天翼云·内容安全

API 帮助文档

天翼云科技有限公司

目录

一、 生成证书

1、 1、生成 RSA 私钥

2、

3、 命令：openssl genrsa -out rsa_private_key.pem 1024

4、

5、

6、 说明：以上命令可生成原始 RSA 私钥文件 rsa_private_key.pem，其内容格式为：

7、 -----BEGIN RSA PRIVATE KEY-----

8、 ...

9、 -----END RSA PRIVATE KEY-----

10、

11、 2、通过 RSA 私钥生成 PKCS8 格式私钥

12、

13、 命令：openssl pkcs8 -topk8 -inform PEM -in rsa_private_key.pem -outform PEM -nocrypt -out pkcs8_private_key.pem

14、

15、

16、 说明：以上命令可生成 PKCS8 格式的私钥文件 pkcs8_private_key.pem，其内容格式为：

17、 -----BEGIN PRIVATE KEY-----

18、 ...

19、 -----END PRIVATE KEY-----

20、

21、

22、 注意：请按照接入服务器所采用的语言选择不同格式私钥进行签名，如：PHP 端签名采用 RSA 私钥，C#.NET、C++、Java 端签名均采用 PKCS8 格式私钥。（请妥善保管私钥防止外泄）

23、

24、

25、 3、生成 RSA 公钥证书

```
26、
27、      命令：openssl rsa -in rsa_private_key.pem -pubout -out rsa_public_ke
   y.pem
28、
29、
30、      说明：以上命令可生成公钥文件 rsa_public_key.pem，其内容格式为：
31、       -----BEGIN PUBLIC KEY-----
32、       ...
33、       -----END PUBLIC KEY-----
34、
35、
36、
37、      4、上传公钥证书
38、
39、
40、      完成上述步骤后，私钥文件（rsa_private_key.pem 和 pkcs8_private_key.pem）
   保存在本地使用。
41、
42、      注意：私钥、公钥文件内容都不要进行人为修改，避免出错。
43、
44、      上传公钥证书 rsa_public_key.pem，以便后续认证您的数字签名。
```

二、 生成签名 PHP

```
function  sign($content, $private_content){
        $private_key=openssl_get_privatekey($private_content);
        $sign="";
        openssl_sign($content,$sign,$private_key);
        openssl_free_key($private_key);
        $sign=base64_encode($sign);
        return $sign;
```

```
}
```

三、 生成签名 Java

```
import java.nio.charset.Charset;
import org.apache.commons.codec.binary.Base64;


public class Base64Utils {
    private static final Base64 BASE64 = new Base64();
    private static final Charset DEFAULT_CHARSET = Charset.forName("UTF-8");
    private static final String PREFIX = "XXXXX";


    public static String encode(String source) {
        if (!isEmpty(source)) {
            new Base64();
            String target = PREFIX + source;
            byte[] bytes = BASE64.encode(target.getBytes(DEFAULT_CHARSET));
            return new String(bytes, DEFAULT_CHARSET);
        }


        return source;
    }


    public static String decode(String source) {
        if (!isEmpty(source)) {
            byte[] bytes = BASE64.decode(source.getBytes(DEFAULT_CHARSET));
            String target = new String(bytes, DEFAULT_CHARSET);
            return target.startsWith(PREFIX) ? target.substring(PREFIX.leng
th()) : target;
        }
```

```
        return source;
    }

    private static boolean isEmpty(String str) {
        return str == null || str.length() == 0;
    }
}

import java.io.ByteArrayOutputStream;
import java.security.Key;
import java.security.KeyFactory;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.Signature;
import java.security.interfaces.RSAPrivateKey;
import java.security.interfaces.RSAPublicKey;
import java.security.spec.PKCS8EncodedKeySpec;
import java.security.spec.X509EncodedKeySpec;
import java.util.HashMap;
import java.util.Map;
import javax.crypto.Cipher;

public static String sign(byte[] data, String privateKey) throws Exception {
       byte[] keyBytes = Base64Utils.decode(privateKey);
       PKCS8EncodedKeySpec pkcs8KeySpec = new PKCS8EncodedKeySpec(keyByte
s);
       KeyFactory keyFactory = KeyFactory.getInstance("RSA");
       PrivateKey privateK = keyFactory.generatePrivate(pkcs8KeySpec);
```

```
        Signature signature = Signature.getInstance("SHA256withRSA");
        signature.initSign(privateK);
        signature.update(data);
        return Base64Utils.encode(signature.sign());
}
```

四、 生成签名 C#

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Security.Cryptography;
using System.Text;

public static string sign(string content, string privateKey)
{
            byte[] Data = Encoding.UTF8.GetBytes(content);
            RSACryptoServiceProvider rsa = DecodePemPrivateKey(privateKey);
            SHA256 sh = new SHA256CryptoServiceProvider();
            byte[] signData = rsa.SignData(Data, sh);
            return Convert.ToBase64String(signData);
}
```

五、 接口规范

一、通用输入参数

LoginName 字符串 用户名

RandomNum 字符串 随机数（客户端生成）

SignString 字符串 签名字符串（客户端生成）

注：所有参数只接受 POST 方式提交

二、文字提交

a) URL

http://ctyun.siweicn.com:76/api/AuditTxt

b) 输入参数

AuditString 字符串 审计文字（不超过 1000 字节）

c) 输出参数

ErrorCode 整型 错误代码（0:正常;-1:签名验证错误;-2:已超过套餐数目;-3:程序错误）

AuditResult 整型 审计结果（1:正常;0:异常;-1:出错）

Reason 字符型 异常原因（或出错原因,审计结果为正常时此项留空）

三、文本文件提交

a) URL

http://ctyun.siweicn.com:76/api/AuditTxtFile

b) 输入参数

UploadFile 文件型 上传文件（支持 rar（不支持 rar5）,zip,7z 三种压缩文件（无密码），以及 txt,log.html,htm 等格式的文本文件（此格式列表可在服务器端配置），支持 gb2312 和 utf8 两种编码格式（自动识别））

c) 输出参数

ErrorCode 整型 错误代码（0:正常;-1:签名验证错误;-2:已超过套餐数目;-3:程序错误）

ReturnList ApiTxtOneFile 型列表 审计结果列表

Reason 字符型 出错原因,错误代码为正常时此项留空

ApiTxtOneFile 类型说明：

FileName 字符型 上传文件名

AuditResult 整型 审计结果（1:正常;0:异常;-1:出错）

Reason 字符型 异常原因（或出错原因,审计结果为正常时此项留空）

返回 JSON

程序运行正常：

{"ErrorCode":0,"ReturnList":[{"FileName":"KeyWord.txt","AuditResult":1,"Reason":"normal","NeedReview":1},{ "FileName":"addurl1.txt","AuditResult":0,"Reason":"ad","NeedReview":1},{"FileName":"addurl2.txt","AuditResult":0,"Reason":"contraband","NeedReview":1}],"Reason":""}

程序运行异常：

{"ErrorCode":-1,"ReturnList":[],"Reason":"签名验证错误"}

四、图片文件提交

a) URL

http://ctyun.siweicn.com:76/api/AuditImgFile

b) 输入参数

UploadFile 文件型 上传文件（支持 rar（不支持 rar5）,zip,7z 三种压缩文件（无密码），以及 jpg,jpeg,png,bmp 等格式的图片文件（此格式列表可在服务器端配置））

c) 输出参数

ErrorCode 整型 错误代码（0:正常;-1:签名验证错误;-2:已超过套餐数目;-3:程序错误）

ReturnList ApiImgOneFile 型列表 审计结果列表

Reason 字符型 出错原因,错误代码为正常时此项留空

ApiImgOneFile 类型说明：

FileName 字符型 上传文件名

AuditResult 整型 审计结果（1:正常;0:异常;-1:出错）

Reason 字符型 异常原因（或出错原因,审计结果为正常时此项留空）

返回 JSON

程序运行正常：

{"ErrorCode":0,"ReturnList":[{"FileName":"8ebfe122gy1fmo7mbmjovj20j60asq4n.jpg","AuditResult":0,"Reason":"normal","Score":90,"NeedReview":0},{"FileName":"abd2b4119313b07ec7119c1707d7912397dd8c25.jpg","AuditResult":0,"Reason":" porn","Score":70,"NeedReview":1},{"FileName":"dabe950a304e251f83

2b2d98ae86c9177e3e5378.jpg","AuditResult":0,"Reason":"sexy","Score":93,"NeedReview":0}],"Reason":""}

程序运行异常:

{"ErrorCode":-1,"ReturnList":[],"Reason":"签名验证错误"}

五、视频文件提交

a) URL

http://ctyun.siweicn.com:76/api/AuditVdoFile

b) 输入参数

UploadFile 文件型 上传文件(支持 rar(不支持 rar5),zip,7z 三种压缩文件(无密码),以及 asf,avi,flv,mkv,mp4,rm,rmvb 等格式的视频文件(此格式列表可在服务器端配置))

注:由于视频文件较大,审计时间较长,此处不建议上传压缩文件

c) 输出参数

ErrorCode 整型 错误代码(0:正常;-1:签名验证错误;-2:已超过套餐数目;-3:程序错误)

ReturnList ApiVdoOneFile 型列表 审计结果列表

Reason 字符型 出错原因,错误代码为正常时此项留空

ApiVdoOneFile 类型说明:

FileName 字符型 上传文件名

AuditResult 整型 审计结果(1:正常;0:异常;-1:出错)

Reason 字符型 异常原因(或出错原因,审计结果为正常时此项留空)

ImgList ApiVdoSnapImg 型列表 违规截图列表(审计结果为正常、审计出错时此项留空)

ApiVdoSnapImg 类型说明:

TimeValue 字符型 图片时间点(格式 HH:mm:ss)

ImageByte 字节列表 图片 Byte 字节(可由客户端获取后另存为图片)

返回 JSON

程序运行正常：

{"ErrorCode":0,"ReturnList":[{"FileName":"123.avi","AuditResult":0,"Reason":"normal","Score":90,"NeedReview":0,"ImgList":["00:01:00":,"ImageByte":"xxx"]}],"Reason":""}

程序运行异常：

{"ErrorCode":-1,"ReturnList":[],"Reason":"签名验证错误"}