云　南　省　生　态　环　境　厅

〔2025〕—911

云南省生态环境厅关于加快推进云南省重点行业企业门禁视频监控系统建设及联网的通知

各州、市生态环境局：

为贯彻执行《重点行业移动源监管与核查技术指南》（HJ 1321—2023）、《云南省柴油货车污染治理攻坚战实施方案》、云南省生态环境厅印发的《2025年大气污染防治工作要点》等有关管理要求，进一步加强云南省移动源环境监管，推动重点行业企业门禁及视频监控系统规范化建设和联网接入。现将有关事项通知如下：

一、涉及火电、钢铁、煤炭、焦化、有色（铝工业）等重点行业的州（市）要督促所辖相关企业严格按照《重点行业移动源监管与核查技术指南》（HJ 1321—2023）及《企业门禁及视频监控系统联网方案》（附件1）进行全面自查与评估，根据企业实际选择与省级联网对接方式，确保系统硬件、软件及数据格式均符合国家规范要求。

二、相关州（市）要督促所辖企业按照《云南省移动源环境综合管理平台-重点行业移动源监管门禁及视频监控信息采集传

输接口规范》（附件2）对现有门禁及视频监控系统进行认真比对，对不符合接口规范的系统，限期进行升级、改造。企业要按照规定的数据传输协议，通过专用网络通道，将门禁系统数据实时、稳定地传输至省级平台。确保数据的完整性、准确性、连续性，同时企业应向云南省生态环境厅提供视频监控摄像头端口、用户名、密码等相关信息供省级平台远程调用。

三、相关州（市）要加强调度，做好组织协调工作，确保所辖火电、钢铁、有色（铝工业）相关企业于2026年1月20日前，煤炭、焦化相关企业于2026年3月31日前务必全面完成系统建设并与省级平台联网。

附件：1．企业门禁及视频监控系统联网方案

2．云南省移动源环境综合管理平台-重点行业移动源监管门禁及视频监控信息采集传输接口规范

云南省生态环境厅

2025年12月25日

（联系人及电话：大气处李明保　0871-64107313

信息中心李媛　0871-64197139）

抄送：省信息中心。

附件 1

企业门禁及视频监控系统联网方案

为贯彻落实省级生态环境主管部门对重点行业移动源监管工作的统一部署，依据《重点行业移动源监管与核查技术指南》（HJ 1321—2023）等相关技术规范，为实现对企业门禁、视频监控等数据的有效采集与集中管理，现就企业端系统与省级平台的联网对接工作，制定本实施方案。请各企业高度重视，参照本方案要求，结合自身实际情况，选择适宜的联网方式并尽快组织实施。

一、门禁数据对接要求

企业门禁数据传输需按照省级平台提供的《云南省移动源环境综合管理平台-重点行业移动源监管门禁及视频监控信息采集传输接口规范》（可参考附件 2），通过专用网络将门禁数据传输至省级平台，保证数据上传的稳定性、可靠性、安全性。

二、视频监控联网对接方式

企业可根据自身现有视频监控系统现状及技术能力，从以下两种对接方式中任选其一，完成与省级平台的联网工作。同时，所采用的门禁及视频监控系统应满足以下要求：具备对进出厂车辆出入口的 24 小时实时监控功能，并完整覆盖车辆进出全过程；视频画面上须显著标注进出厂时间、出入口编号及道闸编号等信息；系统须配备本地存储设备，支持按日期将视频保存在本地硬

盘，并具备手动起杆自动标注与事件视频单独保存或调取的功能，所有历史视频本地保存周期不少于12个月。此外，企业应向生态环境主管部门提供视频监控摄像头的访问端口、用户名、密码等信息，以供远程调用。

（一）方式一：通过第三方视频汇聚平台与省级平台对接

企业可通过两种主要途径实现与省级平台的门禁视频监控对接。其一，借助主流视频厂商云平台（如海康威视、大华股份等），利用其公有云服务作为技术载体，通过专用网络将企业本地视频资源汇聚至厂商云平台，省级平台再通过标准API接口（如GB/T28181协议或厂商开放接口）在云端完成视频调阅，此时省级平台可采用两种方式获取视频：向厂商云平台发起请求并由云平台推送视频流，或通过专用网络直接访问企业上传的摄像头信息（含IP地址、端口、用户名及密码等）。其二，通过第三方运营商视频汇聚平台（如电信海康视频平台、移动千里眼平台、联通云智眼平台等）进行中转，由第三方平台汇聚企业本地视频监控资源并通过专网网络对接省级平台。对于已接入第三方视频汇聚平台的重点行业企业，可在现有联网架构基础上直接增加门禁视频设备，复用既有网络，将门禁视频信号一并接入第三方平台，无需单独建立对接网络通道，由企业负责相关网络部署与调试，确保视频稳定、清晰且无延迟地接入第三方平台。

（二）方式二：通过视频直联省级平台对接

企业门禁视频监控未对接任何第三方视频汇聚平台，可自行建设并管理视频监控平台，并通过专用网络连接省级平台。企业

网络出口需具备固定公网IP地址，视频设备应支持RTSP等主流流媒体协议，与省级平台联网前应开展兼容性测试，需保障省级平台可通过专用网络稳定调阅企业视频。

三、安全要求

企业须严格贯彻落实《重点行业移动源监管与核查技术指南》（HJ 1321—2023）中的安全规定，确保门禁及视频监控系统满足以下安全要求：

（一）**防火墙**：门禁及视频监控系统应建立企业级防火墙，标准配置千兆网络接口，确保数据和视频正常稳定上传，具备入侵防御及防病毒功能，同时支持入侵防御特征库和防病毒库的定期自动更新。

（二）**系统安全**：为保证车辆出入信息访问安全和数据安全，系统应采取必要的安全防护措施，有条件的企业可进行国家信息系统安全等级保护备案。

附件 2

云南省移动源环境综合管理平台-重点行业移动源监管门禁及视频监控信息采集传输接口规范

2025 年 11 月修订

目录

1 制定目的

为贯彻《中华人民共和国环境保护法》、《中华人民共和国大气污染防治法》，积极应对重污染天气，改善环境空气质量，促进精准、科学、依法治污，规范全省重点行业企业和重点用车单位门禁及视频监控系统与省生态环境厅监管系统联网，实现数据共享，特制定本规范。

2 规范性引用文件

GB 17691-2005 车用压燃式、气体燃料点燃式发动机与汽车排气污染物排放限值及测量方法（中国Ⅲ、Ⅳ、Ⅴ阶段）

GB 17691-2018 重型柴油车污染物排放限值及测量方法（中国第六阶段）

GB 20891-2007 非道路移动机械用柴油机排气污染物排放限值及测量方法（中国Ⅰ、Ⅱ阶段）

GB 20891-2014 非道路移动机械用柴油机排气污染物排放限值及测量方法（中国第三、四阶段）

HJ 460 环境信息网络建设规范

HJ 608 排污单位编码规则

GA/T 16　道路交通管理信息代码

HJ 1321-2023 重点行业移动源监管与核查技术指南

环办大气函〔2020〕340号　重污染天气重点行业应急减排措施制定技术指南（2020年修订版）

3 适用范围

本标准适用于云南省重点行业企业端系统与省级平台通过接口方式进行数据对接。

4 数据交换方式

4.1 数据交换清单

接口地址	请求方式	参数类型	说明
http://IP:Port/access/api/dataReceive/init-auth	POST	JSON	企业端基础信息注册接口
http://IP:Port/access/api/dataReceive/getToken	POST	JSON	企业获取访问令牌接口
http://IP:Port/access/api/dataReceive/enterprise	POST	JSON	企业上传企业基本信息接口
http://IP:Port/access/api/dataReceive/cameraInfo	POST	JSON	企业上传摄像头信息接口
http://IP:Port/access/api/dataReceive/heartBeat	POST	JSON	企业上传企业联网心跳信息接口
http://IP:Port/access/api/dataReceive/exceedVehicle	POST	JSON	企业获取平台超标车辆接口
http://IP:Port/access/api/dataReceive/basStrategy	POST	JSON	企业获取平台管控策略接口
http://IP:Port/access/api/dataReceive/strategyConfirm	POST	JSON	企业获取管控策略之后通知平台获取成功接口
http://IP:Port/access/api/dataReceive/alarmVehicle	POST	JSON	企业获取平台违规通行车辆接口

接口地址	请求方式	参数类型	说明
http://IP:Port/access/api/dataReceive/alarmConfirm	POST	JSON	企业获取平台违规通行车辆接口之后通知平台获取成功接口
http://IP:Port/access/api/dataReceive/vehicleInfo	POST	JSON	企业上传运输及厂内车辆信息接口
http://IP:Port/access/api/dataReceive/nonMachinery	POST	JSON	企业上传非道机械台账信息接口
http://IP:Port/access/api/dataReceive/vehicleAccess	POST	JSON	企业上传车辆进出厂记录接口
http://IP:Port/access/api/dataReceive/exceedVehicleConfirm	POST	JSON	企业上传超标车辆核实信息接口

4.2 接口调用方式

采用 http 访问方式，监管部门提供数据服务接口，由企业端通过对应接口进行数据读写。

身份认证：接口采用 HTTP Header 身份认证，所有请求均须在 Header 中携带凭证，以下为请求头字段：

字段名	必填	类型	说明	示例值
Authorization	是	字符	访问令牌	Bearer eyJhbGciOiJIUzI1Niasdad12312312
clientId	是	字符	企业端标识(apiKey)，由监管平台统一分配	YN_Mn2dE7xQa9Bc1fRt4sLpVxYz

请求方式：POST

数据格式：application/json

字符编码：UTF-8

数据加密：传输 JSON 数据使用 RSA 与 AES 进行混合数据加密

注意：接口中 6.1 企业端注册接口无需设置请求头。6.2 获取访问令牌接口需要设置请求头 clientId（接口 key），其余接口均需要设置请求头 Authorization 与 clientId 字段。

4.3 接口调用说明

上传数据请求体格式要求为 JSON，数据结构体为 JSON 类型的数据 signature、encryptedAesKey、encryptedContent 组成。

signature：AES 加密后的数据生成的数字签名，用于监管端校验数据完整性与来源合法性。

encryptedAesKey：经平台 RSA 公钥加密后的 AES 会话密钥。

encryptedContent：经 AES 加密后的实际业务数据（Base64 编码）。

请求体 JSON 实例：

➢ 通用请求体：（所有参数均用此结构进行传输）

```
{
  "signature":"*****",
      "encryptedAesKey": "****",
      "encryptedContent": "*****"
}
```

返回体格式为 JSON，由上传状态 code，提示信息 message，企业端 RSA 公钥加密后的 AES 会话密钥 aesKey，返回数据 data 组成。

其中返回数据 data 为空时，密钥 aesKey 也为空，无需解密。

其中返回数据 data 不为空时，数据为 AES 加密后的 Base64 编码。根据会话密钥 aesKey 解密。

➢ 上传成功返回示例：

```
{
    "code": 200,
     "message": "操作成功",
     "aesKey": null,
    "data": null
```

```
}
```

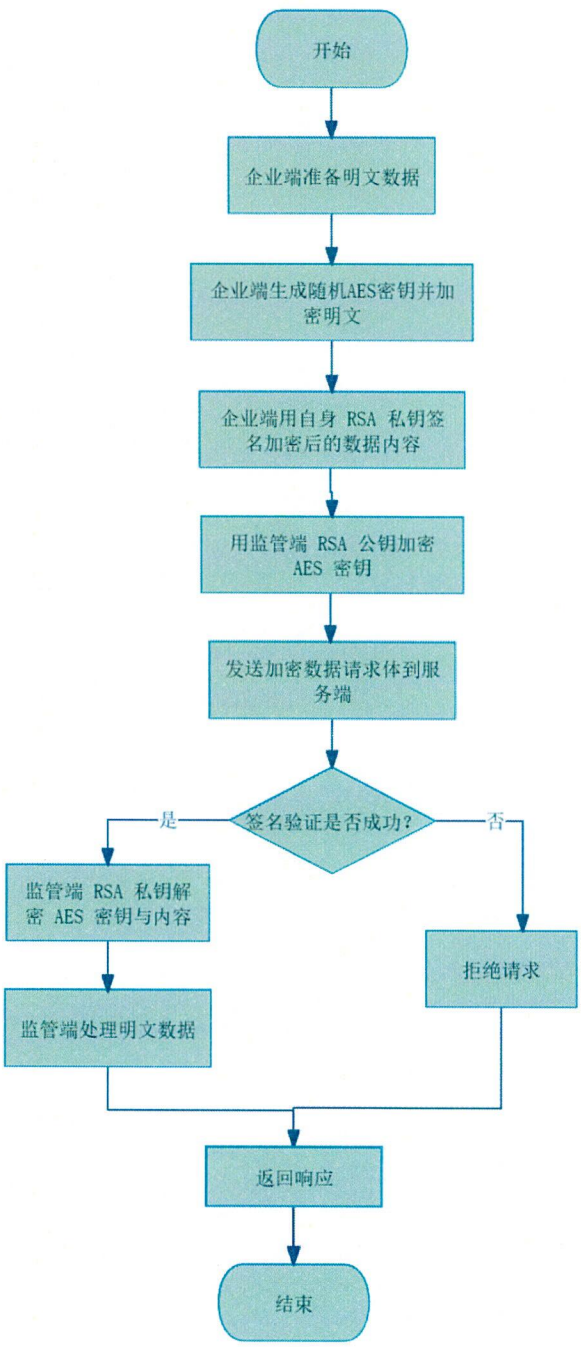
- 上传失败返回示例

```
{
    "code": 500,
      "message": "❌ 验签失败！签名不匹配！数据可能被篡改或来源不可信。",
      "aesKey": null,
    "data": null
}
```

5 安全传输机制

5.1 数据流转加密流程

为保障业务数据在传输过程中的机密性与完整性，系统采用混合加密机制 + 数字签名的双重安全策略。整体数据流转流程如下：



1. 企业端生成自己的 RSA 密钥对（RSA 密钥对生成认证成功了最好就写在配置文件里面了，不要随时去重新调方法重新生成），提供企业信息及 RSA 公钥信息给到监管端，RSA 私钥请自行保存。监管端会根据企业信息提供 apiKey 及监管端 RSA 公钥等内容。（图示片段仅供参考）

企业端密钥生成

```
// 企业端（上传端）生成自己的密钥对（用于签名）
KeyPair clientKeyPair = generateKeyPair(2048);
PublicKey clientPublicKey = clientKeyPair.getPublic();
PrivateKey clientPrivateKey = clientKeyPair.getPrivate();
// 获取 Base64 编码的企业端公钥（供监管端配置使用）
// 企业端公钥 (Base64):
String clientPublicKeyBase64 = getPublicKeyString(clientPublicKey);
// 企业端私钥 (Base64):
String clienPrivateKeyBase64 = getPrivateKeyString(clientPrivateKey);

/**
 * 生成RSA密钥对
 *
 * @param keySize 密钥长度，建议2048
 * @return 密钥对
 */
public static KeyPair generateKeyPair(int keySize) throws Exception {
    KeyPairGenerator keyPairGenerator = KeyPairGenerator.getInstance(KEY_ALGORITHM);
    keyPairGenerator.initialize(keySize);
    return keyPairGenerator.generateKeyPair();
}
```

2. 企业端生成原始业务数据（如 JSON 报文、图片二进制流、长文本等），具体内容参考数据结构说明。

3. 企业端随机生成一个临时 AES-256 会话密钥，用于本次请求的数据加密。（图示片段仅供参考）

企业端密钥生成

```
// 1. 生成 AES 密钥
KeyGenerator keyGen = KeyGenerator.getInstance(ALGORITHM);
// 密钥长度 (256 bits)
keyGen.init(KEY_SIZE);
SecretKey aesKey = keyGen.generateKey();
```

4. 使用该 AES 密钥对原始数据进行对称加密，生成 encryptedContent。（图示片段仅供参考）

企业端AES加密

```
// 2. AES 加密内容
Cipher aesCipher = Cipher.getInstance(AES_TRANSFORMATION);
aesCipher.init(Cipher.ENCRYPT_MODE, aesKey);
byte[] encryptedContent = aesCipher.doFinal(content.getBytes(CHARSET));
```

5. 使用监管端预置下发的 RSA 公钥 对该会话 AES 密钥进行非对称加密，生成 encryptedAesKey。（图示片段仅供参考）

企业端加密

```
// 3. RSA 加密 AES 密钥
Cipher rsaCipher = Cipher.getInstance(RSA_TRANSFORMATION);
rsaCipher.init(Cipher.ENCRYPT_MODE, publicKey);
byte[] encryptedAesKey = rsaCipher.doFinal(aesKey.getEncoded());
```

6. 企业端对 encryptedAesKey 和 encryptedContent 拼接后的数据进行 RSA 数字签名，生成 signature。（图示片段仅供参考）

企业端数据签名

```
// 2. 使用企业端私钥对加密数据结构进行签名
/**
 * 私钥签名 - 对混合加密数据结构
 * CLIENT_PRIVATE_KEY_BASE64为企业端私钥
 */
PrivateKey priKey = getPrivateKeyFromBase64(CLIENT_PRIVATE_KEY_BASE64);
byte[] input = concatenate(encryptedAesKey, encryptedContent);
Signature signature = Signature.getInstance(SIGNATURE_ALGORITHM);
signature.initSign(priKey);
signature.update(input);
String signature数字签名 = Base64.getEncoder().encodeToString(signature.sign());
```

7. 企业端将 encryptedAesKey、encryptedContent、数字签名 signature 一并提交至监管端 API 接口。


```
{
   "signature":"X4Ni1GCfn7dR4Q0S4zQmCO7tU34Fab2jwErMGifXovlGc44EKLFtvO
WJBJ13vMkzPp514/MosjCG+E+q5dAjR9xVeM21WZfMLSadp3Bz57yzrWgAdOznHST
5Rh+uDoRapKtLJioKZrA5L208NRE2J9PAMyUJQnBpv/N0kP+zR09mw9MEf1Tn1AP
DSJiMXso/vBiZRGSgH2H+lMhHZqniHKmUiyqUunZpHfG904BzYStaoEqZia61ikElZO
b/WIytRkxKILVdo2TaxS/OIOP6N0gPae7+M9UfVmkSTBaAb9PjxvWbkVs+mMsX5XzT
equSlQTwAygrMvCHCi/7VTfPOw==",
      "encryptedAesKey":
"iFv5p8eTIL9qWQBQqZI9dDvI8Sj5U7ZKCbZyL6DEAHjf98mQKAI+cpJV1zuSI6MDI
TKIScTiRay8m18vEhs8Uu1wpUeXt74lwnfAXB6eKx0ldr/r4SELsj6dm5A2Mrue0af5QG
AlFoOl0hsTTP6Au6dKxwLm2RO1PUG3GbRuFcMSMMF90hi+z5O9t/xi2WtORrsQOx
Nh60A8ao73V+oExr4/Q/NklA6tx2R3df4l4tOvvX0+j+KlRJ/XvCY1BrQ9u+inB3x/EdRt2
XSnH4n9RE/RBn865lKS3BPiurxMrm1ZfoHp2JrZPrFeAangmmVW0J1G1wcVu4VOD4
Qjp4z9VA==",
      "encryptedConten":
"9SUhyuq5pSCKcqMVaQlpbes41jHsGHcUatfksoyIoS/055f0uUe85OmyZv8lO2Y+jPwPz
q/wG6N5RTWcF9Rm30uWw1vWbgQxeZA4WADG0z2toS8m/ISYhEcqpSXGMSBsJv6X
2hjymIkIEBu4N3ooZA=="
}
```


8. 监管端接收到请求后：

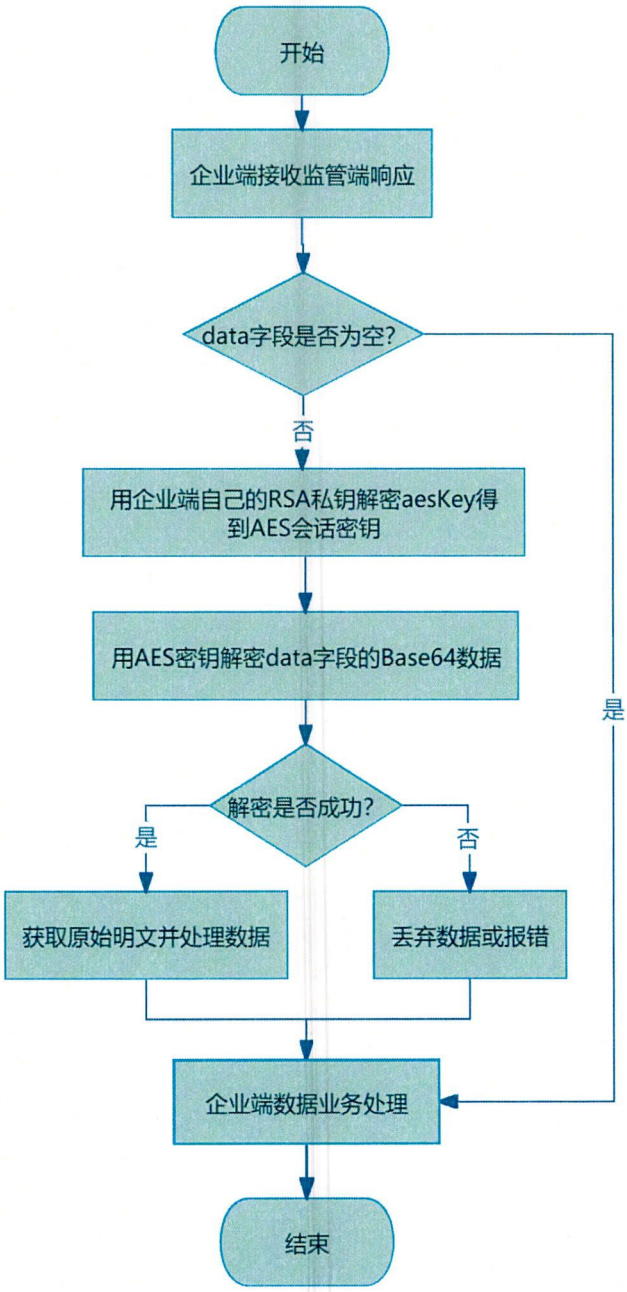
使用企业端 RSA 公钥验证 signature 签名，确保数据来源可信且未被篡改；

使用自身 RSA 私钥解密 encryptedAesKey，还原出原始 AES 密钥；

使用该 AES 密钥解密 encryptedContent，获取明文数据；

9. 验签通过后，监管端处理业务逻辑并返回响应（响应数据同样采用对称加密保护）。

10. 返回响应处理流程



判断状态码看是否需要进行处理：

```
{
    "code": 200,
     "message": "操作成功",
     "aesKey":
"BoueELYLPEIWksGcCD74B3gwxliGuoeNE/5zJCNF10mVz4wOXWleU3pOA2Ua1HpL+NUWk48ab60GS
9cNNiyb924ATpxmjPhFRa5A9qftP9K7Q31zq0/sJkH3dw8nHQjXXhbfOHaSSsWSKghjHCXJuekJUboAzWZ
```

```
vuHy6oTZ4D3bH8QRFmmFbcRHf2zuH6x944jrzJlXx2s3OZxem0osFak61wupLKJEv4oCzvNxP7rvRHfGBC
JXS3m5TGSYVKpGEGg25K+TG3VYCmaHuvDI4SLEZOyF86CWN6swnn4V2lcLeW0/Dj+oVac2VyNzF3
QwBBVy0MPlBcivqvsInhcEv9Q==",
    "data":
"rybzr3fHGHkSGLztIa2zmUXUPZDSepJOOli5zxwnyEL65Oywa06MpY+jUhO+LVEpmIeJTO4Bu2ageTIQ
wOfVpdDO6g0bfUq79pbtSR9ZdfCQXDoasHDdEsmf789B3dXUV8IU05InMqE6IgZtJL4DnFpBgmHX2FszZ
aRKp0GbK5DtN1jaxHOXWv/DPoTof7iYKorMdIApCoo5d3flRQZcQRoI+jl+VHan+a1fXJ1XBXLqKffzgAq
aADOwI0b5/+lXSqm0OTdYfsMczuUT69cC5QN9JV73DwkJt16qAglXIYrKQQgVgxXAUN41e/5f2Seig7HvI
b008YPs7LpiLcGqDag+sSm/Iduyc8WLKf7Lr7gKByssej1j0S1Q0ePN0IuIkzI6K6Gv8UQsOeHyevmrnk8Cpb
T091R3MnCHnTseNkWb9x5GLgojHYa0B2ZLlwQH0mTSEfMxCgcoJNNkV+5m3tN2eH88I+GvRNTluTjg
NVScifw/LOToK8rZepuuLmoHASSObPXZLfAcctcqjKocF+FPMkYwwn3YS/sC0XFgk71eLowlCC9TqjVA2
B3r35VRqOIyQc0HpBq6W1IXXNfHmgwK2frc7WXv7c0dUz+NB11oUOImWZJa5A26Cq1tipzRdt8+XvgSe
1CBb6MVcAuqpg=="
}
```


企业端 RSA 私钥解密 AES 密钥：（图示片段仅供参考）

企业端数据解密

```
// 1. 企业端RSA私钥 解密 AES 密钥
Cipher rsaCipher = Cipher.getInstance(RSA_TRANSFORMATION);
rsaCipher.init(Cipher.DECRYPT_MODE, clientPrivateKey);
byte[] decryptedAesKeyBytes = rsaCipher.doFinal(aesKey);
SecretKey aesKey = new SecretKeySpec(decryptedAesKeyBytes, ALGORITHM);
```

用 AES 密钥解密 data 字段的 Base64 数据：（图示片段仅供参考）

企业端数据解密

```
// 2. AES 解密数据
Cipher aesCipher = Cipher.getInstance(AES_TRANSFORMATION);
aesCipher.init(Cipher.DECRYPT_MODE, aesKey);
byte[] decryptedContent = aesCipher.doFinal(data);
```

得到明文数据后进行业务处理逻辑，明文数据参考数据结构说明。