# kms接口文档

架构图：

Kms（C端）

p端

1,2,3,4,5接口

保存产线公钥，设备公钥，证书，私钥 加密，解密，签名，密钥产生

Db

CA（签发证书）

Keystore

## 产线公钥导入云端。

Url: kms/importProdLinePubKey  
params:

{

id: "0x12345678"--HSM的硬件唯一性ID，由“产线负责人负责完善详细格式。”

rsaPublickey： "0x1234....."

type: "dongle"

oem: "Siemens"

createTime: "20190909121315"

signature: "0x1234...."

}

Response: {errorCode:0,message:”成功”,body:”null”}，其他errorCode失败

## C端公钥导入到P端

URL：kms/downLoadCloudPubKey

Params:无

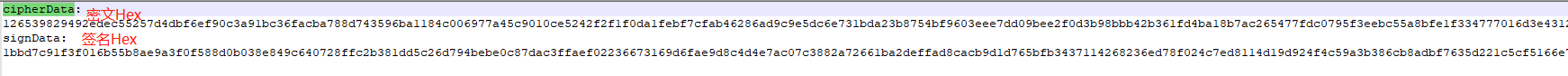
Response：{errorCode:0,message:”成功”,body:”云端公钥Base64String”}

## C端将APPKeyRootKey-SYM和SBKEK，RKTHHex导入P端

URL：kms/downLoadAppRootKeyAndSBKEKAndRKTH

Params:{c2pPath: “密钥文件的名字”}

响应结构：



C2P-KeyDataCipherFile {

CipherData： rsa2048-oaep(C2P-KeyDataPlainFile.bytes()) with P-PubKey

SignData: "0x1234...." -- rsa 2048 PSS{ chpher text of oaep } With C-PriKey

}

C2P-KeyDataPlainFile ：

{

Pid: "12345678"--P加密机的ID   
 SBKEK："0x1234....." 32字节

APPKeyRootKey-SYM： "0x1234....." -- 32字节

RKTHHex： "0x1234....." --32字节的字符串表示，见《LPC55S6x User manual》7.3.2.1 p149

type: "dongle"

oem: "Siemens"

createTime: "20190909121315" 14字节

}

## P端设备列表导入C端，并生成M端密钥和证书

Url：/kms/importDevSnList  
param:file

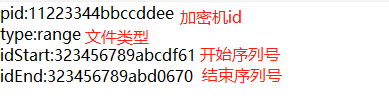
Response: {errorCode:0,message:”导入成功”,body:”null”}：  
文件格式：以前定的。

设备列表（每个8字节）

Oem（2字节）

设备数量（4字节）

导入文件结构：

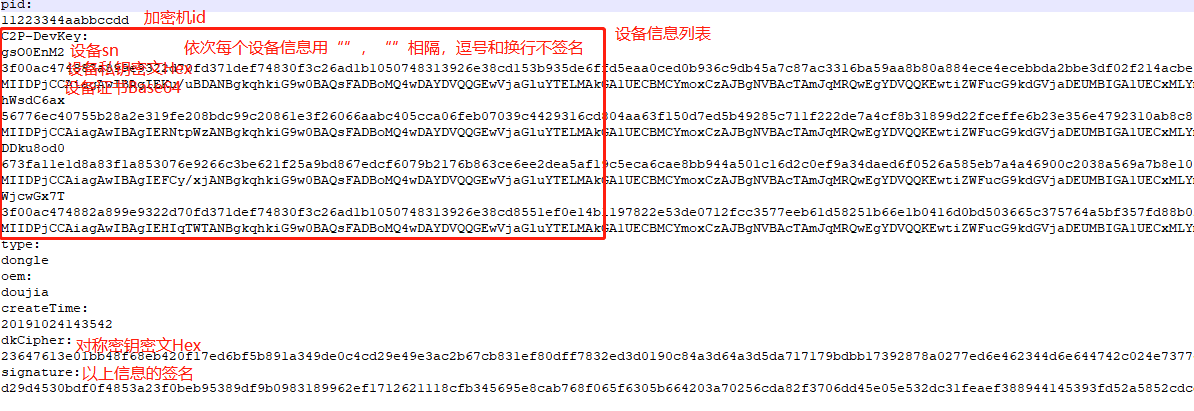


文件类型为”range”,设备序列号为连续的 idStart(开始序列号)<=设备列表<=idEnd(结束序列号)。



文件类型为disperse，设备序列号为不连续散列的。

## C端导入M端密钥和证书到P端



C2P-DevKey

{

SN：。。。。

DevPriKey：。。。encrypt with DK, AES-CBC-Nopadding

DevCert：。。。

}

C2P-DevKeyList {

Pid: "12345678"--P加密机的ID

C2P-DevKey:[“dev1Data”,”dev2Data”,......],

type: "dongle",

oem: "Siemens",

createTime: "20190909121315",

dkCipher: rsa2048-oaep(dk) with P-PubKey,

signature: "0x1234...." -- rsa 2048 PSS{ ........ } With C-PriKey

}

证书文件结构：

第1行：11223344bbccddee（加密机id）。

第2行：100（设备证书的数量）

第3行：第3-102行是设备信息，像这样devSn,dvePriKey,devCert用逗号分割，第一部分为devSn（设备序列号），第二部分dvePriKey（设备私钥），第三部分为devCert设备证书。

323456789abcdf61,,

第4行：dongle（设备类型）

第5行：oem（厂商）

第6行：20191104171634（创建时间）

第7行：密文（加密对称密钥）

第8行：签名。