云密钥安全管理系统

接口文档（web接口）

目录

[接口文档（web接口） 1](#_Toc25739666)

[1. 术语说明 11](#_Toc25739667)

[2. 证书用户模块 11](#_Toc25739668)

[2.1. 用户证书查询 11](#_Toc25739671)

[请求信息 11](#_Toc25739672)

[请求结构说明 12](#_Toc25739673)

[响应结构说明 12](#_Toc25739674)

[请求响应简单示例 12](#_Toc25739675)

[2.2. 身份预验证 13](#_Toc25739676)

[请求信息 13](#_Toc25739677)

[请求结构说明 13](#_Toc25739678)

[响应结构说明 14](#_Toc25739679)

[请求响应简单示例 14](#_Toc25739680)

[2.3. 身份预验证之短信验证码校验 15](#_Toc25739681)

[请求信息 15](#_Toc25739682)

[请求结构说明 16](#_Toc25739683)

[响应结构 16](#_Toc25739684)

[请求响应简单示例 16](#_Toc25739685)

[2.4. 证书用户口令重试次数 17](#_Toc25739686)

[请求信息 17](#_Toc25739687)

[请求结构说明 17](#_Toc25739688)

[响应结构说明 17](#_Toc25739689)

[请求响应简单示例 17](#_Toc25739690)

[2.5. 查询授权有效性配置 18](#_Toc25739691)

[请求信息 18](#_Toc25739692)

[请求结构说明 18](#_Toc25739693)

[响应结构说明 18](#_Toc25739694)

[请求响应简单示例 19](#_Toc25739695)

[3. 授权管理模块 20](#_Toc25739696)

[3.1 查询已被授权信息 20](#_Toc25739700)

[请求信息 20](#_Toc25739701)

[请求结构说明 20](#_Toc25739702)

[响应结构说明 20](#_Toc25739703)

[请求响应简单示例 20](#_Toc25739704)

[3.2 用户分页查询授权记录 21](#_Toc25739705)

[请求信息 21](#_Toc25739706)

[请求结构说明 22](#_Toc25739707)

[响应结构说明 22](#_Toc25739708)

[请求响应简单示例 23](#_Toc25739709)

[3.3 用户授权状态变更 24](#_Toc25739710)

[请求信息 24](#_Toc25739711)

[请求结构说明 24](#_Toc25739712)

[响应结构说明 24](#_Toc25739713)

[请求响应简单示例 25](#_Toc25739714)

[3.4 被授权用户可用证书查询 25](#_Toc25739715)

[请求信息 26](#_Toc25739716)

[请求结构说明 26](#_Toc25739717)

[响应结构说明 26](#_Toc25739718)

[请求响应简单示例 26](#_Toc25739719)

[3.5 授权申请 27](#_Toc25739720)

[请求信息 27](#_Toc25739721)

[请求结构说明 27](#_Toc25739722)

[响应结构说明 28](#_Toc25739723)

[请求响应简单示例 28](#_Toc25739724)

[3.6 授权有效性修改 29](#_Toc25739725)

[请求信息 29](#_Toc25739726)

[请求结构说明 29](#_Toc25739727)

[响应结构说明 30](#_Toc25739728)

[请求响应简单示例 30](#_Toc25739729)

[4. 被授权用户身份验证模块 31](#_Toc25739730)

[4.1. 被授权用户获取口令PIN重试次数 31](#_Toc25739735)

[请求信息 31](#_Toc25739736)

[请求结构说明 32](#_Toc25739737)

[响应结构说明 32](#_Toc25739738)

[请求响应简单示例 32](#_Toc25739739)

[4.2. 被授权用户口令PIN认证 32](#_Toc25739740)

[请求信息 32](#_Toc25739741)

[请求结构说明 33](#_Toc25739742)

[响应结构说明 33](#_Toc25739743)

[请求响应简单示例 33](#_Toc25739744)

[4.3. 口令PIN+短信验证码验证 34](#_Toc25739745)

[获取短信验证码 34](#_Toc25739746)

[验证短信验证码 36](#_Toc25739747)

[5. 身份验证流程 37](#_Toc25739748)

[5.1. 身份验证类型常量(loginTypeBitValue)说明 37](#_Toc25739750)

[5.2. 二维码内容 38](#_Toc25739751)

[5.3. 客户端获取认证二维码 39](#_Toc25739752)

[请求信息 39](#_Toc25739753)

[请求结构说明 39](#_Toc25739754)

[响应结构说明 39](#_Toc25739755)

[请求响应简单示例 40](#_Toc25739756)

[5.4. 手机扫描客户端二维码 40](#_Toc25739757)

[5.5. 用户人脸认证 40](#_Toc25739758)

[请求信息 41](#_Toc25739759)

[请求结构说明 41](#_Toc25739760)

[响应结构说明 41](#_Toc25739761)

[请求响应简单示例 42](#_Toc25739762)

[5.6. 检查手机设备是否能进行身份验证 43](#_Toc25739763)

[请求信息 43](#_Toc25739764)

[请求结构说明 44](#_Toc25739765)

[响应结构说明 44](#_Toc25739766)

[请求响应简单示例 44](#_Toc25739767)

[5.7. 手机提交口令身份验证接口 45](#_Toc25739768)

[请求信息 45](#_Toc25739769)

[请求结构说明 45](#_Toc25739770)

[响应结构说明 45](#_Toc25739771)

[请求响应简单示例 46](#_Toc25739772)

[5.8. 手机提交口令指纹身份验证接口 46](#_Toc25739773)

[请求信息 46](#_Toc25739774)

[请求结构说明 47](#_Toc25739775)

[响应结构说明 47](#_Toc25739776)

[请求响应简单示例 47](#_Toc25739777)

[5.9. 手机提交口令设备人脸身份验证接口 48](#_Toc25739778)

[请求信息 48](#_Toc25739779)

[请求结构说明 49](#_Toc25739780)

[响应结构说明 49](#_Toc25739781)

[请求响应简单示例 49](#_Toc25739782)

[5.10. 客户端轮询二维码验证结果接口 50](#_Toc25739783)

[6. 绑定流程 50](#_Toc25739784)

[6.1. 获取支持绑定的身份验证类型 50](#_Toc25739786)

[请求信息 50](#_Toc25739787)

[请求结构说明 50](#_Toc25739788)

[响应结构说明 50](#_Toc25739789)

[请求响应简单示例 51](#_Toc25739790)

[6.2. 客户端获取绑定二维码 51](#_Toc25739791)

[请求信息 51](#_Toc25739792)

[请求结构说明 52](#_Toc25739793)

[响应结构说明 52](#_Toc25739794)

[请求响应简单示例 52](#_Toc25739795)

[6.3. 绑定人脸数据 53](#_Toc25739796)

[请求信息 53](#_Toc25739797)

[请求结构说明 53](#_Toc25739798)

[响应结构 54](#_Toc25739799)

[请求响应简单示例 54](#_Toc25739800)

[6.4. 检查用户设备是否已经绑定 55](#_Toc25739801)

[请求信息 55](#_Toc25739802)

[请求结构说明 55](#_Toc25739803)

[响应结构说明 55](#_Toc25739804)

[请求响应简单示例 56](#_Toc25739805)

[6.5. 绑定手机设备信息 56](#_Toc25739806)

[请求信息 56](#_Toc25739807)

[请求结构说明 57](#_Toc25739808)

[响应结构说明 57](#_Toc25739809)

[请求响应简单示例 57](#_Toc25739810)

[6.6. 绑定手机设备指纹信息 58](#_Toc25739811)

[请求信息 58](#_Toc25739812)

[请求结构说明 58](#_Toc25739813)

[响应结构说明 59](#_Toc25739814)

[请求响应简单示例 59](#_Toc25739815)

[6.7. 绑定手机设备人脸信息 60](#_Toc25739816)

[请求信息 60](#_Toc25739817)

[请求结构说明 60](#_Toc25739818)

[响应结构说明 61](#_Toc25739819)

[请求响应简单示例 61](#_Toc25739820)

[6.8. 客户端轮询二维码验证结果接口 62](#_Toc25739821)

[7. 登录消息推送（该章节接口未实现） 62](#_Toc25739822)

[7.1. MQTT消息体定义 62](#_Toc25739824)

[7.2. 安卓接入 62](#_Toc25739825)

[7.3. IOS接入 62](#_Toc25739826)

[7.4. 申请登录 63](#_Toc25739827)

[请求消息 63](#_Toc25739828)

[请求结构说明 63](#_Toc25739829)

[响应结构说明 63](#_Toc25739830)

[请求响应简单示例 63](#_Toc25739831)

[7.5. 查询手机登录结果 64](#_Toc25739832)

[请求消息 64](#_Toc25739833)

[请求结构说明 64](#_Toc25739834)

[响应结构说明 65](#_Toc25739835)

[请求响应简单示例 65](#_Toc25739836)

[7.6. 注册推送目标app 66](#_Toc25739837)

[请求消息 66](#_Toc25739838)

[请求结构说明 66](#_Toc25739839)

[响应结构说明 66](#_Toc25739840)

[请求响应简单示例 67](#_Toc25739841)

[7.7. 注销推送目标app 68](#_Toc25739842)

[请求消息 68](#_Toc25739843)

[请求结构说明 68](#_Toc25739844)

[响应结构说明 68](#_Toc25739845)

[请求响应简单示例 69](#_Toc25739846)

[7.8. 确认登录 69](#_Toc25739847)

[请求消息 70](#_Toc25739848)

[请求结构说明 70](#_Toc25739849)

[响应结构说明 70](#_Toc25739850)

[请求响应简单示例 70](#_Toc25739851)

[8. 外部系统接口 71](#_Toc25739852)

[8.1. 机构信息注册 71](#_Toc25739854)

[请求消息 71](#_Toc25739855)

[请求结构说明 71](#_Toc25739856)

[响应结构说明 72](#_Toc25739857)

[请求响应简单示例 72](#_Toc25739858)

[8.2. 用户信息注册 73](#_Toc25739859)

[请求消息 73](#_Toc25739860)

[请求结构说明 74](#_Toc25739861)

[响应结构说明 74](#_Toc25739862)

[请求响应简单示例 74](#_Toc25739863)

[8.3. 业务通知 75](#_Toc25739864)

[请求消息 76](#_Toc25739865)

[请求结构说明 76](#_Toc25739866)

[响应结构说明 84](#_Toc25739867)

[请求响应简单示例 84](#_Toc25739868)

[9. 公共接口 85](#_Toc25739869)

[9.1. 修改用户口令 85](#_Toc25739871)

[请求信息 85](#_Toc25739872)

[请求结构说明 85](#_Toc25739873)

[响应结构说明 86](#_Toc25739874)

[请求响应简单示例 86](#_Toc25739875)

[9.2. 获取证书所支持的身份验证类型 86](#_Toc25739876)

[请求信息 86](#_Toc25739877)

[请求结构说明 87](#_Toc25739878)

[响应结构说明 87](#_Toc25739879)

[请求响应简单示例 87](#_Toc25739880)

[9.3. 轮询二维码验证结果 87](#_Toc25739881)

[请求信息 88](#_Toc25739882)

[请求结构说明 88](#_Toc25739883)

[响应结构说明 88](#_Toc25739884)

[请求响应简单示例 88](#_Toc25739885)

[9.4. 获取用户已经支持（开通）的身份验证类型 89](#_Toc25739886)

[请求信息 89](#_Toc25739887)

[请求结构说明 89](#_Toc25739888)

[响应结构说明 90](#_Toc25739889)

[请求响应简单示例 90](#_Toc25739890)

[9.5. 根据绑定的二维码信息获取用户信息 91](#_Toc25739891)

[请求信息 91](#_Toc25739892)

[请求结构说明 91](#_Toc25739893)

[响应结构说明 91](#_Toc25739894)

[请求响应简单示例 91](#_Toc25739895)

[9.6. 根据认证二维码信息获取用户UID 92](#_Toc25739896)

[请求信息 92](#_Toc25739897)

[请求结构说明 93](#_Toc25739898)

[响应结构说明 93](#_Toc25739899)

[请求响应简单示例 93](#_Toc25739900)

[9.7. 电子签名 94](#_Toc25739901)

[请求信息 94](#_Toc25739902)

[请求结构说明 94](#_Toc25739903)

[响应结构说明 94](#_Toc25739904)

[请求响应简单示例 94](#_Toc25739905)

[9.8. 电子P7签名 95](#_Toc25739906)

[请求信息 95](#_Toc25739907)

[请求结构说明 95](#_Toc25739908)

[响应结构说明 96](#_Toc25739909)

[请求响应简单示例 96](#_Toc25739910)

[9.9. P7验签 97](#_Toc25739911)

[请求信息 97](#_Toc25739912)

[请求结构说明 97](#_Toc25739913)

[响应结构说明 98](#_Toc25739914)

[请求响应简单示例 98](#_Toc25739915)

[9.10. 查询支持的Hash算法 99](#_Toc25739916)

[请求信息 99](#_Toc25739917)

[请求结构说明 100](#_Toc25739918)

[响应结构说明 100](#_Toc25739919)

[请求响应简单示例 100](#_Toc25739920)

[9.11. 查询系统版本 100](#_Toc25739921)

[请求信息 101](#_Toc25739922)

[请求结构说明 101](#_Toc25739923)

[响应结构说明 101](#_Toc25739924)

[请求响应简单示例 101](#_Toc25739925)

[9.12. 查询密码重试次数 101](#_Toc25739926)

[请求信息 102](#_Toc25739927)

[请求结构说明 102](#_Toc25739928)

[响应结构说明 102](#_Toc25739929)

[请求响应简单示例 102](#_Toc25739930)

[9.13. 查询登录令牌是否有效 103](#_Toc25739931)

[请求信息 103](#_Toc25739932)

[请求结构说明 103](#_Toc25739933)

[响应结构说明 103](#_Toc25739934)

[请求响应简单示例 104](#_Toc25739935)

[9.14. 根据登录令牌查询操作者信息 104](#_Toc25739936)

[请求信息 104](#_Toc25739937)

[请求结构说明 104](#_Toc25739938)

[响应结构说明 105](#_Toc25739939)

[请求响应简单示例 105](#_Toc25739940)

[9.15. 查询用户的登录信息 106](#_Toc25739941)

[请求信息 106](#_Toc25739942)

[请求结构说明 106](#_Toc25739943)

[响应结构说明 106](#_Toc25739944)

[请求响应简单示例 107](#_Toc25739945)

[10. 签章接口 107](#_Toc25739946)

[10.1. 下载签章图片 107](#_Toc25739948)

[请求信息 108](#_Toc25739949)

[请求结构说明 108](#_Toc25739950)

[响应结构说明 108](#_Toc25739951)

[请求响应简单示例 108](#_Toc25739952)

[10.2. 上传签章图片 109](#_Toc25739953)

[请求信息 109](#_Toc25739954)

[请求结构说明 109](#_Toc25739955)

[响应结构说明 109](#_Toc25739956)

[请求响应简单示例 109](#_Toc25739957)

[10.3. PDF同步签章 110](#_Toc25739958)

[请求信息 110](#_Toc25739959)

[请求结构说明 110](#_Toc25739960)

[响应结构说明 111](#_Toc25739961)

[请求响应简单示例 112](#_Toc25739962)

[11. 退出登录 113](#_Toc25739963)

[11.1. 通过设备获取退出令牌 113](#_Toc25739965)

[请求信息 113](#_Toc25739966)

[请求结构说明 113](#_Toc25739967)

[响应结构说明 113](#_Toc25739968)

[请求响应简单示例 114](#_Toc25739969)

[11.2. 人脸认证获取退出令牌 114](#_Toc25739970)

[请求信息 114](#_Toc25739971)

[请求结构说明 115](#_Toc25739972)

[响应结构说明 115](#_Toc25739973)

[请求响应简单示例 116](#_Toc25739974)

[11.3. 退出所有应用登录 116](#_Toc25739975)

[请求信息 116](#_Toc25739976)

[请求结构说明 117](#_Toc25739977)

[响应结构说明 117](#_Toc25739978)

[请求响应简单示例 117](#_Toc25739979)

[11.4. 退出指定应用登录 118](#_Toc25739980)

[请求信息 118](#_Toc25739981)

[请求结构说明 118](#_Toc25739982)

[响应结构说明 118](#_Toc25739983)

[请求响应简单示例 118](#_Toc25739984)

[12. 接口附加说明 119](#_Toc25739985)

[12.1. 请求响应结构说明 119](#_Toc25739987)

[12.1.1. Cert结构 119](#_Toc25739988)

[12.1.2. 响应结构 119](#_Toc25739989)

[12.1.3. User结构 119](#_Toc25739990)

[12.1.4. Organization结构 120](#_Toc25739991)

[12.1.5. Linkman结构 120](#_Toc25739992)

[12.1.6. DeviceInfo结构 121](#_Toc25739993)

[12.2. 接口参数常量说明 121](#_Toc25739994)

[12.2.1. Hash算法 121](#_Toc25739995)

[12.2.2. 签名算法 122](#_Toc25739996)

[12.2.3. 机构证件类型 124](#_Toc25739997)

[12.2.4. 个人证件类型 125](#_Toc25739998)

[12.2.5. 机构类型 125](#_Toc25739999)

[12.2.6. 国家、省份、城市常量取值 125](#_Toc25740000)

[12.2.7. 业务状态常量 126](#_Toc25740001)

[12.2.8. 性别 126](#_Toc25740002)

[12.2.9. 响应状态码常量status值 126](#_Toc25740003)

[12.3. 安全性附加说明 127](#_Toc25740004)

[12.3.1. 通信的安全 127](#_Toc25740005)

[12.3.2. 时间格式说明 127](#_Toc25740006)

[12.3.3. 管理员签名说明 127](#_Toc25740007)

# 文件修改记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **段落** | **修改者** | **主要修改内容说明** |
| 201909010 |  |  | 新增接口，在1.8实现以下接口  公共接口：查询登录令牌是否有效  公共接口：根据登录令牌查询操作者信息  公共接口：查询用户的登录信息  退出登录：根据设备获取用于退出令牌  退出登录：人脸认证获取用于退出令牌  退出登录：退出所有应用登录  退出登录：退出指定应用登录 |
| 20190916 |  |  | 补充[响应状态码常量](#_响应编码)，-6 至 -20的特殊失败响应码 |
| 20191022 |  |  | 在1.10实现以下接口  证书用户模块：查询授权有效性配置  授权管理模块：授权申请  授权管理模块：授权有效性修改  授权管理模块：用户授权状态变更  在1.10修改以下接口：  授权管理模块：用户分页查询授权记录  公共接口：根据登录令牌查询操作者信息  在1.10删除以下接口：  授权管理模块：用户授权  授权管理模块：用户重新授权  授权管理模块：用户取消授权 |
| 20191024 |  |  | [PDF同步签章接口](#_请求结构说明)增加默认参数 showSeal  用于控制PDF是否显示签章 |
| 20191115 |  |  | 在1.11实现以下接口  外部系统接口：机构信息注册  外部系统接口：用户信息注册  外部系统接口：业务通知 |
| 20191202 |  |  | 在1.11新增[下载指定格式签章图片](#_下载指定格式签章图片)的接口 |
| 20191220 |  |  | 在1.12实现以下接口  登录消息推送：申请登录  登录消息推送：查询手机登录结果  登录消息推送：注册推送目标app  登录消息推送：注销推送目标app  登录消息推送：注销推送目标app（单个）  登录消息推送：确认登录  登录消息推送：消息通知确认  相关响应增加二维码内容：见[客户端获取认证二维码响应](#_响应结构说明_5)、[客户端获取绑定二维码响应](#_响应结构说明_6) |
| 20200228 |  |  | 在1.13实现以下接口：  证书用户模块：解析证书属性及扩展（仅实现，实际在2.0版本测试）  签章接口：PDF签章（使用参数中的图片做签章）  签章接口：PDF验证 |
| 20200309 |  |  | 在2.1实现以下接口  证书用户模块：用户证书查询（uid+项目uid查询，仅实现，实际在2.2版本测试）  身份验证流程：客户端获取认证二维码（增加type参数）  身份验证流程：绑定认证二维码的状态 |
| 20200409 |  |  | 在2.2实现以下接口  公共接口：查询证书所属项目uid（projectUid）  签章接口：根据UID下载签章图片（2.1实现，2.2测试） |
| 20200518 |  |  | 在2.3实现以下接口  公共接口：获取图形验证码  公共接口：自助解锁获取短信验证码接口  公共接口：自助解锁验证短信验证码接口 |
| 20200804 |  |  | 在2.5实现以下接口  app服务初始化：查询优先展示的服务信息  app服务初始化：分页查询服务信息  根据登录令牌查询操作者信息：增加证书返回值 |
| 20200812 |  |  | 在2.6实现以下接口  8外部系统接口：8.3用户解锁  9公共接口：9.19流程配置查询  13自助服务：13.1.1自助申请获取短信验证码接口  13自助服务：13.1.2自助申请验证短信验证码接口  13自助服务：13.2查询云密钥区域信息  13自助服务：13.3查询云密钥常量类型  13自助服务：13.4查询云密钥项目信息  13自助服务：13.5查询云密钥机构信息 |
| 20200918 |  |  | 在2.7实现以下接口  8外部系统接口：8.4上传签章图片  9公共接口：9.19获取项目相关配置  14微信小程序：14.1获取用于人脸绑定的bizToken  14微信小程序：14.2获取用于人脸登录认证的bizToken  14微信小程序：14.3获取人脸绑定结果  14微信小程序：14.4获取人脸登录结果 |

# 术语说明

## 1.1. UID/云密钥用户唯一标识

用于业务上区分单一用户实体的标识，例如身份证号、工号

## 1.2. 证书用户

云密钥证书所属用户

## 1.3. (被)授权用户

并非某证书所属用户，但经证书用户授权，临时拥有证书使用权限的用户

## 1.4. 双大括号{{XXX}}

本文档中请求响应简单示例中的双大括号均表示变量，调用接口时不需要把双大括号当做参数的一部分。例如：

{

"userUid": "{{uid}}"

}

若userUid为1234，调用接口时的参数为：

{

"userUid": "1234"

}

# 证书用户模块



## 用户证书查询（uid查询）

### 2.1.1 用户证书查询

按照UID（用户唯一标识），查询某个证书用户下所拥有的证书对（Base64编码）。该接口返回UID对应的最新证书。

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/cert/search |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userUid | String | 云密钥用户唯一标识 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为键值对，key为用户uid（即云密钥账号），value为该uid所持有的证书（Base64编码）的数组。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| key键 | String | 云密钥账号/uid |
| value值 | Array | 该用户下所有的证书base64编码数组。当数组长度等于1时，则返回**签名证书**；  当数组长度等于2时，则按顺序返回**签名证书、加密证书** |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userUid": "{{uid}}"  } | userUid（云密钥用户唯一标识） |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "cxy1111": [  "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj..",  "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj.."  ]  }  } | 返回userUid下所有证书(Base64编码)。用户可能会有多张证书  一般返回两张证书时，按照以下格式返回  **[”签名证书”,”加密证书”]** |
| curl请求示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/cert/search -H 'Content-Type:application/json' -d '{"userUid": "{{uid}}"}' | |

### 2.1.2 用户证书查询（uid+项目uid查询）

按照UID（用户唯一标识）和项目UID（项目唯一标识），查询某个证书用户在某个具

体项目下所拥有的证书对（Base64编码）。该接口返回对应的最新证书。

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/cert/search/2CKV1 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userUid | String | 云密钥用户唯一标识 |
| projectUid | String | 云密钥项目唯一标识 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为键值对，key为用户uid（即云密钥账号），value为该uid所持有的证书（Base64编码）的数组。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signCert | String | 用户签名证书 |
| encCert | String | 用户加密证书 |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userUid": "{{uid}}",  "projectUid": "{{projectUid}}"  } | userUid（云密钥用户唯一标识）  projectUid（云密钥项目唯一标识） |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": [{  "signCert": "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj..",  "encCert": "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj.."  }]  } | 返回userUid下的证书(Base64编码) |
| curl请求示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/cert/search/2CKV1 -H 'Content-Type:application/json' -d '{  "userUid": "{{uid}}",  "projectUid": "{{projectUid}}",  }' | |

## 身份预验证

证书用户进行身份验证：

1. 口令PIN验证： 只有一步操作，验证成功返回令牌userToken
2. 口令PIN+短信验证码验证: 涉及两步操作  
   a.获取短信验证码（后台验证口令，根据后台配置，短信下发验证码，或者直接在接口响应值中返回，不通过短信下发验证码）  
   b.验证短信验证码，验证成功，返回令牌userToken

该userToken根据参数loginUsage变更，可能会是

1、变量值为1，临时用于绑定设备、人脸；

2、变量值为0，用于云密钥服务的调用

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/user/certcontent/dologin |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginUsage | Int | 0为正常身份认证  1为绑定用户时的身份认证 |
| userLoginType | Int | 1为pin码验证  2为短信验证码验证 |
| userPin | String | 对用户输入的pin码做Base64编码的值 |
| cert | Cert | 参考12.1.1 [Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| applicationId | String | 应用id，由云密钥分配，若userLoginType为2，则在[2.3 身份预验证之短信验证码校验](#_身份预验证之短信验证码校验)传applicationId |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 如果userLoginType验证类型为1，验证成功后台直接返回userToken |
| smsCodeGenMinCycle | Int | 如果验证类型userLoginType为2，验证成功后台则下发验证短信至用户手机,并返回下次能产生验证码的时间间隔，单位是秒 |
| phoneValidCode | String | 如果配置通过短信下发验证码，该值为空  如果配置不通过短信下发验证码，该值不为空 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | Pin码验证  {  "loginUsage":0,  "userLoginType":1,  "userPin":"{{pinBase64}}",  "cert":{  "certContent":"{{certContent}}"  },  “applicationId”:”{{applicationId}}”  }  短信验证  {  "loginUsage":0,  "userLoginType":2,  "userPin":"{{pin}}",  "cert":{  "certContent":"{{certContent}}"  }  } | **身份验证说明： loginUsage:0为正常登录，1=绑定用户**  **userPin不能为空,用户PIN的Base64编码;**  **certContent不能为空;**  **userLoginType：1为PIN登录，2为PIN+短信验证码** |
| 响应示例 | Pin码验证  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "679401E9D0553FE72A154EFE8492773775"  }  }  短信验证  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "smsCodeGenMinCycle": 60,  "phoneValidCode": "根据配置决定是否返回"  }  } | 如果**userLoginType**认证类型为1，认证成功后台直接返回userToken  如果认证类型为2，认证成功后台则下发验证短信至用户手机,并返回下次能产生验证码的时间间隔，单位是秒**smsCodeGenMinCycle**； |
| curl请求示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/user/certcontent/dologin -H 'Content-Type:application/json' -d '{  "loginUsage":0,  "userLoginType":1,  "userPin":"{{pinBase64}}",  "cert":{ "certContent":"{{certContent}}"},  "applicationId":"{{application}}",  }' | |

## 身份预验证之短信验证码校验

身份验证方式2的短信验证码校验。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/user/certcontent/login/phonevalidcode |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginUsage | int | 0为正常身份认证，  1为绑定用户时的身份认证  此处与[获取验证码](#_身份预验证(V1.5))的loginUsage相同 |
| phoneValidCode | String | 短信验证码 |
| cert | Cert | 参考12.1.1 [Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| applicationId | String | 应用id,由云密钥分配 |

### 响应结构

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 验证成功后台返回userToken |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "loginUsage": 0,  "phoneValidCode": "{{phoneValidCode}}",  "cert": {  "certContent": "{{certContent}}"  },  “applicationId”:”{{applicationId}}”  } | 短信验证码需要填写；**loginUsage要跟产生验证码时填写的一致。 loginUsage:0为正常登录，1=绑定用户;**  **certContent:Base64编码的证书内容** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "08445A0123C073095ED0FFAFE28A68A775"  }  } | 登录成功后，后台返回该次会话的令牌userToken, |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/user/certcontent/login/phonevalidcode -H 'Content-Type:application/json' -d '{  "loginUsage": 0,  "phoneValidCode": "{{phoneValidCode}}",  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 证书用户口令重试次数

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/cert/certcontent/getSurplusNum |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的值为口令剩余重试次数，若已锁死返回-1。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "cert": {  "certContent": "{{certContent}}"  }  } | **certContent:Base64编码的证书内容** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": 2  } | contents的值为口令剩余重试次数，  若已经锁死，则返回-1 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/cert/certcontent/getSurplusNum -H 'Content-Type:application/json' -d '{  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 查询授权有效性配置

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/cert/authorize/validity/config/uid |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userUid | String | 云密钥用户唯一标识 |
| projectUid | String | 云密钥项目标识 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| name | String | uid对应的证书用户名称 |
| defaultAuthorizeInterval | Int | 默认授权有效期间隔，单位为天，-1表示不限制 |
| defaultAuthorizeCount | Int | 默认授权有效期次数，-1表示不限制 |
| permitAuthorize | Boolean | 是否允许通过接口提交授权信息  有些要求严格的环境，只允许通过系统录入授权信息，不能通过接口录入  true表示允许  false表示不允许 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userUid": "{{uid}}",  "projectUid": "{{projectUid}}"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "defaultAuthorizeInterval": 90,  "defaultAuthorizeCount": 100,  "validityTimeLimited": true,  "validityCountLimited": true,  "permitAuthorize": true  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61/cloudkeyserver/cert/authorize/validity/config/uid -H 'Content-Type:application/json' -d '{  "userUid": "{{userUid}}",  "projectUid": "{{projectUid}}"  }' | |

## 解析证书属性及扩展

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/cert/certcontent/parse/1CKV13 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1.[Cert结构](#_响应结构说明) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| version | Int | 版本号，参考[证书的版本号](#_12.2.10_证书的版本号) |
| signAlgo | Int | 签名算法，参考[签名算法](#_签名算法) |
| publicAlgo | Int | 公钥算法，参考[公钥算法](#_12.2.11_公钥算法) |
| publicBits | Int | 公钥位数 |
| keyUsage | Int | 密钥用法，参考[密钥用法](#_12.2.12_密钥用法) |
| publicEccCurve | Int | 公钥的椭圆曲线号，国密算法证书才有  参考[ECC曲线的类型](#_12.2.13_ECC曲线的类型) |
| validityStart | String | 有效期开始时间 |
| validityEnd | String | 有效期结束时间 |
| sn | String | 证书序列号 |
| thumbprints | Thumbprint[] | 证书摘要，参考12.1.8[Thumbprint](#_12.1.8_Thumbprint结构) |
| issuer | String | 颁发者信息，参考12.1.12 [Issuer](#_12.1.12._Issuer结构)结构 |
| subject | String | 主体信息，参考12.1.13 [Subject](#_12.1.13._Subject结构)结构 |
| extendsions | Extension[] | 证书扩展值，参考12.1.9 [Extension](#_12.1.9._Extension结构)  目前只解析4个扩展值，有则返回，否则为空  深圳地标实体唯一id  NETCA的实体唯一id  用户证书服务号  前证书微缩图 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "cert":{  "certContent":"{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "version": 2,  "signAlgo": 25,  "publicAlgo": 4,  "publicBits": 256,  "keyUsage": 1,  "publicEccCurve": 7,  "validityStart": "2019-01-01 12:00:00",  "validityEnd": "2020-01-01 12:00:00",  "sn": "2018052900371275",  "thumbprints": [{  "algorithm": 16384,  "hash": "C372631E606EC8599C5E5E3CAE554C0E8671D7BFE1102D309B3CE0B6C6268B9E"  }],  "issuer": {  "issuer": "颁发者",  "displayName": "颁发者的显示名字",  "hexEncode": "颁发者DER编码后再Hex编码",  "c": [],  "o": [],  "ou": [],  "cn": [],  "email": [],  "st": [],  "l": [],  "dc": []  },  "subject": {  "subject": "",  "displayName": "主体的显示名字",  "hexEncode": "主体DER编码后再Hex编码",  "c": [],  "o": [],  "ou": [],  "cn": [],  "email": [],  "st": [],  "l": [],  "dc": []  },  "extensions": [{  "oid": "",  "value": ""  }]  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/cert/certcontent/parse -H 'Content-Type:application/json' -d '{  "cert":{ "certContent":"{{certContent}}"}  }' | |

# 授权管理模块



## 查询已被授权信息

根据证书信息查询出该证书下，已经被授权的用户；也就是能进行身份验证的被授权人。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/certcontent/list/authorized |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| total | Int | 返回被授权人个数 |
| rows | 数组 | row信息，参考12.1.14.1 [Row](#_12.1.14.1Row结构1)结构1 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "cert": {  "certContent": "{{certContent}}"  }  } | 授权人的Cert信息，查询出该证书下所有的已经被授权的用户  **certContent:Base64编码的证书内容** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "total": 2,  "currentRowsSize": 2,  "rows": [  {  "id": 59,  "name": "1111"  },  {  "id": 61,  "name": "cxytesttestesttes"  }  ]  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/certcontent/list/authorized -H 'Content-Type:application/json' -d '{  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 用户分页查询授权记录

根据证书信息查询出该证书下，所有的授权记录。用于用户自己的授权许可

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/authorizerequest/list |
| **请求数据格式** | application/json |
| **请求消息** |  |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |
| pageIndex | Int | 分页页码，第几页 |
| pageSize | Int | 分页显示数量 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| total | Int | 该证书下被授权人的总数量 |
| currentRowsSize | Int | 当前返回的被授权人数量 |
| rows | 数组 | row信息，参考12.1.14.2[Row结构2](#_12.1.14.2_Row结构2) |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userToken": "{{userToken}}",  "pageIndex": 1,  "pageSize": 3,  "cert": {  "certContent": "{{certContent}}"  }  } | **pageIndex 第几页 pageSize 每页显示数量** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "total": 1,  "currentRowsSize": 1,  "rows": [{  "id": 59,  "name": "1111",  "authStatus": 1,  "certId": null,  "phone": "156\*\*\*\*2293",  "allowAuthorized": false,  "allowReAuthorized": true,  "allowUnAuthorized": true,  "allowActiveAuthorized": true,  "allowRenewAuthorized": true,  "applicationTime": "2019-07-05 11:05:35",  "validStart": "2019-07-05 11:05:35",  "validEnd": "2019-07-05 11:05:35",  "allowCallCount": 10,  "memo": "备注信息"  }]  }  } | **客户端可以利用**  allowAuthorized、allowReAuthorized、allowUnAuthorized  allowActiveAuthorized  allowRenewAuthorized  **这五个字段来显示相应的操作按钮给真正的证书用户进行操作。**  validStart - validEnd  为有效授权时间，null则表示不限制  allowCallCount 为被授权用户允许使用该证书登录并签名的次数， -1表示不限制 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/authorizerequest/list -H 'Content-Type:application/json' -d '{  "userToken": "{{userToken}}",  "pageIndex": 1,  "pageSize": 3,  "cert": {  "certContent": "{{certContent}}"  }  }' | |

## 用户授权状态变更

证书用户已进行身份验证完毕的之后，可以对已录入的授权记录进行真正的授权操作（冻结、解冻、重新授权、重置授权、正常授权）。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/changestatus |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | Stirng | 用户登录令牌 |
| authorizedUserIds | Array | 授权记录的id集合,见[用户分页查询授权记录](#_用户分页查询授权记录信息) |
| expectedOperation | Int | 证书用户本人期望的授权变更操作  0：同意授权  1：冻结授权  2：解冻授权  3：授权解锁(重置授权)  4：删除授权 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "authorizedUserIds": {{authorizedUserIds}},  "userToken": "{{userToken}}",  "expectedOperation": 0  } | authorizedUserIds 是分页查询所得的授权记录的id整型值  e.g "authorizedUserIds":[1,2,6] |
| 响应示例 | 成功响应  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  }  }  失败响应{授权记录不存在，已授权}  {  "responseResult": {  "status": -1,  "msg": "查询不到authorizedIds对应的记录"  },  "contents": null  } | 非法授权操作，返回响应码status -10 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/changestatus -H 'Content-Type:application/json' -d '{  "authorizedUserIds": {{authorizedUserIds}},  "userToken": "{{userToken}}",  "cert": {  "certContent": "{{certContent}}"  }  }' | |

## 被授权用户可用证书查询

带有uid（用户唯一标识）的被授权用户，可使用该接口，查询该uid（用户唯一标识）对应可使用的授权证书。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/cert/search |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authorizedUserUid | String | 被授权用户的uid（用户唯一标识） |
| projectUid | String | 云密钥项目标识 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authorizedUserId | Int | 授权记录id |
| authorizedUserName | String | 授权用户名字 |
| certInfo | certInfo | certInfo信息，见12.1.15[CertInfo](#_12.1.15._Row结构)结构 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "authorizedUserUid":"{{authorizedUserUid}}",  "projectUid":"{{projectUid}}"  } | authorizedUserUid（授权用户唯一标识uid） |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": [{  "authorizedUserId": 1,  "authorizedUserName": "张三",  "certInfo": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ]  }]  } | 返回该uid下所有可以使用授权证书(Base64编码)。 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/cert/search -H 'Content-Type:application/json'  -d '{  "authorizedUserUid": "{{ authorizedUserUid}}",  "projectUid": "{{projectUid}}"  }' | |

## 授权申请

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/certcontent/authorized/apply/batch |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | Stirng | 用户登录令牌 |
| targetUid | String | 目标证书用户的uid  一个uid可能对应多对证书，若不传targetCert，则默认使用[用户证书查询](#_用户证书查询)的第一个证书做授权 |
| targetCert | Cert | 目标授权使用的证书，见12.1.1[Cert](#_响应结构说明)结构 |
| authorizedUsers | Array | 授权信息数组,参考12.1.11.1[authorizedUser](#_12.1.11.1_AuthorizedUser结构（Authoriz)结构1 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userToken": "{{userToken}}",  "targetUid": "{{targetUid}}",  "targetCert": {  "certContent": "{{certContent}}"  },  "authorizedUsers": [{  "uid": "{{uid}}",  "name": "{{name}}",  "phone": "{{phone}}",  "memo": "{{memo}}",  "validStart": "{{validStart}}",  "validEnd": "{{validEnd}}",  "validCount": 50  }]  } | 通过[查询授权有效性配置](#_查询授权有效性配置)获取默认的配置，提交的授权有效性信息不能超过默认配置。  validStart和validEnd的时间间隔不能大于默认时间间隔，validCount不能大于默认次数 |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61/cloudkeyserver/authorizeduser/authorized/apply/batch -H 'Content-Type:application/json' -d '{  "userToken": "{{userToken}}",  "targetUid": "{{targetUid}}",  "targetCert": "{{targetCert}}",  "authorizedUsers": {{authorizedUsers}}  }' | |

## 授权有效性修改

用于修改授权有效期和授权次数，仅已录入、已授权、已过期允许调用，已过期的授权信息修改后会变成已授权。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/authorized/validity/modify |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | Stirng | 用户登录令牌 |
| authorizedUser | AuthorizedUser | 授权用户信息，参考12.1.11.2[authorizedUser](#_12.1.11.1_AuthorizedUser结构2)结构2 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents结构为如下。参考[用户分页查询授权记录的分页响应结构](#_响应结构说明_3)

|  |  |  |
| --- | --- | --- |
| id | Int | 授权记录的id标识（全局唯一） |
| name | String | 被授权人的名字 |
| authStatus | Int | 被授权人当前的状态  0=已录入，尚未被授权 1=已被授权 2=冻结授权  3=授权过期  4=删除授权（不会返回客户端） |
| allowAuthorized | Boolean | 是否能被授权 |
| allowReAuthorized | Boolean | 是否能被重置授权（解锁） |
| allowUnAuthorized | Boolean | 是否能被冻结授权 |
| allowActiveAuthorized | Boolean | 是否能被激活授权（解冻） |
| allowRenewAuthorized | Boolean | 是否能被重新授权 |
| applicationTime | String | 管理员录入时间 |
| validStart | String | 有效授权开始时间 |
| validEnd | String | 有效授权结束时间 |
| allowCallCount | Int | 允许调用次数 |
| memo | String | 备注信息 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userToken": "{{userToken}}",  "authorizedUser": {  "id": 123,  "validStart": "{{validStart}}",  "validEnd": "{{validEnd}}",  "validCount": 50  }  } | 在有配置默认有效期长度和默认次数的情况下，只能延长有效性的配置，不能缩短，通过[查询授权有效性配置](#_查询授权有效性配置)获取默认的配置。  即validStart、validEnd和validCount只能增大，不能缩小 |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "id": 59,  "name": "1111",  "authStatus": 1,  "certId": null,  "phone": "156\*\*\*\*2293",  "allowAuthorized": false,  "allowReAuthorized": true,  "allowUnAuthorized": true,  "allowActiveAuthorized": true,  "allowRenewAuthorized": true,  "applicationTime": "2019-07-05 11:05:35",  "validStart": "2019-07-05 11:05:35",  "validEnd": "2019-07-05 11:05:35",  "allowCallCount": 10,  "memo": "备注信息"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61/cloudkeyserver/authorizeduser/authorized/validity/modify  -H 'Content-Type:application/json' -d '{  "userToken": "{{userToken}}",  "authorizedUser": {{authorizedUser}}  }' | |

# 被授权用户身份验证模块



## 被授权用户获取口令PIN重试次数

当进行身份验证，口令PIN错误（后台返回错误码-3），可以调用此接口返回该用户的剩余尝试口令PIN次数。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST form |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/getSurplusNum |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authorizedUserId | INT | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为口令剩余重试次数，若已锁死返回-1。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "authorizedUserId":"{{authorizedUserId}}"  } | authorizedUserId 则为客户端查询后，用户选择认证的授权记录id |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": 5  } | contents的值为口令剩余重试次数，  若已经锁死，则返回-1 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/getSurplusNum -H 'Content-Type:application/json' -d '{  "authorizedUserId": "{{authorizedUserId}}"  }' | |

## 被授权用户口令PIN认证

只有一步操作，验证成功返回令牌userToken

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/certcontent/pin/login |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginUsage | Int | 0为正常身份认证，1为绑定用户时的身份认证 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| userPin | String | 对用户输入的pin码做Base64编码的值 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| applicationId | String | 应用id |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 验证成功后台直接返回userToken |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "loginUsage":0,  "authorizedUserId":"{{authorizedUserId}}",  "userPin":"{{pinBase64}}",  "cert":{  "certContent":"{{certContent}}"  },  "applicationId":"{{applicationId}}"  } | authorizedUserId 则为客户端查询后，用户选择认证的授权记录id |
| 响应示例 | 成功响应  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "24B9DE6AED996752843C0C514D4D791075#59"  }  }  失败响应  {  "responseResult": {  "status": -1,  "msg": "该被授权用户不存在"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/certcontent/pin/login -H 'Content-Type:application/json' -d '{  "loginUsage":0,  "authorizedUserId":{{authorizedUserId}},  "userPin":"{{pinBase64}}",  "cert":{ "certContent":"{{certContent}}"},  "applicationId":"{{applicationId}}"  }' | |

## 口令PIN+短信验证码验证

涉及两步接口调用操作：  
a.获取短信验证码（后台验证口令，根据后台配置，短信下发验证码，或者直接在接口响应值中返回，不通过短信下发验证码）

b.验证短信验证码，验证成功，返回令牌userToken

### 获取短信验证码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/certcontent/smsvalidcode/login |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginUsage | Int | 0为正常身份认证，1为绑定用户时的身份认证 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| userPin | String | 对用户输入的pin码做Base64编码的值 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| smsCodeGenMinCycle | Int | 下次能产生验证码的时间间隔（单位秒） |
| phoneValidCode | String | 如果配置通过短信下发验证码，该值为空  如果配置不通过短信下发验证码，该值不为空 |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "loginUsage":0,  "authorizedUserId":"{{authorizedUserId}}",  "userPin":"{{pinBase64}}",  "cert":{  "certContent":"{{certContent}}"  }  } | authorizedUserId 则为客户端查询后，用户选择认证的授权记录id |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "smsCodeGenMinCycle": 60,  "phoneValidCode": "根据配置决定是否返回"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/certcontent/pin/login -H 'Content-Type:application/json' -d '{  "loginUsage":0,  "authorizedUserId":{{authorizedUserId}},  "userPin":"{{pinBase64}}",  "cert":{ "certContent":"{{certContent}}"}  }' | |

### 验证短信验证码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/authorizeduser/certcontent/smsvalidcode/login/verify |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| validCode | String | 短信验证码 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| applicationId | String | 应用id |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户身份验证成功后的登录令牌 |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "authorizedUserId":{{authorizedUserId}},  "validCode": "{{validCode}}",  "cert": {  "certContent": "{{certContent}}"  },  "applicationId":"{{applicationId}}"  } | authorizedUserId 则为客户端查询后，用户选择认证的授权记录id  validCode 则是表示手机验证码的变量 |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "08445A0123C073095ED0FFAFE28A68A775"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/authorizeduser/certcontent/smsvalidcode/login/verify -H 'Content-Type:application/json' -d '{  "authorizedUserId":{{authorizedUserId}},  "validCode":"{{validCode}}",  "cert":{ "certContent":"{{certContent}}"},  "applicationId":"{{applicationId}}"  }' | |

# 身份验证流程

下面的接口按顺序调用，或者参考流程图



## 身份验证类型常量(loginTypeBitValue)说明

使用位掩码，用一个二进制数来表示身份验证类型；

二进制的每一位来表示一种身份验证类型，该位上的值：0表示不支持该验证方式，1表示支持该类型。

目前有以下几种验证类型：

用二进制第一位值为1表示支持口令验证，即1(1)

用二进制第二位值为1表示支持口令+短信验证，即10(2)  
用二进制第三位值为1表示支持口令+指纹验证，即100(4)

用二进制第四位值为1表示支持口令+刷脸验证，即1000(8)  
用二进制第五位值为1表示支持口令+手机设备信息验证，即1 0000(16)

用二进制第六位值为1表示支持口令+指纹+手机设备信息验证，即10 0000(32)

用二进制第七位值为1表示支持口令+刷脸+手机设备信息验证，即100 0000(64)

|  |  |  |
| --- | --- | --- |
|  | 二进制值 | 十进制值 |
| 口令 | 0001 | 1 |
| 口令+短信验证码 | 0010 | 2 |
| ~~口令+指纹~~ | ~~0100~~ | 4 |
| 口令+刷脸 | 1000 | 8 |
| 口令+手机设备信息 | 0001 0000 | 16 |
| 口令+指纹+手机设备信息 | 0010 0000 | 32 |
| ~~口令+刷脸+手机设备信息~~ | ~~0100 0000~~ | ~~64~~ |

说明：要支持多种验证类型，请使用‘或’运算。

例如：支持口令 和 （口令+指纹）

获取loginTypeBitValue值的伪代码：int loginTypeBitValue = PIN | PIN\_FINGER // 0001 | 0100  
  
验证一个loginTypeBitValue值是否支持某种验证方式，使用‘与’运算

例如：验证loginTypeBitValue 值是否支持口令+刷脸

伪代码：boolean flag = (PIN\_FACE & loginTypeBitValue) != 0 // 为true说明支持口令+刷脸

## 二维码内容

证书用户二维码内容简单说明：

|  |  |  |
| --- | --- | --- |
| **参数** | **类型** | **说明** |
| q(qrCodeIdentity) | String | 签名随机数（也是二维码的唯一标识） |
| l(loginTypeBitValue) | Int | 参考 [身份验证类型常量说明](#_身份验证类型常量(loginTypeBitValue)说明) |
| u(commitUrl) | String | 签名结果提交的域名 例如：http://ip:port/ |

q:二维码唯一标识qrCodeIdentity；手机需要提交该参数到后台

l:该二维码用户支持的身份验证类型，参考 [身份验证类型常量说明](#_身份验证类型常量(loginTypeBitValue)说明) ,目前该值不会有多种操作同时存在，也就是说，该值二进制位数上为1的只有一个（比如00010000 或00100000）

u:二维码提交的域名地址（如https://192.168.111.124:8080/）后续路径需要自己根据l进行判断拼接上去。（服务器为什么不直接拼接返回？二维码内容过大会导致有些手机识别不出）

身份验证的二维码（netcack1）：  
为了区分二维码类型，在json前面加个前缀netcack1和netcack3，最终的格式如下：

netcack1和netcack3的区别：

netcack3需要调用[5.4 绑定认证二维码状态](#_绑定认证二维码的状态)的接口

netcack1不需要

netcack1://{"l":16,"q":"42084A750AA3955782","u":"http://192.168.200.208:7092/"}

netcack3://{"l":16,"q":"42084A750AA3955782","u":"http://192.168.200.208:7092/"}

绑定的二维码（netcack2）：

netcack2://{"l":32,"q":"42084A750AA3955782#3","u":"http://192.168.200.208:7092/"}

## 客户端获取认证二维码

客户端在选择证书用户之后，获取二维码以供扫码。结合移动客户端，提供更为安全的认证方式

### 5.3.1 客户端获取认证二维码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/certcontent/login/qrcode |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginTypeBitValue | Int | 参考[身份验证类型常量](#_身份验证类型常量(loginTypeBitValue)说明)，十进制整形数字，通常是高位认证的或运算（8+16+32），可从[获取证书所支持的身份验证类型](#_获取证书所支持的身份验证类型)响应获得 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| projectUid | String | 项目唯一标识，由云密钥分配 |
| applicationId | String | 应用id，由云密钥分配 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeExpireDate | Int | 二维码过期时间（单位秒） |
| qrCodeBase64 | String | 二维码图片的Base64编码 |
| qrCodeIdentity | String | 二维码唯一标识 |
| qrCodeContent | String | 二维码表示的所有类JSON文本信息（去掉协议前缀，剩下未JSON的OBJECT），见[二维码内容](#_二维码内容) |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "loginTypeBitValue": {{loginTypeBitValue}},  "cert": {  "certContent": "{{certContent}}"  },  "projectUid": "{{projectUid}}",  "applicationId": "{{applicationId}}"  } | loginTypeBitValue |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "qrCodeBase64": "iVBORw0KGgoAAAANSUhE=..",  "qrCodeIdentity": "1AE7EDC65F90800E",  "qrCodeExpireDate": 120,  "qrCodeContent":"netcack1://{\"l\":16,\"q\":\"42084A750AA3955782\",\"u\":\"http://192.168.200.208:7092/\"}"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/certcontent/login/qrcode -H 'Content-Type:application/json' -d '{  "loginTypeBitValue":{{loginTypeBitValue}},  "cert":{ "certContent":"{{certContent}}"}  }' | |

### 5.3.2 客户端获取认证二维码（无状态）

若是调用完该接口产生的二维码，扫码方在扫码后需立即调用[5.3.3 关联无状态二维码。](#_5.3.3_关联无状态二维码)

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/login/qrcode/nostatus/2CKV1 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginTypeBitValue | Int | 期望的使用的认证类型  参考[身份验证类型常量](#_身份验证类型常量(loginTypeBitValue)说明)，十进制整形数字，通常是高位认证的或运算（8+16+32），可从[获取证书所支持的身份验证类型](#_获取证书所支持的身份验证类型)响应获得，可选，不传时默认为服务端支持的全部类型 |
| type | Int | 认证二维码的类型  0表示产生以下全部类型的二维码  3表示产生app无状态的二维码  4表示产生微信小程序无状态的二维码 |
| projectUid | String | 项目唯一标识，由云密钥分配 |
| applicationId | String | 应用id，由云密钥分配 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeExpireDate | Int | 二维码过期时间（单位秒） |
| qrCodeIdentity | String | 二维码唯一标识 |
| qrCodes | QRCode结构的数组 | 参考[12.2.10. QRCode结构](#_12.2.10._QRCode结构) |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "loginTypeBitValue": "{{loginTypeBitValue}}",  "type": 3,  "projectUid": "{{projectUid}}",  "applicationId": "{{applicationId}}"  } | loginTypeBitValue |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "qrCodes": [{  "type": 3,  "qrCodeBase64": "iVBORw0KGgoAAAANSUhE=..",  "qrCodeContent": "netcack3://{\"l\":16,\"q\":\"42084A750AA3955782\",\"u\":\"http://192.168.200.208:7092/\"}"  }],  "qrCodeIdentity": "1AE7EDC65F90800E",  "qrCodeExpireDate": 120  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/login/qrcode/nostatus/2CKV1 -H 'Content-Type:application/json' -d '{  "loginTypeBitValue":{{loginTypeBitValue}},  "type":"{{type}}",  "projectUid":"{{projectUid}}",  "applicationId":"{{applicationId}}"  }' | |

### 5.3.3 关联无状态二维码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/certcontent/login/qrcode/bind/status/2CKV1 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| qrCodeIdentity | String | 二维码唯一标识 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为可用的认证类型。

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "qrCodeIdentity": "{{qrCodeIdentity}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": 48  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/certcontent/login/qrcode/bind/status -H 'Content-Type:application/json' -d '{  "qrCodeIdentity":"{{qrCodeIdentity}}",  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 手机扫描客户端二维码

PDA产生二维码，移动客户端扫码；或者移动客户端获取二维码后，自行解析

手机扫描 [二维码](#_身份验证类型常量(loginTypeBitValue)说明)，判断二维码是否是netcack1开头，获取到qrCodeIdentity和loginTypeBitValue和commitUrl

## 用户人脸认证

当二维码中的(**loginTypeBitValue & 8)!=0时，可以选择调用此接口。**

**注意：刷脸+pin 方式 不绑定设备，不需要检查 设备是否能进行该类型的身份验证。**

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/certcontent/identity/facedata/login |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| faceType | Int | 人脸类型：0=人脸识别（目前只有0） |
| faceData | String | 人脸图片数据的Base64字符串 |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空。 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| userPin | String | 输入的pin，做Base64编码后的值。 |
| qrCodeIdentify | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 验证通过后返回的用户登录令牌 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "faceType": 0,  "faceData": "{{faceData}}",  "userType": 1,  "authorizedUserId":{{authorizedUserId}},  "userPin": "{{pinBase64}}",  "qrCodeIdentity": "{{qrCodeIdentity}}",  "cert": {  "certContent": "{{certContent}}"  }  } | faceData  是客户端拍照获取的用户人脸照片数据，base64编码 |
| 响应示例 | 成功响应  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "24B9DE6AED996752843C0C514D4D791075#59"  }  }  失败响应{人脸认证失败、pin码错误}  {  "responseResult": {  "status": -1,  "msg": "人脸验证失败"  },  "contents": null  }  {  "responseResult": {  "status": -3,  "msg": "用户口令错误"  },  "contents": null  } | status返回 -3，则说明pin错误；  userToken为验证成功后返回的令牌。 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/certcontent/identity/facedata/login -H 'Content-Type:application/json' -d '{  "faceType": 0,  "faceData": "{{faceData}}",  "userType": 1,  "authorizedUserId":{{authorizedUserId}},  "userPin": "{{pinBase64}}",  "qrCodeIdentity": "{{qrCodeIdentity}}",  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 检查手机设备是否能进行身份验证

当二维码中的 **(loginTypeBitValue & 8)==0**，则不支持pin+人脸的认证方式，调用该接口检查设备能否进行认证

若是可以认证，客户端则根据**loginTypeBitValue** 与[身份验证类型常量](#_身份验证类型常量(loginTypeBitValue)说明)的位与的结果，分别进入不同认证方式的处理流程，并调用以下相应接口 [手机提交口令身份验证接口](#_手机提交口令身份验证接口)、[手机提交口令指纹身份验证接口](#_手机提交口令指纹身份验证接口(V1.5))、[手机提交口令人脸身份验证接口](#_手机提交口令人脸身份验证接口（V1.3）（V1.5）) 进行认证

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/device/login/check |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginTypeBitValue | Int | 参考[身份验证类型常量](#_身份验证类型常量(loginTypeBitValue)说明)，十进制整形数字，此处是使用二维码返回的 loginTypeBitValue |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |

### 请求结构说明

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| checkedStatus | Int | 0 = 可以进行身份验证，**rejectMsg=null**  1 = 不可以进行身份验证,提示信息在**rejectMsg** |
| rejectMsg | String | 根据 **checkedStatus响应信息** |

[参考响应结构](#_响应结构)，contents中的内容见下表。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "qrCodeIdentity": "{{qrCodeIdentity}}",  "loginTypeBitValue": {{loginTypeBitValue}},  "deviceId": "ASDFGHJ设备唯一标志"  } |  |
| 响应示例 | 成功响应  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "checkedStatus": 0,  "rejectMsg": ""  }  }  失败响应{二维码超时}  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "checkedStatus": 1,  "rejectMsg": "验证信息未生成或已失效"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/device/login/check -H 'Content-Type:application/json' -d '{  "qrCodeIdentity": "{{qrCodeIdentity}}",  "loginTypeBitValue":{{loginTypeBitValue}},  "deviceId": "fdshu9fs-fds-fds-fs"  }' | |

## 手机提交口令身份验证接口

当二维码中的(**loginTypeBitValue & 16)!=0时，可以选择调用此接口。**

但应该先调用[检查手机设备是否能进行身份验证](#_检查手机设备是否能进行身份验证)，检查验证可行性

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/base64/identity/pin/login |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userPin | String | **输入的pin，做Base64编码后的值** |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 身份验证通过后返回的用户登录令牌 |

[参考响应结构](#_响应结构)，contents中的内容见下表。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userPin": "{{pinBase64}}",  "qrCodeIdentity": "{{qrCodeIdentity}}",  "deviceId": "{{deviceId}}"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "24B9DE6AED996752843C0C514D4D791075#59"  }  } | status返回-3则说明口令错误； |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/base64/identity/pin/login -H 'Content-Type:application/json' -d '{  "qrCodeIdentity": "{{qrCodeIdentity}}",  "deviceId": {{deviceId}},  "userPin":" {{pinBase64}}"  }' | |

## 手机提交口令指纹身份验证接口

当二维码中的(**loginTypeBitValue & 32)!=0时，可以选择调用此接口。**

但应该先调用[检查手机设备是否能进行身份验证](#_检查手机设备是否能进行身份验证)，检查验证可行性

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/base64/identity/finger/login |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userPin | String | **输入的pin，做Base64编码后的值。** |
| qrCodeIdentify | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |
| type | Int | 指纹方案常量，只能是1 |
| tbs | String | 签名原文字符串（该原文包含了 qrCodeIdentify） |
| tbs\_signature\_algo\_oid | String | 签名算法oid值 |
| tbs\_signature | String | 签名值（Base64编码） |

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 身份验证通过后返回的用户登录令牌 |

[参考响应结构](#_响应结构)，contents中的内容见下表。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 入参 | {     **"userPin"**:**"MTIzNDU2Nzg="**,     **"deviceId"**:**"AAAAAAA"**,     **"type"**:**0**,     **"data"**:{         **"tbs"**:**"{ "raw":"qrCodeIdentity" , "fid":"3" , "counter":246 , "tee\_n":"QSEE" , "tee\_v":"4.0" , "fp\_n":"ET" , "fp\_v":"320" , "cpu\_id":"0f000000262300000b5f094d1cf4c411" , "uid":"10206" }"**,         **"tbs\_signature\_algo\_oid"**:**"1.2.840.113549.1.1.11"**,         **"tbs\_signature"**:**"oosngnoso3jvhii..."**     } } | **userPin：用户输入的pin字符串，做Base64编码后的值。**  **deviceId:手机设备唯一标识**  **tbs:签名原文字符串**  **tbs\_signature\_algo\_oid:签名算法oid**  **tbs\_signature：签名值（Base64编码）**  注意：row中的值为二维码的**qrCodeIdentity.** |
| 返回 | {     **"responseResult"**:{         **"status"**:**0**,         **"msg"**:**"成功"**     },     **"contents"**:{         **"userToken"**:**"B0BADD18550A8291430"**      } } | status返回-3则说明口令错误；  userToken为验证成功后返回的令牌。 |

## 手机提交口令设备人脸身份验证接口

当二维码中的(**loginTypeBitValue & 64)!=0时，可以选择调用此接口。**

但应该先调用[检查手机设备是否能进行身份验证](#_检查手机设备是否能进行身份验证)，检查验证可行性

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/base64/identity/face/login |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| faceType | Int | 人脸类型：0=人脸识别（目前只有0） |
| userPin | String | **输入的pin，做Base64编码后的值。** |
| qrCodeIdentify | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |
| faceData | String | 人脸图片数据Base64字符串 |

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 身份验证通过后返回的用户登录令牌 |

[参考响应结构](#_响应结构)，contents中的内容见下表。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "faceType": 0,  "qrCodeIdentity": "{{qrCodeIdentity}}",  "userPin": "{{pinBase64}}",  "deviceId": "AAAAAAA",  "faceData": "{{faceData}}"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": "24B9DE6AED996752843C0C514D4D791075#59"  }  } | status返回-3则说明口令错误；。 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/base64/identity/face/login -H 'Content-Type:application/json' -d '{  "faceType": 0,  "qrCodeIdentity": "{{qrCodeIdentity}}",  "userPin":" {{pinBase64}}",  "deviceId": {{deviceId}},  "faceData": {{faceData}}  }' | |

## 客户端轮询二维码验证结果接口

调用 [‘轮询二维码验证结果’](#_轮询二维码接口_1)接口，即可

# 绑定流程



## 获取支持绑定的身份验证类型

说明：必须使用该接口支持的方式进行身份认证拿到的userToken才能进行绑定设备。

绑定设备要求获取userToken时, loginUsage 为 1 ，参考 [身份预验证](#_身份预验证) 的 loginUsage 用法

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/certcontent/user/device/registtypebitvalue |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| registTypeBitValue | Int | 支持绑定的[身份验证类型](#_身份验证类型常量(loginTypeBitValue)说明)  该身份验证类型，用于做绑定的前置验证，即获取用于绑定用途的userToken |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "registTypeBitValue": 1  }  } | registTypeBitValue  为 1 ，则说明了该用户要绑定设备或者人脸，应该先通过pin码验证 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/certcontent/user/device/registtypebitvalue -H 'Content-Type:application/json' -d '{  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 客户端获取绑定二维码

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/binddevice/qrcode |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户身份验证成功后的绑定用途令牌  获取userToken时, loginUsage 为 1 ，参考 [身份预验证](#_获取支持的验证方式) 的 loginUsage 用法 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeExpireDate | Int | 二维码过期时间（单位秒） |
| qrCodeBase64 | String | 二维码图片的base64编码 |
| qrCodeIdentity | String | 二维码唯一标识 |
| qrCodeContent | String | 二维码表示的所有类JSON文本信息（去掉协议前缀，剩下未JSON的OBJECT），见[二维码内容](#_二维码内容) |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userToken":"00FD5CA3891B59B12BE0B2D48928415475"  } | 用户身份验证成功后的绑定用途令牌。  获取userToken时, loginUsage 为 1 ，参考 [身份预验证](#_获取支持的验证方式) 的 loginUsage 用法 |
| 响应示例 | 成功  {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "qrCodeBase64": "iVBORw0KGgoAAAANSg==",  "qrCodeIdentity": "CB3F74F3A741EC5575",  "qrCodeExpireDate": 120,  "qrCodeContent": "netcack2://{\"l\":16,\"q\":\"42084A750AA3955782\",\"u\":\"http://192.168.200.208:7092/\"}"  }  }  失败{使用错误用途的userToken}  {  "responseResult": {  "status": -1,  "msg": "产生二维码失败：该令牌UserToken不属于绑定用途"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/binddevice/qrcode -H 'Content-Type:application/json' -d '{  "userToken": "{{userToken}}"  }' | |

## 绑定人脸数据

根据解析出来的二维码内容的(loginTypeBitValue & 8) != 0才可以选择调用该接口。

注意事项：该接口直接扫描二维码后即可调用，不需要进行设备检查。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/certcontent/identity/facedata/bind |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| faceType | Int | 人脸类型：0=人脸识别（目前只有0） |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空。 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取绑定二维码](#_客户端获取绑定二维码) |
| faceDataArray | Array | 人脸图片数据base64编码的文本数组，至少要有一个元素 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构

[参考响应结构](#_响应结构)，contents中的内容为绑定结果提示信息。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "faceType": 0,  "userType": 1,  "authorizedUserId":{{authorizedUserId}},  "qrCodeIdentity": "{{qrCodeIdentity}}",  "faceDataArray": [  "{{faceData1}}",  "{{faceData2}}"  ],  "cert": {  "certContent": "{{certContent}}"  }  } | faceDataArray  是一个文本数组，元素是人脸图片的base64编码文本，数组长度至少为1 |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": "绑定成功"  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/certcontent/identity/facedata/bind -H 'Content-Type:application/json' -d '{  "faceType": 0,  "userType": 1,  "authorizedUserId":{{authorizedUserId}},  "qrCodeIdentity": "{{qrCodeIdentity}}",  "faceDataArray": ["{{faceData}}"],  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 检查用户设备是否已经绑定

当二维码中的 **(loginTypeBitValue & 8)==0**，则不支持pin+人脸的绑定方式，调用该接口检查设备能否进行绑定

若是可以绑定，客户端则根据**loginTypeBitValue** 与[身份验证类型常量](#_身份验证类型常量(loginTypeBitValue)说明)的位与的结果，分别进入不同认证方式的处理流程，并调用以下相应接口 [绑定手机设备信息](#_绑定手机设备信息)、[绑定手机设备指纹信息](#_绑定手机设备指纹信息)、[绑定手机设备人脸信息](#_绑定手机设备人脸信息) 进行认证

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/device/register/check |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginTypeBitValue | Int | 参考[身份验证类型常量](#_身份验证类型常量(loginTypeBitValue)说明)，十进制整形数字，此处是使用二维码返回的 loginTypeBitValue |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取绑定二维码](#_客户端获取绑定二维码) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |

### 请求结构说明

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为绑定结果提示信息。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| registStatus | Int | 0 = 此设备尚未开通该类型的验证方式  1 = 此设备已经开通了该类型的验证方式（继续开通会覆盖该类型的身份验证，比如指纹替换）  2 = 该用户已经绑定了其他设备（继续开通会让之前设备开通的所有身份验证方式失效 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "qrCodeIdentity": "{{qrCodeIdentity}}",  "loginTypeBitValue": {{loginTypeBitValue}},  "deviceId": "ASDFGHJ设备唯一标志"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "registStatus": 0  }  } | registStatus的值：  0 = 设备还没有被绑定过  1 = 此设备绑定了当前用户（继续绑定会覆盖设备相关的身份验证，比如指纹替换）  2 =该设备已经绑定了其他用户（继续绑定会让之前绑定的所有身份验证方式失效） |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/device/register/check -H 'Content-Type:application/json' -d '{  "qrCodeIdentity": "{{qrCodeIdentity}}",  "loginTypeBitValue":{{loginTypeBitValue}},  "deviceId": "fdshu9fs-fds-fds-fs"  }' | |

## 绑定手机设备信息

根据解析出来的二维码内容的(loginTypeBitValue & 16) != 0才可以选择调用该接口。

但应该先调用[检查用户设备是否已经绑定](#_检查用户设备是否已经绑定)，检查绑定可行性

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/identity/pin/bind |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |
| deviceInfo | DeviceInfo | 参考12.1.6[DeviceInfo结构](#_DeviceInfo结构) |

### 请求结构说明

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为绑定结果提示信息。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "qrCodeIdentity": "{{qrCodeIdentity}}",  "deviceId": "ASDFGHJ设备唯一标志",  "deviceInfo": {  "deviceModel": "HUAWEI NXT-AL10",  "sdkVersion": "1.4.0",  "osVersion": "8.0.0",  "osSDKVersion": "26",  "platformType": "Android",  "imei0": "IMEI码",  "imei1": "IMEI码",  "packageID": "安装软件时产生的唯一标识",  "firstInstallTime": "2019-08-14 08:49:10",  "lastUpdateTime": "2019-08-15 13:42:13"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": "绑定成功"  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/identity/pin/bind -H 'Content-Type: application/json' -d '{  "qrCodeIdentity": "{{qrCodeIdentity}}",  "deviceId": "ASDFGHJ设备唯一标志",  "deviceInfo": {  "deviceModel": "HUAWEI NXT-AL10",  "sdkVersion": "1.4.0",  "osVersion": "8.0.0",  "osSDKVersion": "26",  "platformType": "Android",  "imei0": "IMEI码",  "imei1": "IMEI码",  "packageID": "安装软件时产生的唯一标识",  "firstInstallTime": "2019-08-14 08:49:10",  "lastUpdateTime": "2019-08-15 13:42:13"  }  }' | |

## 绑定手机设备指纹信息

根据解析出来的二维码内容的(loginTypeBitValue & 32) != 0 才可以选择调用该接口。

但应该先调用[检查用户设备是否已经绑定](#_检查用户设备是否已经绑定)，检查绑定可行性

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/identity/finger/bind |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| type | Int | 公钥类型（type=0为腾讯soter方式，type=1为google指纹方案），目前固定传1 |
| deviceId | String | 手机设备唯一标识 |
| tbs | String | 签名原文字符串 |
| tbs\_signature\_algo\_oid | String | 签名算法oid值 |
| tbs\_signature | String | 签名值（Base64编码） |
| deviceInfo |  | 参考12.1.6[DeviceInfo结构](#_DeviceInfo结构) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为绑定结果提示信息。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {     **"deviceId"**:**"AAAAAAA"**,     **"type"**:**1**,  **"authkey"**:{         **"keyjson"**:**"{ "pub\_key":"-----BEGIN PUBLIC " , "cpu\_id":"10" , "counter":123 ,"uid":"21" }"**,         **"signature"**:**"对keyjson签名值Base64编码"**     },    **"data"**:{         **"tbs"**:**"{ "raw":"qrCodeIdentity" , "fid":"3" , "counter":246 , "tee\_n":"QSEE" , "tee\_v":"4.0" , "fp\_n":"ET" , "fp\_v":"320" , "cpu\_id":"0f000000262300000b5f094d1cf4c411" , "uid":"10206" }"**,         **"tbs\_signature\_algo\_oid"**:**"1.2.840.113549.1.1.11"**,         **"tbs\_signature"**:**"oosngnoso3jvhii..."**     },  **"deviceInfo":{**  **"deviceModel": "HUAWEI NXT-AL10",**  **"sdkVersion": "1.4.0",**  **"osVersion": "8.0.0",**  **"osSDKVersion": "26",**  **"platformType": "Android",**  **"imei0": "IMEI码",**  **"imei1": "IMEI码",**  **"packageID": "安装软件时产生的唯一标识",**  **"firstInstallTime": "2019-08-14 08:49:10",**  **"lastUpdateTime": "2019-08-15 13:42:13"**  **}** } | **deviceId:手机设备唯一标识**  **tbs:签名原文字符串**  **tbs\_signature\_algo\_oid:签名算法oid**  **tbs\_signature：签名值（Base64编码）** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": "绑定成功"  } |  |

## 绑定手机设备人脸信息

根据解析出来的二维码内容的(loginTypeBitValue & 64) != 0 才可以选择调用该接口。

但应该先调用[检查用户设备是否已经绑定](#_检查用户设备是否已经绑定)，检查绑定可行性

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/identity/face/bind |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| faceType | Int | 人脸类型：0=人脸识别（目前只有0） |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取绑定二维码](#_客户端获取绑定二维码) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串 |
| faceDataArray | 数组 | 人脸图片数据base64编码的文本数组，至少要有一个元素 |
| deviceInfo | DeviceInfo | 参考12.1.6[DeviceInfo结构](#_DeviceInfo结构) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为绑定结果提示信息。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {    **"faceType":** 0**，**  **"qrCodeIdentity": "AFFFFFF"，**   **"deviceId"**:**"ABCDEF"**,     **"faceDataArray"**:[         **"脸部图片1数据的Base64编码"**,         **"脸部图片2数据的Base64编码"**,         **"脸部图片3数据的Base64编码"**     ],  **"deviceInfo":{**  **"deviceModel": "HUAWEI NXT-AL10",**  **"sdkVersion": "1.4.0",**  **"osVersion": "8.0.0",**  **"osSDKVersion": "26",**  **"platformType": "Android",**  **"imei0": "IMEI码",**  **"imei1": "IMEI码",**  **"packageID": "安装软件时产生的唯一标识",**  **"firstInstallTime": "2019-08-14 08:49:10",**  **"lastUpdateTime": "2019-08-15 13:42:13"**  **}** } | faceDataArray  是一个文本数组，元素是人脸图片的base64编码文本，数组长度至少为1 |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": "绑定成功"  } |  |

## 客户端轮询二维码验证结果接口

调用 [‘轮询二维码验证结果’](#_轮询二维码接口_1)接口，即可

# 登录消息推送

登录消息推送，主要用于两大平台的推送：安卓（Android）和苹果（IOS）

其中苹果采用苹果推送服务，定向对ios客户端推送消息。

安卓则采用MQTT协议，仅在客户端运行时，订阅推送通知。



## MQTT消息体定义

采用UTF-8编码的JSON格式的字节数组

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| msgSignature | String | 消息署名，例如“某某医院云密钥” |
| msg | JSON | 消息拓展内容,json字符串，用于兼容旧客户端对新消息类型的处理 |
| msgType | Int | 消息类型  0：登录请求通知 |
| businessTime | String | 业务时间，参考[时间格式](#_时间格式说明) |

msg结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| application | String | 应用名称 |
| applicationId | String | 应用id |
| pushApplyId | String | 申请id |
| certContent | String | 签名证书Base64编码 |
| userType | int | 用户类型 |
| authorizedUserId | int | 被授权人id |

## 安卓接入

安卓需要自己集成MQTT客户端，使用[注册推送目标app](#_响应结构说明_4)响应的MQTT服务器相关信息，自行订阅消息

## IOS接入

IOS自行接入APNs，并保证deviceToken的正确性，推送结构如下：

{

"aps": {},

"push": {{MQTT响应体}}

}

## 申请登录

调用该接口，向有权限做身份认证登录的手机设备推送登录申请的消息。

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/certcontent/login/apply/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| applicationId | String | 申请登录的应用id |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| pushApplyId | String | 登录申请id，用于后续调用查询登录结果接口 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userType": 1,  "authorizedUserId": "{{authorizedUserId}}",  "cert": {  "certContent": "{{certContent}}"  },  "applicationId": "{{applicationId}}"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": "{{pushApplyId}}"  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/certcontent/login/apply/1CKV12 -H 'Content-Type:application/json' -d '{  "userType":"{{userType}}",  "authorizedUserId":"{{authorizedUserId}}",  "cert":"{{cert}}",  "applicationId":"{{applicationId}}"  }' | |

## 查询手机登录结果

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/login/apply/status/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| pushApplyId | String | 登录申请id，登录申请接口的响应值 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| verifyStatus | Int | 0 = 验证成功。  1 = 用户未开始验证，需要继续轮询  -1 = 验证失败，有错误信息在errorMsg |
| userToken | String | verifyStatus返回0时验证成功，userToken有值。 |
| errorMsg | String | 验证失败时的错误提示信息 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "pushApplyId": "{{pushApplyId}}"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "verifyStatus": 1,  "userToken": null,  "errorMsg": null  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/login/apply/status/1CKV12 -H 'Content-Type:application/json' -d '{  "pushApplyId":"{{pushApplyId}}"  }' | |

## 注册推送目标app

注册推送

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/certcontent/app/subscribe/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| platform | Int | 平台类型  0表示iphone  1表示android |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空 |
| deviceToken | String | ios设备上app对应的固定唯一标识  安卓设备上app产生的随机唯一标识，应用数据清除将重新生成 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| notificationSubscribeUrl | String | 用于接收消息推送的服务地址  移动客户端通过url，订阅消息通知，默认采用mqtt协议，参考[MQTT消息体定义](#_MQTT消息体定义) |
| expectedTopic | String | 需要关心的topic |
| expectedWildcardTopic | String | 包含通配符的topic |
| userName | String | mqtt只读用户名 |
| key | String | mqtt只读用户名对应的密码，已加密 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "platform": 0,  "userType": 1,  "deviceToken": "{{deviceToken}}",  "authorizedUserId": "{{authorizedUserId}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "notificationSubscribeUrl": "tcp://192.168.200.80:1888",  "expectedTopic": "cloudkey/1/1/1",  "userName": "cloudkey",  "key": "AD678"  }  }  失败{不支持消息推送}  {  "responseResult": {  "status": -30,  "msg": "不支持消息通知"  },  "contents": null  } | 如果不支持MQTT，则返回错误码 -30 |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/certcontent/app/subscribe/1CKV12 -H 'Content-Type:application/json' -d '{  "userType":"{{userType}}",  "authorizedUserId":"{{authorizedUserId}}",  "cert":"{{cert}}"  }' | |

## 注销推送目标app（注销证书下的全部用户）

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/certcontent/app/unsubscribe/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| platform | Int | 平台类型  0表示iphone  1表示android |
| deviceToken | String | 设备上app对应的唯一标识  只供ios设备使用 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，这里是空，null。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "platform": 0,  "deviceToken": "{{deviceToken}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/certcontent/app/unsubscribe/1CKV12 -H 'Content-Type:application/json' -d '{  "platform":"{{platform}}",  "deviceToken":"{{deviceToken}}",  "cert":"{{cert}}"  }' | |

## 注销推送目标app（注销当前用户）

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/certcontent/app/unsubscribe/current/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| platform | Int | 平台类型  0表示iphone  1表示android |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空 |
| deviceToken | String | 设备上app对应的唯一标识  只供ios设备使用 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |

### 响应结构说明

[参考响应结构](#_响应结构)，这里是空，null。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "platform": 0,  "userType": 1,  "deviceToken": "{{deviceToken}}",  "authorizedUserId": "{{authorizedUserId}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/certcontent/app/unsubscribe/current/1CKV12 -H 'Content-Type:application/json' -d '{  "userType":"{{userType}}",  "authorizedUserId":"{{authorizedUserId}}",  "cert":"{{cert}}"  }' | |

## 确认登录

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/app/login/confirm/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| pushApplyId | String | 登录申请id，登录申请接口的响应值 |
| userToken | String | 用户登录的userToken，非空 |
| deviceToken | String | ios设备上app对应的固定唯一标识  安卓设备上app产生的随机唯一标识，应用数据清除将重新生成 |
| result | Int | 登录结果  0成功  1拒绝登录，失败时reason非空 |
| reason | String | result不为0时非空  表示登录失败的原因说明 |

### 响应结构说明

[参考响应结构](#_响应结构)，这里是空，null。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "pushApplyId": "{{pushApplyId}}",  "userToken": "{{userToken}}",  "deviceToken": "{{deviceToken}}",  "result": 0,  "reason": "{{reason}}"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/app/login/confirm/1CKV12 -H 'Content-Type:application/json' -d '{  "pushApplyId":"{{pushApplyId}}",  "userToken":"{{userToken}}",  "deviceToken":"{{deviceToken}}",  "result":0,  "reason":"{{reason}}"  }' | |

## 消息通知确认

仅针对android设备，收到消息推送后调用该接口，确认已收到消息，避免重复接收。

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/push/certcontent/app/ login/message/confirm/1CKV12 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| deviceToken | String | ios设备上app对应的固定唯一标识  安卓设备上app产生的随机唯一标识，应用数据清除将重新生成 |
| userType | Int | 登录申请id，登录申请接口的响应值 |
| authorizedUserId | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空。 |
| cert | Cert | 参考12.1.1[cert](#_响应结构说明)结构 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "deviceToken": "{{deviceToken}}",  "userType": 1,  "authorizedUserId": "{{authorizedUserId}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": null  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/push/certcontent/app/login/ message/confirm/1CKV12 -H 'Content-Type:application/json' -d '{  "deviceToken":"{{deviceToken}}"，  "userType":"{{userType}}"，  "authorizedUserId":"{{authorizedUserId}}"，  "cert":{ "certContent":"{{certContent}}"}  }' | |

# 外部系统接口



## 机构信息注册

用于向云密钥系统添加机构信息，添加成功后返回该机构在云密钥系统的唯一标识organizationId，业务系统自行保存。系统会根据机构名称+证件类型+证件号码判断机构是否存在，已存在直接返回organizationId，不存在则添加成功后返回organizationId。

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/system/interface/org/register/1CKV11 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authId | String | 接入系统的标识，由云密钥分配 |
| authType | Int | 认证类型，1表示签名验签方式，目前该接口只有1 |
| authCode | String | 预留字段，暂时没用 |
| requestInfoEncode | String | 请求原文节点json字符串的base64编码 |
| systemSignature | String | 对requestInfoEncode的签名值 |
| cloudkeyProjectId | String | 预留字段，暂时没用 |
| organization | Organization | 机构信息，参考12.1.4[Organization结构](#_Organization结构) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents内容如下。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| organizationId | Int | 机构注册成功后返回的机构id，唯一 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "authId": "cloudkey4sz",  "authType": 1,  "authCode": "",  "requestInfoEncode": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQ",  "systemSignature": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQyUwQSUwOSUw"  }  requestInfoEncode：对请求原文节点json字符串的base64编码  systemSignature：对请求原文节点json字符串的签名值的base64编码，使用utf-8编码。  请求原文节点json字符串示例：  {  "cloudkeyProjectId": "",  "organization": {  "name": "",  "organizationType": 1,  "identityType": 501,  "identity": "123456",  "phone": "38861212",  "officialResidence": "广东省广州",  "email": "",  "memo": "",  "countryName": "CN",  "province": "Guangdong",  "legalPersonName": "张三",  "legalPersonPhone": "13512345678",  "legalPersonIdentityType": 2,  "legalPersonIdentity": "123456"  }  } | authId:接入系统的标识，由云密钥分配  authType:认证类型，1表示签名验签方式，目前只有1  authCode:预留字段，暂时没用  cloudkeyProjectId:预留字段，暂时没用  organization节点下的信息：  name：机构名称，必传  organizationType：机构类型，必传  identityType：证件类型，必传  identity：证件号码，必传  phone：联系电话，必传  officialResidence：法定地址，必传  email：邮箱，可选  memo：备注，可选  countryName：国家，必传，只能为 CN  province：省份拼音，必传  legalPersonName：法人代表名称，可选  legalPersonPhone：法人电话，可选  legalPersonIdentityType：法人证件类型，可选  legalPersonIdentity：法人证件号码，可选 |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "organizationId":1  }  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/system/interface/org/register/1CKV11 -H 'Content-Type:application/json' -d '{  "authId":"{{authId}}",  "authType":"{{authType}}",  "authCode":"{{authCode}}",  "requestInfoEncode":"{{requestInfoEncode}}",  "systemSignature":"{{systemSignature}}"  }' |  |

## 用户信息注册

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/system/interface/user/register/1CKV11 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authId | String | 接入系统的标识，由云密钥分配 |
| authType | Int | 认证类型，1表示签名验签方式，目前该接口只有1 |
| authCode | String | 预留字段，暂时没用 |
| requestInfoEncode | String | 请求原文节点json字符串的base64编码 |
| systemSignature | String | 对requestInfoEncode的签名值 |
| cloudkeyProjectId | String | 预留字段，暂时没用 |
| userType | Int | 用户类型  1表示员工用户信息  2表示个人用户信息 |
| user | User | 机构信息，参考12.1.3[User结构](#_User结构) |

### 响应结构说明

[参考响应结构](#_响应结构)，若非自动签发证书，contents内容为空，若为自动签发证书，返回证书内容，结构如下。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signCert | String | 签名证书base64编码 |
| encCert | String | 加密证书base64编码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "authId": "cloudkey4sz",  "authType": 1,  "authCode": "",  "requestInfoEncode": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQ",  "systemSignature": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQyUwQSUwOSUw"  }  requestInfoEncode：对请求原文节点json字符串的base64编码  systemSignature：对请求原文节点json字符串的签名值的base64编码，使用utf-8编码。  请求原文节点json字符串示例：  occupation：职业，可选  {  "cloudkeyProjectId": "",  "userType": 1,  "user": {  "organizationId": 1,  "uid": "4000",  "name": "李二",  "phone": "13512345678",  "email": "lier@cnca.net",  "countryName": "CN",  "province": "Guangdong",  "city": "Guangzhou",  "department": "市场部",  "occupation": "医生",  "officialResidence": "广东省广州市",  "gender": 1,  "identityType": 2,  "identity": "12345",  "memo": "备注信息"  }  } | authId:接入系统的标识，由云密钥分配  authType:认证类型，1表示签名验签方式，目前只有1  authCode:预留字段，暂时没用  cloudkeyProjectId:预留字段，暂时没用  userType：用户类型，1表示员工用户信  息，2表示个人用户信息  user节点下的信息：  organizationId:机构id，必传。调用机构信息注册接口的返回值，由调用者自行保存  uid：用户唯一标识，必传  name：用户名，必传  phone：手机号码，必传  email：邮箱，可选  countryName：国家，必传，只能为 CN  province：省份的拼音，必传  city：城市的拼音，可选  department：部门，可选  occupation：职业，可选  officialResidence：办公地址，可选  gender：性别，0男1女，必传  identityType：证件类型，必传  identity：证件号码，必传  memo：备注，可选 |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "certContentInfo": {  "signCert": "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj",  "encCert": "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj"  }  }  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/system/interface/user/register/1CKV11 -H 'Content-Type:application/json' -d '{  "authId":"{{authId}}",  "authType":"{{authType}}",  "authCode":"{{authCode}}",  "requestInfoEncode":"{{requestInfoEncode}}",  "systemSignature":"{{systemSignature}}"  }' |  |

## 用户解锁

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/system/interface/user/unlock/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authId | String | 接入系统的标识，由云密钥分配 |
| authType | Int | 认证类型，1表示签名验签方式，目前该接口只有1 |
| authCode | String | 预留字段，暂时没用 |
| requestInfoEncode | String | 请求原文节点json字符串的base64编码 |
| systemSignature | String | 对requestInfoEncode的签名值 |
| cloudkeyProjectId | String | 预留字段，暂时没用 |
| userUid | String | 用户uid |
| name | String | 用户名称 |
| cert | Cert | 证书对象，参考[Cert](#_响应结构说明) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents内容如下。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| phone | String | 发送解锁PIN码的目标手机号码 |
| userPin | String | 根据配置决定是否返回解锁后的PIN码  若返回，则是一个经过authId对应证书加密过的base64编码的字符串；否则使用短信下发解锁密码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "authId": "cloudkey4sz",  "authType": 1,  "authCode": "",  "requestInfoEncode": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQ",  "systemSignature": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQyUwQSUwOSUw"  }  requestInfoEncode：对请求原文节点json字符串的base64编码  systemSignature：对请求原文节点json字符串的签名值的base64编码，使用utf-8编码。  请求原文节点json字符串示例：  {  "cloudkeyProjectId": "CKP201907121647500ECD1C37",  "userUid": "4000",  "name": "李二",  "cert": {  "certContent": "MIIFKTCCAxGgAwIBAgIQIpsPKwiWFPvWytebllITtDANBgkqhkiG9w0BAQsFADBP"  }  } | userUid：用户唯一标识，必传  name：用户名，必传  cert：证书内容，必传 |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "phone": "135\*\*\*\*4567",  "userPin": "MIIFKT"  }  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/system/interface/user/unlock/2CKV6 -H 'Content-Type:application/json' -d '{  "authId":"{{authId}}",  "authType":"{{authType}}",  "authCode":"{{authCode}}",  "requestInfoEncode":"{{requestInfoEncode}}",  "systemSignature":"{{systemSignature}}"  }' |  |

## 上传签章图片

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/system/interface/seal/certcontent/uploadsealpic/2CKV7 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authId | String | 接入系统的标识，由云密钥分配 |
| authType | Int | 认证类型，1表示签名验签方式，目前该接口只有1 |
| authCode | String | 预留字段，暂时没用 |
| sealPic | String | 签章图片base64编码 |
| requestInfoEncode | String | 请求原文节点json字符串的base64编码 |
| systemSignature | String | 对requestInfoEncode的签名值 |

### 响应结构说明

[参考响应结构](#_响应结构)，这里contents内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "authId": "cloudkey4sz",  "authType": 1,  "authCode": "",  "sealPic": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQyUwQSUwOSU",  "requestInfoEncode": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQ",  "systemSignature": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQyUwQSUwOSUw"  }requestInfoEncode：对请求原文节点json字符串的base64编码  systemSignature：对请求原文节点json字符串的签名值的base64编码，使用utf-8编码。  请求原文节点json字符串示例：  {  "sealPicHash": "1338798AD123B",  "cert": {  "certContent": "MIIFKTCCAxGgAwIBAgIQIpsPKwiWFPvWytebllITtDANBgkqhkiG9w0BAQsFADBP"  }  } | authId:接入系统的标识，由云密钥分配  authType:认证类型，1表示签名验签方式，目前只有1  authCode:预留字段，暂时没用  sealPicHash：签章图片sha256摘要的16进制编码，大写 |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": null  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/system/interface/seal/certcontent/uploadsealpic/2CKV7 -H 'Content-Type:application/json' -d '{  "authId":"{{authId}}",  "authType":"{{authType}}",  "authCode":"{{authCode}}",  "sealPic":"{{sealPic}}",  "requestInfoEncode":"{{requestInfoEncode}}",  "systemSignature":"{{systemSignature}}"  }' |  |

## 证书续期

**该接口未实现。实现版本待定。**

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/system/interface/user/certcontent/renewal/2CKV8 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authId | String | 接入系统的标识，由云密钥分配 |
| authType | Int | 认证类型，1表示签名验签方式，目前该接口只有1 |
| authCode | String | 预留字段，暂时没用 |
| requestInfoEncode | String | 请求原文节点json字符串的base64编码 |
| systemSignature | String | 对requestInfoEncode的签名值 |

### 响应结构说明

[参考响应结构](#_响应结构)，若非自动签发证书，contents内容为空，若为自动签发证书，返回证书内容，结构如下。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signCert | String | 签名证书base64编码 |
| encCert | String | 加密证书base64编码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "authId": "cloudkey4sz",  "authType": 1,  "authCode": "",  "requestInfoEncode": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQ",  "systemSignature": "M0ElMjAlMjJjbG91ZGtleTRzeiUyMiUyQyUwQSUwOSUw"  }  requestInfoEncode：对请求原文节点json字符串的base64编码  systemSignature：对请求原文节点json字符串的签名值的base64编码，使用utf-8编码。  请求原文节点json字符串示例：  {  "cert": {  "certContent": "MIIFKTCCAxGgAwIBAgIQIpsPKwiWFPvWytebllITtDANBgkqhkiG9w0BAQsFADBP"  }  } | authId:接入系统的标识，由云密钥分配  authType:认证类型，1表示签名验签方式，目前只有1  authCode:预留字段，暂时没用  cert：证书结构，参考[Cert结构](#_响应结构说明) |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "certContentInfo": {  "signCert": "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj",  "encCert": "MIIE0zCCA7ugAwIBAgILEIpXnFSlXFAYj"  }  }  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/system/interface/user/certcontent/renewal/2CKV8 -H 'Content-Type:application/json' -d '{  "authId":"{{authId}}",  "authType":"{{authType}}",  "authCode":"{{authCode}}",  "requestInfoEncode":"{{requestInfoEncode}}",  "systemSignature":"{{systemSignature}}"  }' |  |

## 业务通知

接收通知的应用方需实现该接口。若调用接口通知失败，会重试3次，3次都失败则不再进行通知。

### 请求消息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | 由接收通知的应用方自行确定 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

#### 请求参数结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| authId | String | 预留字段，暂时无用 |
| authType | Int | 认证类型，2表示数字信封加密，目前该接口只有2 |
| authCode | String | 预留字段，暂时没用 |
| requestInfoEncryption | String | 请求原文节点json字符串的数字信封加密的base64编码 |

#### 原文请求参数结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 通知所属的业务类型  0表示密钥产生，包含证书初始密码  1表示证书签发  2表示证书解锁，包含证书解锁后的密码  3表示授权录入，待证书用户确认  4表示授权完成，对证书用户的通知  5表示授权完成，对被授权用户的通知，包含被授权用户的初始使用密码  6表示短信验证码的通知，包括证书用户或被授权用户的验证码通知  7表示授权修改的通知，修改授权信息后会进行通知  8表示证书续期成功的通知  9表示证书注销成功的通知 |
| contents | 结构体 | 根据requestType有不同的类型 |
| thirdSystemLogId | String | 消息id，预留字段 |

##### 密钥产生通知（包含证书初始密码）

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 0，表示当前的通知是密钥产生 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| request | Request | 云密钥业务信息，参见 12.1.16[Request](#_12.1.16._Request结构)结构 |
| pin | String | 证书初始使用pin码 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 0,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "用户信息备注",  "pin": "123456",  "request": {  "requestId": "CKR20191115161225CD5ED7F9"  }  }  } |  |

##### 2.2 证书签发通知

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 1，表示当前的通知是证书签发 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| request | Request | 云密钥业务信息，参见 12.1.16[Request](#_12.1.16._Request结构)结构 |
| certInfos | CertInfo | 证书内容，见 12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 1,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "用户备注信息",  "request": {  "requestId": "CKR20191115161225CD5ED7F9"  },  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ]  }  } |  |

##### 2.3 证书解锁（包含证书解锁密码）

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 2，表示当前的通知是证书解锁 |
| userType | Int | 0表示证书用户，即证书用户解锁  1表示被授权用户，即被授权用户解锁 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| pin | String | 证书解锁密码 |
| request | Request | 云密钥业务信息，参见 [Request](#_12.1.16._Request结构)结构 |
| certInfos | CertInfo | 证书内容，见12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |
| authorizedUser | AuthorizedUser | 授权用户信息,见12.1.11.6[AuthorizedUser](#_12.1.11.6_AuthorizedUser结构6)结构6 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | userType为0时：  {  "requestType": 2,  "thirdSystemLogId": 789,  "contents": {  "userType": 1,  "uid": "1234",  "name": "张三",  "memo": "用户信息备注",  "pin": "123456",  "request": {  "requestId": "CKR20191115161225CD5ED7F9"  },  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ]  }  }  userType为1时  {  "requestType": 2,  "thirdSystemLogId": 789,  "contents": {  "userType": 1,  "pin": "123456",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ],  "authorizedUser": {  "id": 1,  "uid": "123",  "name": "李四"  }  }  } |  |

##### 2.4 授权录入

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 3，表示当前的通知是授权录入 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| certInfos | CertInfo | 证书内容，见 12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |
| authorizedUser | AuthorizedUser | 授权用户信息，见12.1.11.3[authorizedUser](#_12.1.11.1_AuthorizedUser结构3)结构3 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 3,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "备注",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ]  },  "authorizedUser": {  "id": 1,  "uid": "123",  "name": "李四",  "memo": "备注信息",  "applicationTime": "2019-07-05 11:05:35",  "validStart": "2019-07-05 11:05:35",  "validEnd": "2019-07-15 11:05:35",  "allowCallCount": 10  }  } |  |

##### 2.5 授权完成（对证书用户的通知）

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 4，表示当前的通知是授权完成，对证书用户的通知 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| certInfos | CertInfo | 证书内容，见12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |
| authorizedUser | AuthorizedUser | 授权用户信息，见12.1.11.4[AuthorizedUser](#_12.1.11.4_AuthorizedUser结构4) 结构4 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 4,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "备注信息",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ],  "authorizedUser": [{  "id": 1,  "uid": "123",  "name": "李四",  "memo": "备注信息",  "applicationTime": "2019-07-05 11:05:35",  "confirmTime": "2019-07-06 11:05:35",  "validStart": "2019-07-06 11:05:35",  "validEnd": "2019-07-15 11:05:35",  "allowCallCount": 10  }]  }  } |  |

##### 2.6 授权完成（对被授权用户的通知，包含被授权人使用证书的初始密码）

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 5，表示当前的通知是授权完成，对被授权人的通知 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| certInfos | CertInfo | 证书内容，见 12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |
| authorizedUser | AuthorizedUser | 授权用户信息，见12.1.11.5[AuthorizedUser](#_12.1.11.4_AuthorizedUser结构5)结构5 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 5,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "备注信息",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ],  "authorizedUser": [{  "id": 1,  "uid": "123",  "name": "李四",  "memo": "备注信息",  "pin": "123456",  "applicationTime": "2019-07-05 11:05:35",  "confirmTime": "2019-07-06 11:05:35",  "validStart": "2019-07-06 11:05:35",  "validEnd": "2019-07-15 11:05:35",  "allowCallCount": 10  }]  }  } |  |

##### 2.7 短信验证码通知

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 6，表示当前的通知是短信验证码通知 |
| userType | Int | 0表示证书用户，即证书用户通知  1表示被授权用户，即被授权用户通知 |
| validCodeUsageType | Int | 1表示登录验证类型  2表示解锁验证类型 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| certInfos | CertInfo | 证书内容，见 12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |
| authorizedUser | AuthorizedUser | 授权用户信息,见12.1.11.6[AuthorizedUser](#_12.1.11.6_AuthorizedUser结构6)结构6 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | userType为0时：  {  "requestType": 6,  "thirdSystemLogId": 789,  "contents": {  "userType": 0,  "validCodeUsageType": 1,  "uid": "1234",  "name": "张三",  "validCode": "123456",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ]  }  }  userType为1时：  {  "requestType": 6,  "thirdSystemLogId": 789,  "contents": {  "userType": 1,  "validCodeUsageType": 1,  "validCode": "123456",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ],  "authorizedUser": {  "id": 1,  "uid": "123",  "name": "李四"  }  }  } |  |

##### 2.8 授权修改

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 7，表示当前的通知是授权修改，对被授权人的通知 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| certInfos | CertInfo | 证书内容，见 12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |
| authorizedUser | AuthorizedUser | 授权用户信息，见12.1.11.5[AuthorizedUser](#_12.1.11.4_AuthorizedUser结构5)结构5 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  |  |  |

##### 2.9 证书续期成功通知

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 8，表示当前的通知是证书续期成功 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| request | Request | 云密钥业务信息，参见12.1.16 [Request](#_12.1.16._Request结构)结构 |
| certInfos | CertInfo | 证书内容，见12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 8,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "备注信息",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ],  "authorizedUser": [{  "id": 1,  "uid": "123",  "name": "李四",  "memo": "备注信息",  "pin": "123456",  "applicationTime": "2019-07-05 11:05:35",  "confirmTime": "2019-07-06 11:05:35",  "validStart": "2019-07-06 11:05:35",  "validEnd": "2019-07-15 11:05:35",  "allowCallCount": 10  }]  }  } |  |

##### 2.10 证书注销成功通知

请求结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| requestType | Int | 9，表示当前的通知是证书注销成功 |
| uid | String | 用户唯一标识 |
| name | String | 用户名 |
| memo | String | 备注信息 |
| request | Request | 云密钥业务信息，参见12.1.16 [Request](#_12.1.16._Request结构)结构 |
| certInfos | CertInfo | 证书内容，见12.1.7[CertInfo](#_12.1.7_certInfos结构)结构 |

请求示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "requestType": 9,  "thirdSystemLogId": 789,  "contents": {  "uid": "1234",  "name": "张三",  "memo": "备注信息",  "certInfos": [{  "certContent": "MIICgzCCAiqgAwIBAg",  "certUsage": 2  },  {  "certContent": "MIIChT111C",  "certUsage": 1  }  ],  "authorizedUser": [{  "id": 1,  "uid": "123",  "name": "李四",  "memo": "备注信息",  "pin": "123456",  "applicationTime": "2019-07-05 11:05:35",  "confirmTime": "2019-07-06 11:05:35",  "validStart": "2019-07-06 11:05:35",  "validEnd": "2019-07-15 11:05:35",  "allowCallCount": 10  }]  }  } |  |

### 响应结构说明

[参考响应结构](#_响应结构)，contents内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  “authId”: “cloudkey4sz”,  “authType”: 1,  “authCode”: “”,  “requestInfoEncryption”: “M0ElMjAlMjJjbG91ZgtleTRzeiUyMiUyQ”  }  requestInfoEncryption：对请求原文节点json字符串的公钥加密的base64编码 | requestInfoEncryption的原文有有个字段requestType字段，用于标识不同的业务类型，不同的类型对应的通知内容不一样，即不同的requestType，contents内容不一样  原文例子：  { “requestType”: 1,  “thirdSystemLogId”: 789,  “contents”: {}  } |
| 返回 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: “第三方系统日志id（和推送的thirdSystemLogId一致）”  } | 0表示成功  -1表示失败 |
| curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/system/interface/business/notify/1CKV11 –H ‘Content-Type:application/json’ -d ‘{  “authId”:”{{authId}}”,  “authType”:”{{authType}}”,  “authCode”:”{{authCode}}”,  “requestInfoEncryption”:”{{requestInfoEncryption}}”  }’ |  |

# 公共接口



## 修改用户口令

身份认证通过之后即可修改。（支持证书用户，和被授权用户）

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/base64/user/pin/change |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户身份验证成功后的令牌 |
| oldPin | String | 旧口令 Base64编码值。 |
| newPin | String | 新口令 Base64编码值 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “userToken”:”{{userToken}}”,  “oldPin”:”{{pinBase64}}”,  “newPin”:”{{pinBase64}}”  } |  |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: null  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/base64/user/pin/change –H ‘Content-Type:application/json’ -d ‘{  “userToken”:”{{userToken}}”,  “oldPin”:”{{oldPinBase64}}”,  “newPin”:”{{newPinBase64}}”  }’ | |

## 获取证书所支持的身份验证类型

服务器返回**loginTypeBitValue值，客户端参考返回值来进行相应的处理**

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/certcontent/user/logintypebitvalue |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginTypeBitValue | Int | [**支持的身份验证类型**](#_身份验证类型常量(loginTypeBitValue)说明) |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “cert”:{  “certContent”:”{{certContent}}”  }  } |  |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “loginTypeBitValue”: 43  }  } | **loginTypeBitValue的值**  **参考**[**身份验证类型常量(loginTypeBitValue)说明**](#_身份验证类型常量(loginTypeBitValue)说明) |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/base64/user/pin/change –H ‘Content-Type:application/json’ -d ‘{  “cert”:{ “certContent”:”{{certContent}}”}  }’ | |

## 轮询二维码验证结果

客户端向后台查询，二维码验证是否成功

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/qrcode/status |
| **请求数据格式** | application/json |
| **请求消息** | Jsonstr |
| **应答消息** | Jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeIdentity | String | 二维码的唯一标识  参考[客户端获取认证二维码](#_客户端获取认证二维码) 和 [客户端获取绑定二维码](#_客户端获取绑定二维码) |

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| verifyStatus | Int | 0 = 验证成功。  1 = 用户未开始验证，需要继续轮询  -1 = 验证失败，有错误信息在errorMsg |
| userToken | String | 当二维码类型为验证，且验证成功，userToken有值。 |
| errorMsg | String | 验证失败时的错误提示信息 |

[参考响应结构](#_响应结构)，contents中的内容见下表。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **Value** | **说明** |
| 请求示例 | {  “qrCodeIdentity”:”{{qrCodeIdentity}}”  } |  |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “verifyStatus”: 1,  “userToken”: null,  “errorMsg”: null  }  } | **verifyStatus常量说明：**  **0 = 验证成功。**  **1 = 用户未开始验证，需要继续轮询**  **-1 = 验证失败，有错误信息在errorMsg**  **特别说明：当二维码类型为验证，且验证成功，userToken有值。** |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/qrcode/status –H ‘Content-Type: application/json’  -d ‘{  “qrCodeIdentity”:”6DB607FCE1304B26”  }’ | |

## 获取用户已经支持（开通）的身份验证类型

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/certcontent/user/regist/logintypebitvalue |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| type | Int | type用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空。 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |
| deviceId | String | 设备ID，由客户端维护的基于UUID的随机字符串，此处可默认为空字符串。不传会返回证书最新的已开通的类型，指定该参数能返回该设备上已开通的类型。 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| loginTypeBitValue | Int | 已经开通的身份验证类型 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “type”:0,  “authorizedUserId”:59,  “cert”:{  “certContent”:”{{certContent}}”  },  “deviceId”: “”  } | 设备ID，由客户端维护的基于UUID的随机字符串，此处可默认为空字符串 |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “loginTypeBitValue”: 56  }  } | **loginTypeBitValue的值**  **参考**[**身份验证类型常量(loginTypeBitValue)说明**](#_身份验证类型常量(loginTypeBitValue)说明) |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/certcontent/user/regist/logintypebitvalue –H ‘Content-Type: application/json’ –d ‘{  “type”:0,  “authorizedUserId”:59,  “cert”:{  “certContent”:””  }  }’ | |

## 根据绑定的二维码信息获取用户信息

该接口调用前，一定要先产生 绑定设备的二维码，也就是要先调用 [客户端获取绑定的二维码接口](#_客户端获取绑定手机设备的二维码接口)

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/user/info/get |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取绑定二维码](#_客户端获取绑定二维码) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userType | Int | 0=证书用户；1=被授权用户 |
| uid | String | 云密钥用户唯一标识，非空 |
| projectUid | String | 云密钥项目标识 |
| authorizedUserId | Int | 授权记录ID，可用于确定具体被授权用户 |
| certId | Int | **证书id** |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “qrCodeIdentity”:”{{qrCodeIdentity}}”  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userType": 0,  "uid": "cxy1111",  "projectUid": "123",  "authorizedUserId": null,  "certId": 75,  "cert": {  "keySN": null,  "keyType": null,  "certSN": "108A579C54A55C50188C11",  "issuer": null,  "certThumbPrints": [{  "algorithm": 16384,  "hash": "gGDHf9AcOih4oNBvdtsFH3tY2rq/Lh4CpgxeA0v4Cyc="  }],  "certContent": null  }  }  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/user/info/get –H ‘Content-Type: application/json’ –d ‘{  “qrCodeIdentity”:”{{ qrCodeIdentity }}”  }’ | |

## 根据认证二维码信息获取用户UID

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/user/uid/get |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeIdentity | String | 二维码唯一标识，参考[客户端获取认证二维码接口](#_客户端获取认证二维码接口) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| uid | String | **用户uid（云密钥用户唯一标识）** |
| projectUid | String | **云密钥项目uid（项目标识）** |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “qrCodeIdentity”:”{{qrCodeIdentity}}”  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "uid": "cxy1111",  "projectUid": "123"  }  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/user/uid/get –H ‘Content-Type: application/json’ –d ‘{  “qrCodeIdentity”:”{{ qrCodeIdentity }}”  }’ | |

## 电子P7签名

用户身份验证通过后，获取到userToken，即可调用该接口进行P7签名。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/pki/signdata |
| **请求数据格式** | application/json |
| **请求消息** | **jsonStr** |
| **应答消息** | **jsonStr** |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| originData | String | 待P7签名的原文(**Base64编码**) |
| detached | bool | P7签名是否**不带原文**  False表示带原文  True表示**不带原文** |
| useSubjectKeyId | bool | 是否使用主体密钥标志符，默认为false |
| algo | Int | 使用的[签名算法](#_签名算法)，见接口常量说明 |
| includeCertOption | Int | 包含证书的标识，目前**仅能包含自身签名证书，**  1 不包含签名证书  2 包含签名证书 |
| useQ7 | bool | 是否使用国密signedData, false表示不使用。 |
| useTimestamp | bool | 是否携带签名时间戳，默认 false，不携带时间戳 |
| userToken | String | 用户身份验证成功后的令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signedData | String | P7签名值(**Base64编码**) |

### 请求响应简单示例

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “userToken”: “{{userToken}}”,  “originData”: “MTIzNDU2”,  “detached”: false,  “algo”: 4,  “includeCertOption”: 2,  “useQ7”: false,  “useTimestamp”: true  } | **originData：原文的base64编码文本** |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “signedData”: “MIIWLgYJKoZIhvcNEzWI”  }  } | **signedData 为p7签名值的base64编码** |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/pki/signdata –H ‘Content-Type: application/json’ –d ‘{  “userToken”: “612F4D986F958EBCF9CAF21FBB9426EF75”,  “originData”: “MTIzNDU2”,  “detached”: false,  “algo”: 4,  “includeCertOption”: 2,  “useQ7”: false,  “useTimestamp”: true  }’ | |

## P7验签

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/pki/verifysigndata |
| **请求数据格式** | application/json |
| **请求消息** | **jsonStr** |
| **应答消息** | **jsonStr** |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| originData | String | 待P7签名的原文(**Base64编码**) |
| signedData | String | 云密钥P7签名返回签名值，见[电子P7签名](#_响应结构说明_2) |
| detached | bool | P7签名是否不带原文，用于做原文比对  False表示带原文  True表示不带原文 |
| verifyTimestamp | bool | 是否验证时间戳 |
| includeCert | bool | 是否对签名证书验证  用于验证签名值内的签名证书  False表示不验证签名内部的签名证书，  True表示验证签名内部的签名证书 |
| certContent | String | 云密钥返回的证书(**Base64编码**)，用于P7验签以及签名值内部的签名证书的比对，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signInfo | SignInfo | 签名信息，参见12.1.22[SignInfo](#_12.1.22._SignInfo结构)结构 |

### 请求响应简单示例

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “originData”: “MTIzNDU2”,  “detached”: false,  “verifyTimestamp”:false,  “includeCert”: false,  “certContent”:””,  “signedData”:”MIIWCAYJKoZIh “  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "signingTime": "2020-11-27 15:35:00",  "tsaTime": "2020-11-27 15:37:08"  }  }  失败响应{时间戳验证失败，验签失败，中间件/系统异常}  {  “responseResult”: {  “status”: -20,  “msg”: “时间戳验证失败”  },  “contents”: null  }  {  “responseResult”: {  “status”: -20,  “msg”: “原文与签名信息不匹配”  },  “contents”: null  }  {  “responseResult”: {  “status”: -1,  “msg”: “\*\*\*\*\*\*\*”  },  “contents”: null  } | **验签失败的返回的status 为 -20**  **验签成功则返回的status 为 0** |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/pki/verifysigndata –H ‘Content-Type: application/json’ –d ‘{  “originData”: “MTIz”,  “detached”: false,  “includeCert”: false,  “verifyTimestamp”:true,  “certContent”:””,  “signedData”:”MIIG6AYJKoZIhvcN”  }’ | |

## 查询支持的Hash算法

获取云密钥系统支持的hash算法

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/hashalgo |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **key** | **String** | Hash算法常量值 |
| **value** | **String** | Hash算法名称字符串 |

### 请求响应简单示例

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  |  |  |
| 响应示例 | {     **“responseResult”**:{         **“status”**:**0**,         **“msg”**:**”成功”**     },     **“contents”**:[  {  **“key”** : **“28672”**,  **“value”** : **“SM3”** },  {  **“key”** : **“8192”**,  **“value”** : **“SHA1”** },  {  **“key”** : **“12288”**,  **“value”** : **“SHA224”** },  {  **“key”** : **“16384”**,  **“value”** : **“SHA256”** },  {  **“key”** : **“20480”**,  **“value”** : **“SHA384”** },  {  **“key”** : **“24576”**,  **“value”** : **“SHA512”** }  ]} | 返回的列表  Key表示算法的id，  Value表示对应的算法名称  Hash算法对应的key，可以查阅Netca中间件的Hash常量的取值 |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/hashalgo –H ‘Content-Type: application/json’ | |

## 查询系统版本

获取云密钥系统当前版本号

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/sysinfo |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **类型** | **约束** | **说明** |
| **version** | **String** |  | 当前云密钥版本号 |
| **openLoginPushService** | **Boolean** |  | 手机登录消息推送是否开启 |
| **openEleSignService** | **Boolean** |  | 电子签名系统是否开启 |
| **openSelfRegService** | **Boolean** |  | 自助申请服务是否开启 |
| **openUploadSealPic** | **Boolean** |  | 上传签章接口是否开启 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  |  |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "version": "1.2.0",  "openLoginPushService": true,  "openEleSignService": true,  "openSelfRegService": true  }  } | 返回的系统版本号 |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/sysinfo –H ‘Content-Type: application/json’ | |

## 查询密码重试次数

根据usertoken查询密码重试次数，证书用户、被授权用户均可使用该接口进行查询。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/user/pin/getSurplusNum |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的值为口令剩余重试次数，若已锁死返回-1。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  **“userToken”**:**”用户身份验证成功后台返回的令牌”**  } |  |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: 5  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/user/pin/getSurplusNum –H ‘Content-Type: application/json’ –d ‘{  “userToken”:”{{ userToken }}”  }’ | |

## 查询登录令牌是否有效

若响应值的responseResult中status字段为-2，表示userToken无效或已过期。

调用该接口时，首先应该先检查responseResult中status字段，然后再检查validTime和validStatus字段。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/usertoken/status |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| status | Int | status为-2，表示当前usertoken无效或已过期，参考[响应状态码常量status值](#_响应状态码常量status值)。 |
| validTime | Int | userToken剩余有效时间，单位为秒  若userToken失效或已过期，值为-1 |
| validStatus | Int | userToken状态  0表示有效，validTime值大于0  1表示无效或已过期，validTime值为-1 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  **“userToken”**:**”用户身份验证成功后台返回的令牌”**  } |  |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “validTime”: 1800,  “validStatus”: 0  }  } | 若 responseResult 中的status为-2，也表示当前usertoken无效或已过期，参考[响应状态码常量status值](#_响应状态码常量status值)。 |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/usertoken/status –H ‘Content-Type: application/json’ –d ‘{  “userToken”:”{{ userToken }}”  }’ | |

## 根据登录令牌查询操作者信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/oprinfo/search/all |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| name | String | 当前操作者姓名 |
| uid | String | 当前操作者工号 |
| projectUid | String | 当前操作者所属项目标识 |
| phone | String | 当前操作者手机号 |
| memo | String | 备注信息 |
| userType | Int | 0表示证书用户  1表示被授权用户 |
| authorizedUserId | Int | 被授权用户id，userType为0时该值为空 |
| signCert | String | 签名证书的base64编码 |
| encCert | String | 加密证书的base64编码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  **“userToken”**:**”用户身份验证成功后台返回的令牌”**  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "name": "张三",  "uid": "0001",  "projectUid": "123",  "phone": "13512345678",  "memo": "备注",  "userType": 0,  "certContentInfo":{  "signCert":"MIIFKTCCAxGgAwIBAgIQIpsPKwiWFPvWytebllITtDANBgkqhkiG9w0BAQsFADBP",  "encCert":"MIIFKTCCAxGgAwIBAgIQIpsPKwiWFPvWytebllITtDANBgkqhkiG9w0BAQsFADBP"  }  }  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/oprinfo/search/all -H ‘Content-Type: application/json’ –d ‘{  “userToken”:”{{ userToken }}”  }’ | |

## 查询用户的登录信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/logininfo/search |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| token | String | 用户登录令牌或退出令牌  可以是身份验证后拿到的登录令牌  也可以是调用[通过设备](#_1_通过设备获取退出登陆的usertoken)、[人脸认证](#_2_人脸认证获取退出登陆的usertoken)获取退出令牌拿到的退出令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| name | String | 登录用户姓名 |
| uid | String | 登录用户工号 |
| projectUid | String | 登录用户所属项目标识 |
| userType | Int | 0表示证书用户  1表示被授权用户 |
| authorizedUserId | Int | 被授权用户id，userType为0时该值为空 |
| total | Int | 登录信息数量 |
| currentRowsSize | Int | Rows数组的长度，当前与total一致，为后续可能的分页查询预留 |
| rows | 数组 | 参见12.1.14.3[Row](#_12.1.14.3_Row结构3)结构3 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  **“token”**:**”用户身份验证成功后台返回的令牌”**  } |  |
| 响应示例 | {  "responseResult ": {  "status ": 0,  "msg": "成功"  },  "contents": {  "name ": "张三 ",  "uid": "0001",  "projectUid": "123456",  "userType": 0,  "total": 10,  "currentRowsSize": 10,  "rows": [{  "loginUuid ": "111111111111 ",  "loginTime": "2019 - 08 - 14 08: 49: 10",  "current": true,  "applicationId": "YaofangHis",  "applicationName": "药房his系统"  }]  }  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/api/logininfo/search -H ‘Content-Type: application/json’ –d ‘{  “token”:”{{token}}”  }’ | |

## 查询证书所属项目uid（projectUid）

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/api/certcontent/projectinfo/search/2CKV2 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考12.1.1.[Cert结构](#_响应结构说明) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| projectUid | String | 证书所属项目标识 |
| userUid | String | 证书所属用户的标识 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "cert":{  "certContent":"{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "projectUid": "123",  "userUid": "456"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/cert/certcontent/projectinfo/search -H 'Content-Type:application/json' -d '{  "cert":{ "certContent":"{{certContent}}"}  }' | |

## 获取图形验证码

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/generate/imgvalidcode/2CKV3 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

无需请求参数

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| imgValidCodeIdentity | String | 图形验证码标识 |
| img | String | 图形验证码的图片base64编码 |
| needVerify | Boolean | 是否需要校验图形验证码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "imgValidCodeIdentity": "2C9CE167",  "img": "",  "needVerify": true  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/generate/imgvalidcode/2CKV3 | |

## 自助解锁

### 9.18.1 获取短信验证码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/selfunlock/certcontent/smsvalidcode/2CKV3 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userUid | String | 证书用户唯一标识 |
| name | String | 用户名 |
| cert | Cert | 参考12.1.1.[Cert结构](#_响应结构说明) |
| imgValidCodeIdentity | String | 图形验证码标识，调用[9.17 获取图形验证码](#_获取图形验证码)接口获得 |
| imgValidCode | String | 图形验证码，调用[9.17 获取图形验证码](#_获取图形验证码)接口获得图形验证码图片，展示后由用户输入 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| smsCodeGenMinCycle | Int | 下次能产生验证码的时间间隔（单位秒） |
| phone | String | 手机号码，中间4位为\*号  如135\*\*\*\*4567 |
| phoneValidCode | String | 如果配置通过短信下发验证码，该值为空  如果配置不通过短信下发验证码，该值不为空 |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userUid": "{{userUid}}",  "name": "{{name}}",  "cert": {  "certContent": "{{certContent}}"  },  "imgValidCodeIdentity": "2C9CE167",  "imgValidCode": "zsf1"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "smsCodeGenMinCycle": 60,  "phone": "135\*\*\*\*4567",  "phoneValidCode": "根据配置决定是否返回"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/selfunlock/certcontent/smsvalidcode/2CKV3 -H 'Content-Type:application/json' -d '{  "userUid":"{{userUid}}",  "name":"{{name}}",  "cert":{"certContent":"{{certContent}}"}  }' | |

### 9.18.2 验证短信验证码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/selfunlock/certcontent/smsvalidcode/verify/2CKV3 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userUid | String | 用户唯一标识 |
| name | String | 用户名 |
| phoneValidCode | String | 手机短信验证码 |
| cert | Cert | 参考12.1.1.[Cert结构](#_响应结构说明) |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为空。

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userUid": "{{userUid}}",  "name": "{{name}}",  "phoneValidCode": "{{phoneValidCode}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/selfunlock/certcontent/smsvalidcode/verify/2CKV3 -H 'Content-Type:application/json' -d '{  "userUid":"{{userUid}}",  "name":"{{name}}",  "phoneValidCode":"{{phoneValidCode}}",  "cert":{"certContent":"{{certContent}}"}  }' | |

## 流程配置查询

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/process/config/list/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| platform | Int | -1表示查询全部  0表示查询ios平台  1表示查询android平台  2表示查询微信小程序平台  3表示查询钉钉小程序平台 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| platform | Int | 0表示ios平台  1表示android平台  2表示微信小程序平台  3表示钉钉小程序平台 |
| processType | Int | 流程类型，目前只有1，表示初始化流程 |
| operationSequence | Array | 操作步骤数组  1表示绑定设备  2表示上传签章 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "platform": 0  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": [{  "platform": 1,  "process": [{  "processType": 1,  "operationSequence": [1, 2]  }]  }]  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/process/config/list/2CKV6  -H 'Content-Type:application/json' -d '{  "platform": 0  }' | |

## 获取项目相关配置

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/api/project/config/2CKV7 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cert | Cert | 参考15.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| faceTypeBitValue | Int | 系统支持的人脸识别方式  参考[faceTypeBitValue](#_人脸识别类型常量（faceTypeBitValue）说明)结构 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": [{  " faceTypeBitValue":1  }]  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/project/config/2CKV7  -H 'Content-Type:application/json' -d '  {  "cert": {  "certContent": "{{certContent}}"  }  }' | |

# 签章接口



## 下载签章图片

身份认证通过之后即可调用。（支持证书用户，和被授权人）

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/seal/downloadsealpic |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **picBase64** | **String** | 签章图片的base64编码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {     **“userToken”**:**”BF2D1280F656FEC98910E12D1656D5”** } | **userToken：登录令牌。** |
| 响应示例 | {     **“responseResult”**:{         **“status”**:**0**,         **“msg”**:**”成功”**     },     **“contents”**: {  **“picBase64”** : (图片的Base64编码)  } } | **picBase64 为 签章图片编码，如果不存在，picBase64为空字符串** |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/seal/downloadsealpic –H ‘Content-Type: application/json’ –d ‘{  “userToken”:”{{ userToken }}”  }’ | |

## 上传签章图片

只有证书用户本人，才能上传更改签章图片

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/seal/uploadsealpic |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **userToken** | **String** | 用户登录令牌 |
| **picBase64** | **String** | 签章图片base64编码 |
| **hashValue** | **String** | 签章图片hash值的hex编码 |
| **hashAlgo** | **Int** | 签章图片进行[hash的算法值](#_机构类型) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {     **“userToken”**:**”BF2D1280F656FEC98910E12D15D5”,**  **“picBase64”**:**”BF2D1280F656FEC98910E12D65D5”,**  **“hashValue”**:**”BF2D1280F656FEC98910E12D16D5”,**  **“hashAlgo”**:**0**  } | **userToken登录令牌,**  **picBase64为签章图片的Base64编码，**  **hashValue为图片的hash值的hex编码**  **hashAlgo为服务器支持的hash编码的key,见**[**查询支持的Hash算法（V1.3）**](#_查询支持的Hash算法（V1.3）)**。** |
| 响应示例 | 成功：{     **“responseResult”**:{         **“status”**:**0**,         **“msg”**:**”成功”**     },     **“contents”**:{} }  失败：{     **“responseResult”**:{         **“status”**:**-10**,         **“msg”**:**”被授权用户不能修改签章图片”**     },     **“contents”**:{} } | **只有证书用户本人才能上传图片，被授权用户只能查看、使用签章**  **被授权用户调用上传签章图片接口，返回错误响应码 -10** |

## PDF同步签章

证书用户（包括授权人）登录之后，即可调用PDF同步签章功能，返回PDF签章文件流。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/seal/pdfseal |
| **请求数据格式** | application/json |
| **请求消息** | **jsonstr** |
| **应答消息** | **jsonstr** |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **pdfData** | String | PDF文件的base64编码文本 |
| **sealSize** | **SealSize** | 签章的大小，见[SealSize](#_12.1.17._SealSize结构)结构 |
| **position** | **Position** | 签章的位置，见[Position](#_12.1.18._Position结构)结构 |
| **fieldName** | String | **签名域的域名，当该参数非空有值，优先使用签名域进行签章，忽略position 和 sealSize** |
| **fieldText** | String | **签名域显示的文字，默认显示证书CN项，服务端暂不支持该参数配置** |
| **useFieldText** | bool | **是否使用签名文本，true为使用签名文本填充签名域， false为使用签章图片填充签名域， 服务端暂不支持** |
| **useTimestamp** | bool | **是否携带签名时间戳，默认 false，不携带时间戳** |
| **showSeal** | bool | **是否显示签章，默认true，显示签章** |
| **userToken** | String | **登录令牌** |
| adjustRotation | Int | **逆时针旋转角度** |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **sealPdf** | String | **签章完成的PDF的base64编码文本** |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 入参 | {  “**pdfData**”: “**JVBERi0xLjcNCiW1t**”,  “**sealSize**”: {  “**width**”: **20**,  “**height**”: **20**  },  “**position**”: {  “**x**”: **20**,  “**y”:** **20**,  “**page**”: **-1**  },  “**ieldname**”: “”,  “**fieldText**”: “”,  “**useFieldText**”: **false**,  “**useTimestamp**”: **false**，  “**showSeal**”: **false**，  “**userToken**”: “**85EA759629702CBAE11BB119FED50445505**”  } | **详见下表** |
| 返回 | {     **“responseResult”**:{         **“status”**:**0**,         **“msg”**:**”成功”**     },     **“contents”**:{  “**sealPdf**”: “**sealPdf**”:  } } | **sealPdf 是签章完成的PDF的base64编码文本** |

## 下载指定格式签章图片

身份认证通过之后即可调用。（支持证书用户，和被授权人）

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip:port/cloudkeyserver/seal/downloadsealpic/1CKV11> |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |
| format | int | 0：原样输出  1：jpg  2：bmp  3：png |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| picBase64 | String | 签章图片的base64编码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {     **“userToken”**:**”BF2D1280F656FEC98910E12D1656D5”,**    **“format”**: **0** } | **userToken：登录令牌。** |
| 响应示例 | {     **“responseResult”**:{         **“status”**:**0**,         **“msg”**:**”成功”**     },     **“contents”**: {  **“picBase64”** : (图片的Base64编码)  } } | **picBase64 为 签章图片编码，如果不存在，picBase64为空字符串** |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/seal/downloadsealpic/1CKV11 –H ‘Content-Type: application/json’ –d ‘{  “userToken”:”{{ userToken }}”,  “format”:”{{ format }}”  }’ | |

## 10.5 PDF签章（使用参数中的图片进行签章）

使用接口参数中的签章图片进行签章，返回PDF签章文件的base64编码。

### 请求信息

|  |  |
| --- | --- |
| 请求方式 | https协议 POST |
| 接口URL | <https://ip>:port/cloudkeyserver/seal/with/sealpic/1CKV13 |
| 请求数据格式 | application/json |
| 请求消息 | jsonstr |
| 应答消息 | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| pdfData | String | 待签章的PDF的base64编码文本 |
| pdfDataUuid | String | 待签章的pdf唯一标识，预留字段 |
| sealPic | String | 签章图片 |
| sealSize | SealSize | 签章的大小，见12.1.17[SealSize](#_12.1.17._SealSize结构)结构 |
| position | Position | 签章的位置，见12.1.18[Position](#_12.1.18._Position结构)结构 |
| fieldName | String | 签名域的域名，当该参数非空有值，优先使用签名域进行签章，忽略position 和 sealSize |
| fieldText | String | 签名域显示的文字，默认显示证书CN项，服务端暂不支持该参数配置 |
| useFieldText | bool | 是否使用签名文本，true为使用签名文本填充签名域， false为使用签章图片填充签名域， 服务端暂不支持 |
| useTimestamp | bool | 是否携带签名时间戳，默认 false，不携带时间戳 |
| showSeal | bool | 是否显示签章，默认true，显示签章 |
| userToken | String | 登录令牌 |
| adjustRotation | Int | **逆时针旋转角度** |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| sealPdf | String | 签章完成的PDF的base64编码文本 |
| sealPdfUuid | String | 签章完成的文件的唯一标识，预留字段，暂时没用 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 入参 | {  “pdfData”: “JVBERi0xLjcNCiW1t”,  “pdfDataUuid”: “”,  “sealPic”: “JVBERi0xLjcrasweweew==”,  “sealSize”: {  “width”: 20,  “height”: 20  },  “position”: {  “x”: 20,  “y”: 20,  “page”: -1  },  “ieldname”: “”,  “fieldText”: “”,  “useFieldText”: false,  “useTimestamp”: false,  “showSeal”: false,  “userToken”: “85EA759629702CBAE11BB119FED50445505”  } |  |
| 返回 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “sealPdf”: “sealPdf”,  “sealPdfUuid”: “”  }  } | sealPdf 是签章完成的PDF的base64编码文本 |

## 10.6 PDF验证

### 请求信息

|  |  |
| --- | --- |
| 请求方式 | https协议 POST |
| 接口URL | <https://ip>:port/cloudkeyserver/seal/verify/1CKV13 |
| 请求数据格式 | application/json |
| 请求消息 | jsonstr |
| 应答消息 | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| pdfData | String | pdf的base64编码 |
| verifyLevel | int | PDF验证等级，验证过程根据验证级别由低到高验证 0：”校验签名(PDF原文完整性)” 1：”校验签名时间及更低验证等级” 2：”校验可信证书链及更低验证等级” 3：”校验签名时OCSP响应及更低验证等级”  e.g当该参数指定为2时，将执行0、1、2的验证。  其中，可信证书链在服务器端统一配置，默认为NETCA正式运营证书链  签名时间，默认使用PDF签章时间或者可信时间戳时间 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| 参数名 | 类型 | 说明 |
| globalState | Int | pdf验证状态，0为验证成功，1为验证失败 |
| signFieldInfos | Array | 签名域信息，详见[12.1.19 SignFieldInfo](#_12.1.18._SignFieldInfo结构)结构 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  “pdfData”: “M0ElMjAlMjJjbG91ZgtleTRzeiUyM”  } |  |
| 返回 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “globalState”: 1,  “signFieldInfos”: [{  “signFieldName”: “签名域名称”,  “validity”: {  “state”: 1,  “faultReason”: {  “code”: “”,  “desc”: “”  }  },  “signInfo”: {  “signer”: “签名者名称，CN”,  “signingTime”: “2019-09-09 09:09:09”,  “tsaTime”: “2019-09-09 09:09:09”  },  “certContent”: “”  }]  }  } |  |

## 10.7 根据UID下载签章图片

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/seal/downloadsealpic/uid/2CKV1 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userUid | String | 云密钥用户唯一标识 |
| projectUid | String | 云密钥项目标识 |
| format | int | 0：原样输出  1：jpg  2：bmp  3：png |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| picBase64 | String | 签章图片的base64编码 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userUid": "{{uid}}",  "projectUid": "{{projectUid}}",  "format": {{format}}  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "picBase64":"签章图片的base64编码"  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61/cloudkeyserver/seal/downloadsealpic/uid/2CKV1 -H 'Content-Type:application/json' -d '{  "userUid": "{{userUid}}",  "projectUid": "{{projectUid}}",  "format": {{format}}  }' | |

# 退出登录



## 通过设备获取退出令牌

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/logout/certcontent/logouttoken/device |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| deviceId | String | 设备的唯一标识 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| logoutToken | String | 用于退出登录的令牌  使用该接口获取到的退出令牌，只能调用以下接口：   1. [退出所有应用登录](#_3_使用usertoken退出登陆) 2. [退出指定应用登录](#_4_退出指定应用登录) 3. [查询用户的登录信息](#_15_根据usertoken查询用户所有的登陆信息) |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “deviceId”: “ASDFGHJ设备唯一标识”,  “cert”: {  “certContent”: “{{certContent}}”  }  } |  |
| 响应示例 | {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “logoutToken”: “679401E9D0553FE72A152773”  }  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/logout/certcontent/logouttoken/device -H ‘Content-Type: application/json’ –d ‘{  “deviceId”:”{{deviceId}}”,  “cert”:{“certContent”:”{{certContent}}”}  }’ | |

## 人脸认证获取退出令牌

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/logout/certcontent/logouttoken/facedata |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| faceType | Int | 人脸类型：0=人脸识别（目前只有0） |
| faceData | String | 人脸图片数据的Base64字符串 |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空。 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| logoutToken | String | 用于退出登录的令牌  使用该接口获取到的退出令牌，只能调用以下接口：   * 1. [退出所有应用登录](#_3_使用usertoken退出登陆)   2. [退出指定应用登录](#_4_退出指定应用登录)   3. [查询用户的登录信息](#_15_根据usertoken查询用户所有的登陆信息) |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “faceType”: 0,  “faceData”: “{{faceData}}”,  “userType”: 1,  “authorizedUserId”:{{authorizedUserId}},  “cert”: {  “certContent”: “{{certContent}}”  }  } | faceData  是客户端拍照获取的用户人脸照片数据，base64编码 |
| 响应示例 | 成功响应  {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: {  “logoutToken”: “24B9DE6AED996752843C0C514D4D791075#59”  }  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/logout/certcontent/logouttoken/facedata –H ‘Content-Type:application/json’ -d ‘{  “faceType”: 0,  “faceData”: “{{faceData}}”,  “userType”: 1,  “authorizedUserId”:{{authorizedUserId}},  “cert”:{ “certContent”:”{{certContent}}”}  }’ | |

## 退出所有应用登录

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/logout/logouttoken |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| logoutToken | String | 用户登录令牌或退出令牌  可以是身份验证后拿到的登录令牌  也可以是调用[通过设备](#_1_通过设备获取退出登陆的usertoken)、[人脸认证](#_2_人脸认证获取退出登陆的usertoken)获取退出登录令牌拿到的退出令牌 |

### 响应结构说明

[参考响应结构](#_响应结构)，这里contents为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “logoutToken”: “24B9DE6AED996752843C0C514D4D79105”  } |  |
| 响应示例 | 成功响应  {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: null  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/logout/logouttoken –H ‘Content-Type:application/json’ -d ‘{  “logoutToken”: “{{logoutToken}}”  }’ | |

## 退出指定应用登录

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | <https://ip>:port/cloudkeyserver/logout/loginuuid |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| logoutToken | String | 用户登录令牌或退出令牌  可以是身份认证后拿到的登录令牌  也可以是调用通过设备、刷脸获取退出令牌拿到的退出令牌 |
| loginUuid | String | 登录信息uuid，通过[查询用户登录信息](#_15_查询用户的登录信息)获取 |

### 响应结构说明

[参考响应结构](#_响应结构)，这里contents为空。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  “logoutToken”: “24B9DE6AED996752840C514D4D791075”,  “loginUuid”: “DE6AED996752843C0C5”  } |  |
| 响应示例 | 成功响应  {  “responseResult”: {  “status”: 0,  “msg”: “成功”  },  “contents”: null  } |  |
| Curl示例 | curl –kX POST <https://192>.168.200.61:443/cloudkeyserver/logout/loginUuid –H ‘Content-Type:application/json’ -d ‘{  “loginUuid”: “{{loginUuid}}”,  “logoutToken”: “{{logoutToken}}”  }’ | |

# app服务初始化

## 查询优先展示的服务信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/serviceinfo/prior/list/2CKV5 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| serviceName | String | 需要搜索的服务信息关键字，为空则查询全部 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| total | Int | 数据总数 |
| name | String | 服务名称 |
| serviceInfo | String | 服务配置信息，与产生初始化二维码的字符串一致 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "serviceName": "中山"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "total": 1,  "currentRowsSize":1,  "rows" :[  {  "name":"中山附一",  "serviceInfo":"netcack0://{\"u\":[\"$192.168.111.124:8443\",\"#192.168.1.25:7443\"],\"p\":\"xxxx\"}"  }  ]  }  } |  |
| Curl示例 | curl –kX POST https://192.168.200.61:443/cloudkeyserver/serviceinfo/prior/list/2CKV5 –H ‘Content-Type:application/json’ | |

## 分页查询服务信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/serviceinfo/list/2CKV5 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| serviceName | String | 需要搜索的服务信息关键字，为空则查询全部 |
| pageIndex | Int | 分页页码，第几页 |
| pageSize | Int | 分页显示数量 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| total | Int | 数据总数量 |
| currentRowsSize | Int | 当前返回数据的总数量 |
| name | String | 服务名称 |
| serviceInfo | String | 服务配置信息，与产生初始化二维码的字符串一致 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "serviceName": "中山",  "pageIndex": 1,  "pageSize": 10  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "total":1,  "currentRowsSize":1,  "rows" :[  {  "name":"中山附一",  "serviceInfo":"netcack0://{\"u\":[\"$192.168.111.124:8443\",\"#192.168.1.25:7443\"],\"p\":\"xxxx\"}"  }  ]  }  } |  |
| Curl示例 | curl –kX POST https://192.168.200.61:443/cloudkeyserver/serviceinfo/list/2CKV5 –H ‘Content-Type:application/json’ -d ‘{  “pageIndex”: “{{pageIndex}}”,  “pageSize”: “{{pageSize}}”  }’ | |

# 自助服务

## 自助申请

### 获取短信验证码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/register/smsvalidcode/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| cloudkeyProjectId | String | 云密钥项目id，必传 |
| userType | Int | 1表示机构员工用户，目前只有1，必传 |
| organizationId | Int | 用户所属机构的id，必传 |
| uid | String | 用户唯一标识，必传 |
| name | String | 用户名，必传 |
| phone | String | 手机号码，必传 |
| email | String | 邮箱，可选 |
| countryName | String | 国家，必传，目前只有CN |
| province | String | 省份的拼音，必传 |
| city | String | 城市的拼音，必传 |
| department | String | 部门，可选 |
| occupation | String | 职业，可选 |
| officialResidence | String | 办公地址，可选 |
| gender | Int | 性别，0男1女，必传 |
| identityType | Int | 证件类型，必传 |
| identity | String | 证件号码，必传 |
| memo | String | 备注，可选 |
| imgValidCodeIdentity | String | 图形验证码标识，调用[9.17 获取图形验证码](#_获取图形验证码)接口获得 |
| imgValidCode | String | 图形验证码，调用[9.17 获取图形验证码](#_获取图形验证码)接口获得图形验证码图片，展示后由用户输入 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| smsCodeGenMinCycle | Int | 下次能产生验证码的时间间隔（单位秒） |
| phone | String | 手机号码，中间4位为\*号  如135\*\*\*\*4567 |
| phoneValidCode | String | 如果配置通过短信下发验证码，该值为空  如果配置不通过短信下发验证码，该值不为空 |

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "imgValidCodeIdentity": "AC128DE",  "imgValidCode": "xs3d",  "cloudkeyProjectId": "",  "userType": 1,  "user": {  "organizationId": 1,  "uid": "4000",  "name": "李二",  "phone": "13512345678",  "email": "lier@cnca.net",  "countryName": "CN",  "province": "Guangdong",  "city": "Guangzhou",  "department": "市场部",  "occupation":"医生",  "officialResidence": "广东省广州市",  "gender": 1,  "identityType": 2,  "identity": "12345",  "memo": "备注信息"  }  } | cloudkeyProjectId:云密钥项目id  userType：用户类型，目前只有1，表示员工用户  user节点下的信息：  organizationId:机构id，必传。调用机构信息注册接口的返回值，由调用者自行保存  uid：用户唯一标识，必传  name：用户名，必传  phone：手机号码，必传  email：邮箱，可选  countryName：国家，必传，只能为 CN  province：省份的拼音，必传  city：城市的拼音，必传  department：部门，可选  occupation：职业，可选  officialResidence：办公地址，可选  gender：性别，0男1女，必传  identityType：证件类型，必传  identity：证件号码，必传  memo：备注，可选 |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "smsCodeGenMinCycle": 60,  "phone": "135\*\*\*\*4567",  "phoneValidCode": "根据配置决定是否返回"  }  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/selfregister/smsvalidcode/2CKV6 -H 'Content-Type:application/json' -d ' {  "imgValidCodeIdentity":"AC128DE",  "imgValidCode":"xs3d"  "cloudkeyProjectId": "",  "userType": 1,  "user": {  "organizationId": 1,  "uid": "4000",  "name": "李二",  "phone": "13512345678",  "email": "lier@cnca.net",  "countryName": "CN",  "province": "Guangdong",  "city": "Guangzhou",  "department": "市场部",  "officialResidence": "广东省广州市",  "gender": 1,  "identityType": 2,  "identity": "12345",  "memo": "备注信息"  }  }' |  |

### 13.1.2 验证短信验证码

#### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/register/smsvalidcode/verify/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

#### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| phoneValidCode | String | 手机短信验证码 |
| cloudkeyProjectId | String | 云密钥项目id，必传 |
| userType | Int | 1表示机构员工用户，目前只有1，必传 |
| organizationId | Int | 用户所属机构的id，必传 |
| uid | String | 用户唯一标识，必传 |
| name | String | 用户名，必传 |
| phone | String | 手机号码，必传 |
| email | String | 邮箱，可选 |
| countryName | String | 国家，必传，目前只有CN |
| province | String | 省份的拼音，必传 |
| city | String | 城市的拼音，必传 |
| department | String | 部门，可选 |
| occupation | String | 职业，可选 |
| officialResidence | String | 办公地址，可选 |
| gender | Int | 性别，0男1女，必传 |
| identityType | Int | 证件类型，必传 |
| identity | String | 证件号码，必传 |
| memo | String | 备注，可选 |

#### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为空。

#### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
|  | {  "cloudkeyProjectId": "",  "userType": 1,  "user": {  "organizationId": 1,  "uid": "4000",  "name": "李二",  "phone": "13512345678",  "email": "lier@cnca.net",  "countryName": "CN",  "province": "Guangdong",  "city": "Guangzhou",  "department": "市场部",  "occupation": "医生",  "officialResidence": "广东省广州市",  "gender": 1,  "identityType": 2,  "identity": "12345",  "memo": "备注信息"  }  } | cloudkeyProjectId:云密钥项目id  userType：用户类型，目前只有1，表示员工用户  user节点下的信息：  organizationId:机构id，必传。调用机构信息注册接口的返回值，由调用者自行保存  uid：用户唯一标识，必传  name：用户名，必传  phone：手机号码，必传  email：邮箱，可选  countryName：国家，必传，只能为 CN  province：省份的拼音，必传  city：城市的拼音，必传  department：部门，可选  occupation：职业，可选  officialResidence：办公地址，可选  gender：性别，0男1女，必传  identityType：证件类型，必传  identity：证件号码，必传  memo：备注，可选 |
| 返回 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  } |  |
| curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/api/selfregister/smsvalidcode/verify/2CKV6 -H 'Content-Type:application/json' -d '{  "cloudkeyProjectId": "",  "userType": 1,  "phoneValidCode": "123456",  "user": {  "organizationId": 1,  "uid": "4000",  "name": "李二",  "phone": "13512345678",  "email": "lier@cnca.net",  "countryName": "CN",  "province": "Guangdong",  "city": "Guangzhou",  "department": "市场部",  "officialResidence": "广东省广州市",  "gender": 1,  "identityType": 2,  "identity": "12345",  "memo": "备注信息"  }  }' |  |

## 查询云密钥区域信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/area/search/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| type | int | 查询类型  0查询所有省份及其对应的城市  1查询所有省份  2查询省份下辖的城市，此时province不为空 |
| province | String | 查询该省份下辖所有城市，不支持模糊查询 |

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| name | String | 省份中文名称 |
| spelling | String | 省份中文名称对应的拼音 |
| city节点  name | String | 城市中文名称 |
| city节点  spelling | String | 城市中文名称对应的拼音 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "type": 1,  "province": "Guangdong"  } |  |
| 响应示例 | [{  "name": "广东",  "spelling": "Guangdong",  "citys": [{  "name": "广州",  "spelling": "Guangzhou"  }]  }] |  |
| Curl示例 | curl -kX POST <https://192.168.200.61:443/cloudkeyserver/selfservice/area/search/2CKV6> -H 'Content-Type: application/json'  -d '{  "type": 2,  "province": "广东"  }' | |

## 查询云密钥常量类型

目前只查员工证件类型常量。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/constant/list/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

无需请求参数

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| identityType节点下的  key | Int | 证件类型常量数值 |
| identityType节点  value | String | 证件类型对应的名称 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "identityType": [  {  "key": 1,  "value": "身份证"  },  {  "key": 2,  "value": "军官证"  },  {  "key": 3,  "value": "护照"  },  {  "key": 4,  "value": "工作证"  },  {  "key": 5,  "value": "回乡证"  },  {  "key": 6,  "value": "户口本"  },  {  "key": 7,  "value": "其他个人证件"  },  {  "key": 8,  "value": "香港身份证"  }  ]  }  } |  |
| Curl示例 | curl -kX POST <https://192.168.200.61:443/cloudkeyserver/selfservice/constant/list/2CKV6> -H 'Content-Type: application/json' | |

## 查询云密钥项目信息

目前只查机构员工项目。

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/project/list/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| name | String | 需要搜索的项目名称关键字，为空则查询全部 |
| pageIndex | Int | 分页页码，第几页  和pageSize组合才会做分页查询，否则查询全部 |
| pageSize | Int | 分页显示数量  和pageIndex组合才会做分页查询，否则查询全部 |

### 响应结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| id | String | 云密钥项目id |
| name | String | 云密钥项目名称 |
| projectUid | String | 云密钥项目标识 |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "name": "江门医院",  "pageIndex": 1,  "pageSize": 50  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "total": 1,  "currentRowsSize": 1,  "rows": [{  "name": "江门医院",  "id": "CKP4879674ABDC",  "projectUid": "JiangmenZhongXin"  }]  }  } |  |
| Curl示例 | curl -kX POST <https://192.168.200.61:443/cloudkeyserver/selfservice/project/list/2CKV6> -H 'Content-Type: application/json'  -d '{  "name": "江门医院",  "pageIndex": 1,  "pageSize": 50  }' | |

## 查询云密钥机构信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/org/list/2CKV6 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| projectId | String | 查询项目下的机构信息 |
| name | String | 需要搜索的机构名称关键字，为空则查询全部 |
| pageIndex | Int | 分页页码，第几页  和pageSize组合才会做分页查询，否则查询全部 |
| pageSize | Int | 分页显示数量  和pageIndex组合才会做分页查询，否则查询全部 |

### 响应结构说明

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "projectId": "CKPAB123DBA",  "name": "江门医院",  "pageIndex": 1,  "pageSize": 50  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "total": 1,  "currentRowsSize": 1,  "rows": [{  "name": "江门医院",  "id": 123  }]  }  } |  |
| Curl示例 | curl -kX POST <https://192.168.200.61:443/cloudkeyserver/selfservice/project/list/2CKV6> -H 'Content-Type: application/json'  -d '{  "projectId": "CKPAB123DBA",  "name": "江门医院",  "pageIndex": 1,  "pageSize": 50  }' | |

## 查询云密钥项目的部门信息

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/selfservice/department/list/2CKV9 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| projectId | String | 项目id |

### 响应结构说明

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "projectId": "CKPAB123DBA"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": ["门诊部","信息科"]  } |  |
| Curl示例 | curl -kX POST <https://192.168.200.61:443/cloudkeyserver/selfservice/department/list/2CKV9> -H 'Content-Type: application/json'  -d '{  "projectId": "CKPAB123DBA"  }' | |

# 微信小程序

## 获取用于人脸绑定的bizToken

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/weapp/face/bind/bizToken/2CKV7 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 用户登录令牌 |
| qrCodeIdentity | String | 二维码唯一标识 |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| bizToken | String | 用于启动人机交互核验sdk的bizToken |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "userToken": "BF2D1280F656FEC98910E12D15D5",  "qrCodeIdentity": "656FEC98"  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "bizToken": “**RY98FR0F656FEC98910E12D90RL**”  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/weapp/face/bind/bizToken/2CKV7  -H 'Content-Type:application/json' -d '{  "usertoken": “**BF2D1280F656FEC98910E12D15D5**”  }' | |

## 获取用于人脸登录认证的bizToken

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/weapp/face/login/bizToken/2CKV7 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| qrCodeIdentity | String | 二维码唯一标识 |
| faceType | Int | 人脸类型：0=人脸识别（目前只有0） |
| userType | Int | userType用于区分不同的用户类型。  0表示证书用户本人，为0则authorizedUserId参数可以不传或传空；  1表示被授权人用户，authorizedUserId不能为空。 |
| authorizedUserId | Int | 授权记录ID，见[查询已被授权信息](#_查询已被授权信息)和[被授权用户可用证书查询](#_被授权用户可用证书查询)的响应 |
| userPin | String | 输入的pin，做Base64编码后的值。 |
| cert | Cert | 参考12.1.1[Cert结构](#_响应结构说明),该参数通过证书指定了具体的证书用户，参考[用户证书查询的响应](#_响应结构说明_1) |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| bizToken | String | 用于启动人机交互核验sdk的bizToken |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "qrCodeIdentity": "656FEC98",  "faceType": 0,  "userType": 1,  "authorizedUserId": {{authorizedUserId}},  "userPin": "{{pinBase64}}",  "cert": {  "certContent": "{{certContent}}"  }  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "bizToken": “**RY98FR0F656FEC98910E12D90RL**”  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/weapp/face/login/bizToken/2CKV7  -H 'Content-Type:application/json' -d '{  "faceType": 0,  "userType": 1,  "authorizedUserId":{{authorizedUserId}},  "userPin": "{{pinBase64}}",  "cert": {  "certContent": "{{certContent}}"  }  }' | |

## 获取人脸绑定结果

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/weapp/face/bind/result/2CKV7 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| bizToken | String | 人机交互核验sdk的bizToken |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容为绑定结果提示信息。

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  "bizToken": “**BF2D1280F656FEC98910E12D15D5”**  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": “绑定成功”  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/weapp/face/bind/result/2CKV7  -H 'Content-Type:application/json' -d '{  "bizToken": “**BF2D1280F656FEC98910E12D15D5**”  }' | |

## 获取人脸登录结果

### 请求信息

|  |  |
| --- | --- |
| **请求方式** | https协议 POST |
| **接口URL** | https://ip:port/cloudkeyserver/weapp/face/login/result/2CKV7 |
| **请求数据格式** | application/json |
| **请求消息** | jsonstr |
| **应答消息** | jsonstr |

### 请求结构说明

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| bizToken | String | 人机交互核验sdk的bizToken |

### 响应结构说明

[参考响应结构](#_响应结构)，contents中的内容见下表。

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| userToken | String | 验证成功后台返回userToken |

### 请求响应简单示例

|  |  |  |
| --- | --- | --- |
| **Name** | **value** | **说明** |
| 请求示例 | {  " bizToken": “**BF2D1280F656FEC98910E12D15D5”**  } |  |
| 响应示例 | {  "responseResult": {  "status": 0,  "msg": "成功"  },  "contents": {  "userToken": “**RY98FR0F656FEC98910E12D90RL**”  }  } |  |
| Curl示例 | curl -kX POST https://192.168.200.61:443/cloudkeyserver/weapp/face/login/result/2CKV7  -H 'Content-Type:application/json' -d '{  "bizToken": “**BF2D1280F656FEC98910E12D15D5**”  }' | |

# 接口附加说明

## 请求响应结构说明

### Cert结构

请求参数中，出现频率很高的cert json对象说明。

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名** | **类型** | **约束** | **说明** |
| certContent | String | NOT NULL | Base64编码的证书内容 |

### 响应结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| status | int | 响应编码（0为成功，其他都是失败） |
| msg | String | 响应信息 |
| contents | 结构体 | 响应内容 |

### User结构

办理机构证书时，此对象代表机构信息(org)。

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| organizationId | Int | 用户所属的机构id，调用[机构信息注册](#_1_机构信息注册)获取 |
| uid | String | 用户唯一标识 |
| name | String | 名称 |
| phone | String | 电话 |
| email | String | 邮箱，可选 |
| countryName | String | 国家，只支持CN |
| province | String | 省份，省份拼音 |
| city | String | 城市，城市拼音 |
| department | String | 部门 |
| officialResidence | String | 办公地址，可选 |
| gender | Int | 性别，0男1女 |
| identityType | Int | 证件类型 |
| identity | String | 证件号码 |
| memo | String | 备注，可选 |

### Organization结构

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| name | String | 机构名称 |
| organizationType | Int | 机构类型 |
| identityType | Int | 机构证件类型 |
| identity | String | 证件号码 |
| phone | String | 电话 |
| officialResidence | String | 法定住所 |
| email | String | 邮箱，可选 |
| memo | String | 备注，可选 |
| countryName | String | 国家，只支持CN |
| province | String | 省份，省份拼音 |
| legalPersonName | String | 法人代表名称，可选 |
| legalPersonPhone | String | 法人电话，可选 |
| legalPersonIdentityType | String | 法人证件类型，可选 |
| legalPersonIdentity | String | 法人证件号码，可选 |

### Linkman结构

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| identityType | Long | 个人证件类型, |
| identity | String | 证件号码 |
| phone | String | 经办人电话 |
| email | String | 经办人email |
| address | String | 经办人地址，非必填 |
| name | String | 经办人姓名 |

### DeviceInfo结构

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| deviceModel | String | 设备的型号 |
| sdkVersion | String | 云密钥SDK版本号 |
| osVersion | String | OS版本号 |
| osSDKVersion | String | OS的SDK版本号 |
| platformType | String | 平台类型 |
| imei0 | String | IMEI码 |
| imei1 | String | IMEI码 |
| packageID | String | 安装软件时产生的唯一标识 |
| firstInstallTime | String | 首次安装时间，yyyy-MM-dd HH:mm:ss，24小时制 |
| lastUpdateTime | String | 最近一次更新时间，yyyy-MM-dd HH:mm:ss，24小时制 |

### CertInfo结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| certContent | String | 证书内容的base64编码 |
| certUsage | Int | 证书类型  1表示加密证书  2表示签名证书 |

### Thumbprint结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| algorithm | Int | 摘要算法，参考[Hash算法](#_机构类型) |
| hash | String | 摘要值 |

### Extension结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| oid | String | 扩展oid |
| value | String | Oid对应的扩展值 |

### QRCode结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| type | String | 认证二维码的类型  3表示产生app无状态的二维码  4表示产生微信小程序无状态的二维码 |
| qrCodeBase64 | String | 二维码图片的base64编码 |
| qrCodeContent | String | 表示的所有类JSON文本信息（去掉协议前缀，剩下未JSON的OBJECT），见[二维码内容](#_二维码内容) |

### AuthorizedUser结构



#### AuthorizedUser结构1

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| uid | Int | 被授权人uid |
| name | String | 被授权人名字 |
| phone | String | 被授权人联系电话 |
| memo | String | 备注 |
| validStart | String | 有效授权开始时间 |
| validEnd | String | 有效授权结束时间 |
| validCount | Int | 有效授权次数 |

#### AuthorizedUser结构2

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| id | Int | 授权用户id, 通过接口[用户分页查询授权记录](#_用户分页查询授权记录)可以获取 |
| validStart | String | 授权有效期开始时间  参考[时间格式](#_时间格式说明) |
| validEnd | String | 授权有效期结束时间  参考[时间格式](#_时间格式说明) |
| validCount | Int | 授权使用次数 |

#### AuthorizedUser结构3

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| id | Int | 被授权用户的id |
| uid | String | 被授权用户的唯一标识 |
| name | String | 被授权用户的名字 |
| memo | String | 被授权用户的备注信息 |
| applicationTime | String | 授权录入时间，参考[时间格式](#_时间格式说明) |
| validStart | String | 授权开始时间，参考[时间格式](#_时间格式说明) |
| validEnd | String | 授权结束时间，参考[时间格式](#_时间格式说明) |
| allowCallCount | Int | 授权使用次数，-1表示没有限制 |

#### AuthorizedUser结构4

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| id | Int | 被授权用户的id |
| uid | String | 被授权用户的唯一标识 |
| name | String | 被授权用户的名字 |
| memo | String | 被授权用户的备注信息 |
| applicationTime | String | 授权录入时间，参考[时间格式](#_时间格式说明) |
| confirmTime | String | 确认授权时间，参考[时间格式](#_时间格式说明) |
| validStart | String | 授权开始时间，参考[时间格式](#_时间格式说明) |
| validEnd | String | 授权结束时间，参考[时间格式](#_时间格式说明) |
| allowCallCount | Int | 授权使用次数。-1表示没有限制 |

#### AuthorizedUser结构5

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| id | Int | 被授权用户的id |
| uid | String | 被授权用户的唯一标识 |
| name | String | 被授权用户的名字 |
| memo | String | 被授权用户的备注信息 |
| pin | String | 证书初始使用pin码 |
| applicationTime | String | 授权录入时间，参考[时间格式](#_时间格式说明) |
| confirmTime | String | 确认授权时间，参考[时间格式](#_时间格式说明) |
| validStart | String | 授权开始时间，参考[时间格式](#_时间格式说明) |
| validEnd | String | 授权结束时间，参考[时间格式](#_时间格式说明) |
| allowCallCount | Int | 授权使用次数，-1表示没有限制 |

#### AuthorizedUser结构6

|  |  |  |
| --- | --- | --- |
| **参数** | **类型(字符长度)** | **参数说明** |
| id | Int | 被授权用户的id |
| uid | String | 被授权用户的唯一标识 |
| name | String | 被授权用户的名字 |

### Issuer结构

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| issuer | String | 颁发者 |
| displayName | String | 颁发者的显示名字 |
| hexEncode | String | 颁发者DER编码后再Hex编码 |
| c | String[] | 在issuer中表示颁发者的c项 |
| o | String[] | 在issuer中表示颁发者的o项 |
| ou | String[] | 在issuer中表示颁发者的ou项 |
| cn | String[] | 在issuer中表示颁发者的cn项 |
| email | String[] | 在issuer中表示颁发者的email项 |
| st | String[] | 在issuer中表示颁发者的st项 |
| l | String[] | 在issuer中表示颁发者的l项 |
| dc | String[] | 在issuer中表示颁发者的dc项 |

### Subject结构

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| Subject | String | 主体 |
| displayName | String | 主体的显示名字 |
| hexEncode | String | 主体DER编码后再Hex编码 |
| c | String[] | 在subject中表示颁发者的c项 |
| o | String[] | 在subject中表示颁发者的o项 |
| ou | String[] | 在subject中表示颁发者的ou项 |
| cn | String[] | 在subject中表示颁发者的cn项 |
| email | String[] | 在subject中表示颁发者的email项 |
| st | String[] | 在subject中表示颁发者的st项 |
| l | String[] | 在subject中表示颁发者的l项 |
| dc | String[] | 在subject中表示颁发者的dc项 |

### 15.1.14. Row结构

#### 15.1.14.1 Row结构1

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| id | Int | 授权记录的id标识（全局唯一） |
| name | String | 该授权记录中被授权人的名字 |

#### 16.1.14.2 Row结构2

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| id | Int | 授权记录的id标识（全局唯一） |
| name | String | 被授权人的名字 |
| certId | String | 证书id |
| phone | String | 电话号码 |
| authStatus | Int | 被授权人当前的状态  0=已录入，尚未被授权 1=已被授权 2=冻结授权  3=授权过期  4=删除授权（不会返回客户端） |
| allowAuthorized | Boolean | 是否能被授权 |
| allowReAuthorized | Boolean | 是否能被重置授权（解锁） |
| allowUnAuthorized | Boolean | 是否能被冻结授权 |
| allowActiveAuthorized | Boolean | 是否能被激活授权（解冻） |
| allowRenewAuthorized | Boolean | 是否能被重新授权 |
| applicationTime | String | 管理员录入时间 |
| validStart | String | 有效授权开始时间 |
| validEnd | String | 有效授权结束时间 |
| allowCallCount | Int | 允许调用次数 |
| memo | String | 备注信息 |

#### 16.1.14.3 Row结构3

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| loginUuid | String | 登录UUID，可作为退出登录的参数 |
| loginTime | String | 登录时间，，参考[时间格式](#_时间格式说明) |
| current | Boolean | 当前信息是否为usertoken对应的登录信息  true为是  false为否 |
| applicationName | String | 应用名称 |

### CertInfo结构

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| certContent | String | 证书内容的base64编码 |
| certUsage | Int | 证书类型，1表示加密证书，2表示签名证书 |

### Request结构

|  |  |  |
| --- | --- | --- |
| 参数 | 类型(字符长度) | 参数说明 |
| requestId | String | 云密钥业务单号 [request](#_12.1.16._CertInfo结构) |

### SealSize结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **width** | Int | 签章的宽度 |
| **height** | Int | 签章的高度 |

### Position结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| **page** | Int | 签章所在的PDF页数，从1开始计数  -1表示最后一页 |
| **x** | Int | 签章的x坐标 |
| **y** | Int | 签章的y坐标 |

### SignFieldInfo结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signFieldName | String | 签名域名称 |
| validity | Validity | 验证信息，参见12.1.20[Validity](#_12.1.20._Validity结构)结构 |
| signInfo | SignInfo | 签名信息，参见12.1.22[SignInfo](#_12.1.22._SignInfo结构)结构 |
| certContent | String | 签名证书内容 |

### Validity结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| state | Int | 该签名域对应的验证状态，0为验证成功，1为验证失败 |
| faultReason | FaultReason | state为1时存在，表示验证失败的原因，见12.1.21 [FaultReason](#_12.1.21._FaultReason结构)结构 |

### FaultReason结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| code | Int | 错误代号 |
| desc | String | 错误描述 |

### SignInfo结构

|  |  |  |
| --- | --- | --- |
| **参数名** | **类型** | **说明** |
| signer | String | 签名者名称 |
| signingTime | String | 签名时间 |
| tsaTime | String | 可信时间戳时间 |
| userCert | String | 用户证书base64编码 |
| tsaCert | String | 时间戳证书base64编码 |

## 接口参数常量说明

### Hash算法

|  |  |  |
| --- | --- | --- |
| Hash算法 | 值 | 说明 |
| MD5 | 4096 | MD5算法 |
| SHA1 | 8192 | SHA1算法 |
| SHA224 | 12288 | SHA224算法 |
| SHA256 | 16384 | SHA256算法 |
| SHA384 | 20480 | SHA384算法 |
| SHA512 | 24576 | SHA512算法 |
| SM3 | 28672 | SM3算法 |
| SHA512\_224 | 32768 | SHA512/224算法 |
| SHA512\_256 | 36864 | SHA512/256算法 |
| SHA3\_224 | 40960 | SHA3-224算法 |
| SHA3\_256 | 45056 | SHA3-256算法 |
| SHA3\_384 | 49152 | SHA3-384算法 |
| SHA3\_512 | 53248 | SHA3-512算法 |

### 签名算法

|  |  |  |
| --- | --- | --- |
| 签名算法 | 值 | 说明 |
| MD5WITHRSA | 1 | MD5WithRSA签名算法 |
| SHA1WITHRSA | 2 | SHA1WithRSA签名算法 |
| SHA224WITHRSA | 3 | SHA224WithRSA签名算法 |
| SHA256WITHRSA | 4 | SHA256WithRSA签名算法 |
| SHA384WITHRSA | 5 | SHA384WithRSA签名算法 |
| SHA512WITHRSA | 6 | SHA512WithRSA签名算法 |
| MD5WITHRSA\_NOHASHID | 7 | 只对Hash值填充的MD5WithRSA签名算法 |
| SHA1WITHRSA\_NOHASHID | 8 | 只对Hash值填充的SHA1WithRSA签名算法 |
| SHA224WITHRSA\_NOHASHID | 9 | 只对Hash值填充的SHA224WithRSA签名算法 |
| SHA256WITHRSA\_NOHASHID | 10 | 只对Hash值填充的SHA256WithRSA签名算法 |
| SHA384WITHRSA\_NOHASHID | 11 | 只对Hash值填充的SHA384WithRSA签名算法 |
| SHA512WITHRSA\_NOHASHID | 12 | 只对Hash值填充的SHA512WithRSA签名算法 |
| DSAWITHSHA1 | 13 | DSAWithSHA1签名算法 |
| DSAWITHSHA224 | 14 | DSAWithSHA224签名算法 |
| DSAWITHSHA256 | 15 | DSAWithSHA256签名算法 |
| ECDSAWITHSHA1 | 16 | ECDSAWithSHA1签名算法 |
| ECDSAWITHSHA224 | 17 | ECDSAWithSHA224签名算法 |
| ECDSAWITHSHA256 | 18 | ECDSAWithSHA256签名算法 |
| ECDSAWITHSHA384 | 19 | ECDSAWithSHA384签名算法 |
| ECDSAWITHSHA512 | 20 | ECDSAWithSHA512签名算法 |
| SSL\_RSA | 21 | SSL3客户端认证使用的RSA签名算法 |
| RSA\_PSS | 22 | PKCS#1 PSS的RSA签名算法 |
| ECDSAWITHSM3 | 23 | ECDSAWithSM3签名算法 |
| SM3WITHRSA\_NOHASHID | 24 | 只对Hash值填充的SM3WithRSA签名算法 |
| SM3WITHSM2 | 25 | 使用SM3的SM2签名算法 |
| SHA1WITHSM2 | 26 | 使用SHA1的SM2签名算法 |
| SHA224WITHSM2 | 27 | 使用SHA224的SM2签名算法 |
| SHA256WITHSM2 | 28 | 使用SHA256的SM2签名算法 |
| SHA384WITHSM2 | 29 | 使用SHA384的SM2签名算法 |
| SHA512WITHSM2 | 30 | 使用SHA512的SM2签名算法 |
| SM3WITHRSA | 31 | SM3WithRSA签名算法 |
| RSASSA\_PKCS1\_V1\_5 | 32 | 输入Hash值的PKCS#1 V1.5签名，仅用于NetcaPKICryptoSignHash和NetcaPKICryptoVerifyHash |
| SJS\_SM3WITHSM2 | 33 | 非标准的使用SM3的SM2签名算法，不使用Z值，直接对要签名的数据进行Hash。 |
| SHA512\_224WITHRSA | 34 | SHA512\_224WithRSA签名算法 |
| SHA512\_256WITHRSA | 35 | SHA512\_256WithRSA签名算法 |
| SHA3\_224WITHRSA | 36 | SHA3-224WthRSA签名算法 |
| SHA3\_256WITHRSA | 37 | SHA3-256WthRSA签名算法 |
| SHA3\_384WITHRSA | 38 | SHA3-384WthRSA签名算法 |
| SHA3\_512WITHRSA | 39 | SHA3-512WthRSA签名算法 |
| ECDSAWITHSHA3\_224 | 40 | ECDSAWithSHA3-224 |
| ECDSAWITHSHA3\_256 | 41 | ECDSAWithSHA3-256 |
| ECDSAWITHSHA3\_384 | 42 | ECDSAWithSHA3-384 |
| ECDSAWITHSHA3\_512 | 43 | ECDSAWithSHA3-512 |
| ECDSA\_SIGNHASH | 44 | ECDSA，仅用于NetcaPKICryptoSignHash和NetcaPKICryptoVerifyHash |
| DSA\_SIGNHASH | 45 | DSA，仅用于NetcaPKICryptoSignHash和NetcaPKICryptoVerifyHash |
| DSAWITHSHA384 | 46 | DSAWithSHA384 |
| DSAWITHSHA512 | 47 | DSAWithSHA512 |
| DSAWITHSHA3\_224 | 48 | DSAWithSHA3\_224 |
| DSAWITHSHA3\_256 | 49 | DSAWithSHA3\_256 |
| DSAWITHSHA3\_384 | 50 | DSAWithSHA3\_384 |
| DSAWITHSHA3\_512 | 51 | DSAWithSHA3\_512 |
| WAPI\_ECDSA192WITHSHA256 | 52 | WAPI的ECDSAWithSHA256 |

### 机构证件类型

|  |  |
| --- | --- |
| **常量值** | **描述** |
| 501 | 组织机构代码证 |
| 502 | 营业执照 |
| 503 | 事业单位登记证 |
| 504 | 税务登记证 |
| 505 | 社会团体登记证 |
| 506 | 人民团体登记证 |
| 507 | 企业法人营业执照 |
| 508 | 事业单位法人登记证 |
| 509 | 社会团体法人登记证 |
| 510 | 人民团体法人登记证 |
| 511 | 社会保险登记证 |
| 512 | 其他机构证件 |
| 513 | 统一社会信用代码 |

### 个人证件类型

|  |  |
| --- | --- |
| **常量值** | **描述** |
| 1 | 身份证 |
| 2 | 军官证 |
| 3 | 护照 |
| 4 | 工作证 |
| 5 | 回乡证 |
| 6 | 户口本 |
| 7 | 其他个人证件 |
| 8 | 香港身份证 |

### 机构类型

|  |  |
| --- | --- |
| **常量值** | **描述** |
| 1 | 企业 |
| 2 | 事业 |
| 3 | 国家机关 |
| 4 | 社会团体 |
| 5 | 人民团体 |
| 6 | 其他 |

### 国家、省份、城市常量取值

说明：

国家字段，取值为：CN ，表示中国。暂不支持外国取值。

省份和城市的取值都使用拼音字符串，如：广东对应取值为：Guangdong ；此时城市的取值为广东下面划分的城市，如广州取值为：Guangzhou 。即，省份和城市是二级联动的关系。

其联动的对应关系见如下两个excel表。





### 业务状态常量

|  |  |
| --- | --- |
| **常量值** | **描述** |
| 0 | 成功 |
| 其他 | 失败 |

### 性别

|  |  |
| --- | --- |
| **常量值** | **描述** |
| 0 | 男 |
| 1 | 女 |

### 响应状态码常量status值

对服务器返回的status值，以及对应的msg简单说明

|  |  |  |
| --- | --- | --- |
| **常量值（status）** | **提示信息（msg）** | **返回该status的可能原因** |
| 0 | 成功 | 操作成功 |
| -1 | 系统出错 | 云密钥系统错误 |
| -2 | 身份未认证或令牌userToken已失效 | userToken失效或错误 |
| -3 | 口令（PIN）错误 | 用户口令错误，解不开用户私钥 |
| -4 | 账号被锁住，请联系管理人解锁 | 口令错误次数超过规定值，被锁 |
| -5 | 口令（PIN）错误太频繁，被限制验证 | 口令错误太频繁被限制 |
| -6 | 短信验证码错误 | 输入短信验证码有误 |
| -7 | 验证信息未生成或已失效 | userToken过期或者二维码过期 |
| -8 | 下发短信失败 | 网络异常或者短信服务器异常 |
| -9 | 证书未生效或已过期 | 证书不在有效期内 |
| -10 | 授权用户非法操作 | 授权用户修改非本人证书的签章图片 |
| -20 | 签名验证失败 | 签名验证不通过或验签参数有误 |
| … | … | … |
| 100 | 入参错误 | 特殊参数校验不满足 |

### 证书的版本号

|  |  |
| --- | --- |
| 值 | 说明 |
| 0 | 版本V1 |
| 1 | 版本V2 |
| 2 | 版本V3 |

### 公钥算法

|  |  |
| --- | --- |
| 值 | 说明 |
| 1 | RSA算法 |
| 2 | DSA算法 |
| 3 | DH算法 |
| 4 | ECC算法 |

### 密钥用法

|  |  |
| --- | --- |
| 常量 | 说明 |
| 1 | 数字签名 |
| 2 | 不可否认 |
| 4 | 加密密钥 |
| 8 | 加密数据 |
| 16 | 密钥协商 |
| 32 | 签证书 |
| 64 | 签CRL |
| 128 | 密钥协商时，只加密 |
| 256 | 密钥协商时，只解密 |

### ECC曲线的类型

|  |  |
| --- | --- |
| 值 | 说明 |
| 1 | NIST的P192 |
| 2 | NIST的P224 |
| 3 | NIST的P256 |
| 4 | NIST的P384 |
| 5 | NIST的P512 |
| 6 | WAPI指定使用的曲线 |
| 7 | SM2推荐曲线 |
| 8 | brainpoolP192r1 |
| 9 | brainpoolP224r1 |
| 10 | brainpoolP256r1 |
| 11 | brainpoolP320r1 |
| 12 | brainpoolP384r1 |
| 13 | brainpoolP512r1 |
| 14 | ANSSI ECC FRP256V1 |
| 16 | secp256k1 |

### 人脸识别类型常量（faceTypeBitValue）说明

使用位掩码，用一个二进制数来表示身份验证类型；

二进制的每一位来表示一种身份验证类型，该位上的值：0表示不支持该验证方式，1表示支持该类型。

目前有以下几种验证类型：

用二进制第一位值为1表示支持网证通人脸识别，即1(1)

用二进制第二位值为1表示支持微信小程序人脸识别，即10(2)

|  |  |  |
| --- | --- | --- |
|  | 二进制值 | 十进制值 |
| 网证通人脸识别 | 0001 | 1 |
| 微信小程序人脸识别 | 0010 | 2 |

说明：要支持多种验证类型，请使用‘或’运算。

例如：支持（网证通人脸识别+微信小程序人脸识别）

获取faceTypeBitValue值的伪代码：int faceTypeBitValue = 0001 | 0010  
  
验证一个faceTypeBitValue值是否支持某种验证方式，使用‘与’运算

例如：验证faceTypeBitValue 值是否支持网证通人脸识别

伪代码：

FACE\_TYPE = 1

boolean flag = (FACE\_TYPE & faceTypeBitValue) != 0 // 为true说明支持网证通人脸识别

## 安全性附加说明

### 通信的安全

为了通信双方的身份认证及数据加密，接口的通信层采用单向证书认证的HTTPS协议。通信的数字证书由NETCA签发。

### 时间格式说明

yyyy-MM-dd HH:mm:ss格式，北京时间（+8:00）。内网环境尽量使用NTP时间同步

### 管理员签名说明

签名相关参数设置如下：

|  |  |
| --- | --- |
| 签名算法 | SHA256WithRSA签名算法 |
| 字符集 | UTF-8 |
| 签名格式 | PKCS7 |
| 是否带原文 | 带原文 |
| 包含证书的选项 | 仅包含签名的证书即可，无需包含整个证书链 |
| 输出结果 | 签名结果用base64编码字符串，不带回车换行 |