**策略imei修改设计**

* **文件上传VFS**

将imei文件上传，解析imei列表

* **数据库操作**

将策略ID，类型, 数据，版本号保存至数据库**t\_publish\_imei\_version**

如果是文件上传方式，则**type**为file，data为文件名

如果是手工数据方式，则**type**为insert，data为imei列表，逗号分隔

每次修改strategy，版本号**version**加1

**cache\_cleaned**表示记录对应的缓存数据是否已被清理

**create\_time**

**update\_time**

数据库例子：

**strategy\_id, type data, version cache\_cleaned**

strategy\_123, file aa/bb/123.txt, 1 1

strategy\_123, flie 12345.txt 2 0

strategy\_456 insert imei444, imei555, imei666 1 0

* **Redis操作**

将每一个策略ID，版本号，imei组装为key，保存至redis

Key例子：

SIP:strategy\_123:1:imei333

SIP:strategy\_123:1:imei444

…

**旧版本定时删除**

1. 定时从t\_publish\_imei\_version中获取每个策略非最大version，且cache\_cleaned=0的记录
2. 获取记录对应的所有imei号（type为file则从vfs下载文件，type为insert则从data中获取）
3. 将strategy\_id, version, imei组装为key，从redis中删除对应的key，删除完成后，cache\_cleaned更新为1
4. 从vfs删除对应文件

**api轮询**

获取所有已发布的策略

对每个策略，从t\_publish\_imei\_version获取对应strategy\_id的最新版本（本地缓存处理）

将策略ID，版本号，imei组装为key，从redis中查询是否存在