end": 9  }  }  }  **response：**  {  "error\_code": 0,  "authentication": {  "portal\_auth\_list": [{  "portal\_auth\_list\_2": {  "onekey\_free\_time": "30",  "remote\_url": "",  "sms\_free\_time": "30",  "wechat\_qrcode\_pic\_url": "",  "sms\_app\_id": "",  "portal\_id": "2",  "wechat\_free\_time": "30",  "note": "",  "web\_remind\_time": "1",  "sms\_enable": "off",  "fail\_url": "",  "cmcc\_url": "",  "wechat\_ssid": "",  "cmcc\_nosense\_auth": "",  "web\_auth\_server\_type": "remote",  "wechat\_secret\_key": "",  "remote\_nosense\_auth": "",  "web\_auth\_server\_group": "test",  "serv\_id": "0",  "wechat\_app\_id": "",  "cmcc\_enable": "",  "entry\_id": "2",  "wechat\_token\_server": "",  "remote\_auth\_server\_group": "",  "sms\_http\_template": "",  "web\_remind\_page\_title": "",  "web\_enable": "on",  "web\_remind\_enable": "off",  "sms\_channel\_type": "alibaba\_cloud",  "sms\_username": "",  "wechat\_force\_concern": "off",  "fail\_id": "10004",  "web\_free\_time": "30",  "wechat\_app\_secret": "",  "sms\_template\_id": "",  "jump\_page\_name": "ffffffff",  "enable": "on",  "web\_nosense\_auth": "off",  "default\_page\_type": "",  "sms\_http\_url": "",  "web\_remind\_type": "0",  "interface": "",  "remote\_enable": "",  "wechat\_enable": "off",  "sms\_http\_code\_type": "1",  "cmcc\_ac\_name": "",  "cmcc\_auth\_server\_group": "",  "web\_remind\_page\_content": "",  "sms\_http\_mode": "post",  "cmcc\_free\_time": "30",  "web\_remind\_interval": "30",  "onekey\_enable": "on",  "remote\_free\_time": "30",  "wechat\_qrcode": "",  "sms\_auth\_expiry": "1",  "success\_url": "",  "wechat\_shop\_id": "",  "sms\_password": "",  "remote\_auth\_server\_type": "",  "cmcc\_auth\_server\_type": "",  "vlan\_id": "0",  "is\_cloud": "n",  "auth\_type": "5",  "sms\_signname": ""  }  }],  "count": {  "portal\_auth\_list": 2  }  }  } |

获取认证服务器组信息

|  |
| --- |
| **request：**  {  "method": "get",  "authentication": {  "table": "server\_group\_list"  }  }  **response：**  {  "error\_code": 0,  "authentication": {  "count": {  "server\_group\_list": 1  },  "server\_group\_list": [{  "server\_group\_list\_1": {  "recovery\_time": "30",  "backup\_server": "",  "entry\_id": "1",  "main\_server": "test",  "note": "",  "server\_type": "radius",  "reference": "1",  "group\_name": "test"  }  }]  }  } |

获取认证服务器信息

|  |
| --- |
| **request：**  {"method":"get","authentication":{"table":"radius\_list","para":{"start":0,"end":9}}}  **response：**  {  "error\_code": 0,  "authentication": {  "radius\_list": [{  "radius\_list\_1": {  "radius\_auth\_type": "pap",  "auth\_port": "1812",  "reference": "1",  "nas\_ip": "",  "account\_port": "0",  "share\_key": "123456",  "server\_addr": "1.11.1.1",  "entry\_id": "1",  "retry\_interval": "3",  "retry\_count": "3",  "server\_name": "test"  }  }],  "count": {  "radius\_list": 1  }  }  } |

#### 5.6.2 配置同步流程

整体流程基本一致，构造云端条目有一定的特殊处理，具体如下。



#### 5.6.3 组合认证支持

设备目前支持组合认证方式，但是云端只支持单认证方式，因此云端需要新增组合认证支持。

设备端需要在capability中声明自己支持组合认证，以便云端能够正确识别，具体声明方式请设备端确定。云端绑定SSID到设备时，如果SSID带portal且portal是组合认证，那么该条目不允许绑定到不支持组合认证的设备。

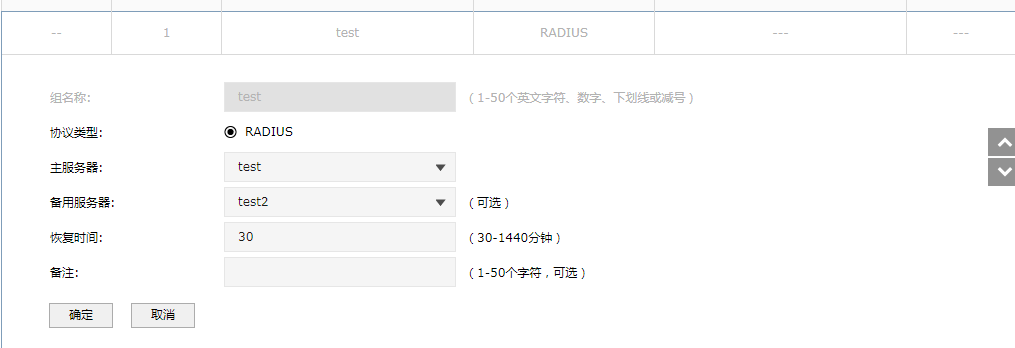
#### 5.6.4 CMCC+远程portal支持

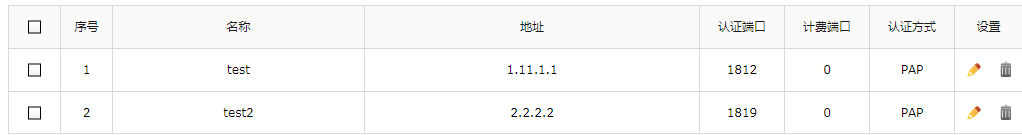
待定。

#### 5.6.5补充

设备端的web认证服务器是以条目形式添加并绑定到portal认证条目的，一个条目绑定的主服务器和备服务器配置可以不同，但是云端配置下主服务器和备服务器除地址外，其他配置都要求相同。







web认证时设备端主备认证服务器配置可以不一致，云端只处理主服务器信息，备服务器留空；

### 5.7 射频

射频配置目前AC/路由已经是拉取的，FAT AP 直接沿用现有的拉取方案即可，TDCP接口复用设备端的接口。

当前拉取同步射频配置时如果拉取失败会重试3次，如果3次都失败，则会在云端生成默认条目并覆盖设备端配置。如果设备支持双向同步，3次都拉取失败后不允许AP上线，让设备重新上报add ap再次进行配置同步；拉取射频失败后，云端生成默认条目，但是不下发到设备。

### 5.8 无线服务绑定关系

FAT AP和WVR自身的无线服务在同步SSID时进行初始化， Fit AP的绑定关系在添加时进行初始化。

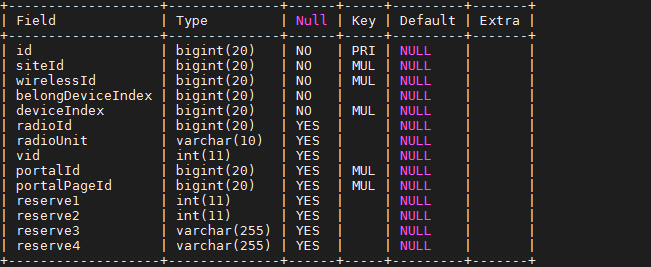
#### 5.8.1 TDCP接口

|  |
| --- |
| **request：**  {"method":"get","apmng\_wserv":{"table":"radio\_bind\_list","filter":[{"group\_id":"0","serv\_id":"0"}],"para":{"start":0,"end":9}}}  **response：**  {  "apmng\_wserv": {  "radio\_bind\_list": [{  "radio\_bind\_list\_0": {  "radio\_id": "160001",  "bind\_status": "1",  "ap\_type": "template",  "freq\_name": "2.4GHz",  "vlan\_id": "0",  "freq\_unit": "1",  "freq\_mode": "6",  "ap\_name": "aaaaa"  }  }],  "count": {  "radio\_bind\_list": 1  }  },  "error\_code": 0  } |

设备端需要支持根据radio\_id参数批量获取绑定关系

#### 5.8.2 Fit AP绑定关系同步

5.8.2.1云端绑定关系字段信息



5.8.2.2 同步流程



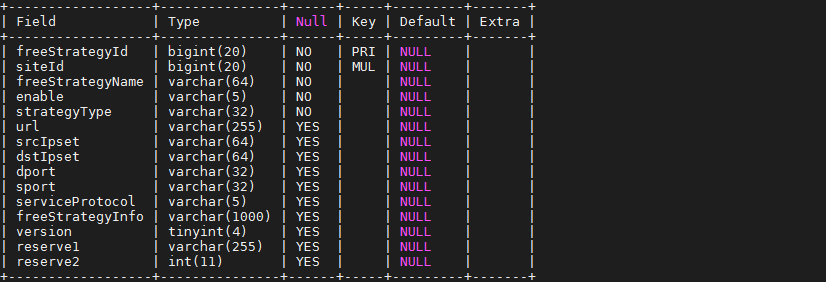
## 6. 云端兼容

主要需要考虑免认证、web认证账号和认证参数的兼容。

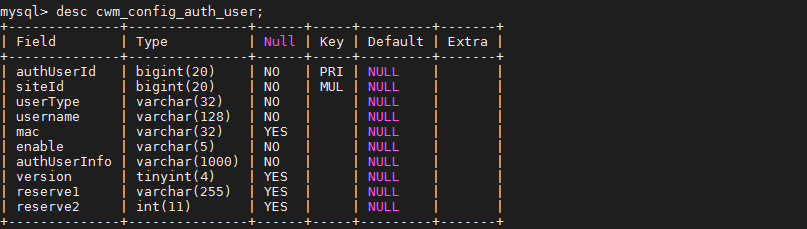
### 6.1 配置版本识别

云端相关数据表设计如下：

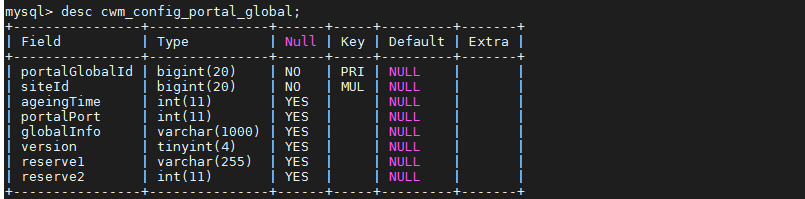
免认证策略：



web认证用户：



认证参数：



表中都带有version字段，目前没有使用。增加同步方案后新增或者编辑条目时都将version字段修改为2，以便区分同步时是采用新方案还是原来旧方案。

### 6.2 旧设备配置同步流程

以免认证策略为例，设备上线配置同步配置时需要查找所有version为null的条目，这些条目都需要同步到设备；然后查找所有version=2的条目，针对version=2的条目，从配置绑定关系表中查找自己的对应关系，如果存在绑定关系，那么需要下发到设备，否则忽略该条目。

### 6.3云端接口调整

以免认证策略为例，云端需要新增获取绑定设备列表接口和绑定设备接口，初步定义如下：

#### 6.3.1获取绑定设备列表接口

|  |  |  |
| --- | --- | --- |
| 名称 | 内容 | |
| 方向 | WEB🡪商云 | |
| 请求路径 | /tums/wireless/v2/getFreeStrategyBindDeviceList | |
| 请求参数(params)  （params最大长度2k） | freeStrategyId  string | 查找条目的ID  限制：可选  添加条目时设置为"",获取可以绑定的设备列表  编辑条目时必传，用户填充已绑定设备信息 |
| projectId  string | 项目ID  限制：必选 |
| 响应内容(result) | bindDeviceList  object | 符合条件的设备列表信息 |
| total  int | 符合条件的条目总数 |
| Error Code | -20104 | 参数不存在 |
| 使用限制 | WEB/APP成功登录后 | |
| 功能描述 | 向商云请求获取绑定设备列表信息 | |

bindDeviceList定义如下：

siteList内容定义

|  |  |
| --- | --- |
| 字段 | 内容 |
| deviceIndex  String | 必需。设备ID，对应设备表中的ID信息。 |
| deviceName  string | 必需。设备名称。 |
| bindStatus  string | 必选。设备是否已经绑定到该条目。  “bind” : 已绑定  “unbind“: 未绑定 |

示例：

|  |
| --- |
| **request：**  **POST**  **url:** <https://xxxx.xxxx.xxxx/tums/wireless/v2/getFreeStrategyBindDeviceList>  **HTTP/1.1**  **Content-Type: application/json;charset=UTF-8**  {  "projectId":"123456",  "freeStrategyId":"123456"  }  **response：**  **HTTP/1.1 200 OK**  {  "errorCode":0,  "result":{  "bindDeviceList":[  {  "deviceIndex":"11111",  "deviceName":"AC500",  "bindStatus":"bind"  },{  "deviceIndex":"222222",  "deviceName":"AC1000",  "bindStatus":"unbind"  },  ],  "total":2  }  } |

#### 6.3.2 绑定到设备接口

|  |  |  |
| --- | --- | --- |
| 名称 | 内容 | |
| 方向 | WEB🡪商云 | |
| 请求路径 | /tums/wireless/v2/bindFreeStrategy | |
| 请求参数(params)  （params最大长度2k） | freeStrategyId  string | 查找条目的ID  限制：可选  添加条目时设置为"",获取可以绑定的设备列表  编辑条目时必传，用户填充已绑定设备信息 |
| projectId  string | 项目ID  限制：必选 |
|  | bindDeviceList  list<String> | 绑定的设备列表ID  限制：必选，当无条目时传空数组 |
|  | unbindDeviceList  list<string> | 解绑的设备列表ID  限制：必选，唔条目时传空数组 |
| 响应内容(result) | error\_code  object | 符合条件的设备列表信息 |
| Error Code |  | 参数不存在 |
| 使用限制 | WEB/APP成功登录后 | |
| 功能描述 | 请求绑定免认证策略到设备 | |

示例：

|  |
| --- |
| **request：**  **POST**  **url:** <https://xxxx.xxxx.xxxx/tums/wireless/v2/bindFreeStrategy>  **HTTP/1.1**  **Content-Type: application/json;charset=UTF-8**  {  "projectId":"123456",  "freeStrategyId":"123456",  "bindDeviceList":["123456", "123455"],  "unbindDeviceList":["333333", "444444"],  }  **response：**  **HTTP/1.1 200 OK**  {  "errorCode":0,  } |