基于量子随机数的服务器密码机SDK接口说明文档

> 版本：`v1.0.1`

目录

前提说明

请求前缀（格式）

http://IP:port/cm/v1

(具体信息根据实际部署情况替换，默认端口 `7002`)

全局认证

除身份认证接口外，其余接口请求头均需携带认证参数（`Authorization`：`Bearer ${token}`，token请替换为身份认证成功后的实际返回值，且值与Bearer之间存在一个用于分割的空格）

返回结果

参数名	参数类型	参数描述
requestId	string	请求编号
code	number	结果代码(0-成功)
msg	string	结果消息
data	object	数据对象

接口调用流程

身份认证 ==> 计算会话密钥 ==> 接口调用 ==> 解密接口返回数据

其中，计算会话密钥使用用户**私钥**解密后进行**异或**计算得到，解密接口返回的数据采用 `SM4ECB` 模式，需要客户端具备 `SM4` 解密能力。

身份认证

接口说明

> 身份鉴别，获取身份凭证token

接口URL

> /auth/getToken

请求方式

> POST

Content-Type

> application/json

请求Body参数

```
{
    "clientId": "",
    "ra": "",
    "clientSecret": ""
}
```

参数名	参数类型	是否必传	参数描述
clientId	string	是	客户端身份编号
ra	string	是	客户端产生的加密后的随机数（明文为16字节）
clientSecret	string	是	客户端身份密钥

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "rb":"",
        "token":""
    }
}
```

参数名	参数类型	参数描述
rb	string	服务端随机数（需使用客户端自身私钥解密后获得）
token	string	身份凭证

SM2算法

SM2根据数量获取密钥对

接口说明

> 获取指定数量的密钥对

接口URL

> /sm2/getKeyByNumber

请求方式

> POST

Content-Type

> application/json

请求Body参数

```
{
    "number": 1,
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
number	number	是	请求数量
resEncode	string	是	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "keyId":"",
        "publicKey":"",
        "privateKey":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
keyId	string	密钥编号
publicKey	string	公钥
privateKey	string	私钥
resEncode	string	输出结果编码规则（HEX, BASE64）

SM2根据密钥Id获取密钥对

接口说明

获取指定keyId对应的密钥值

接口URL

/sm2/getKeyByKeyId

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "keyIds": [""],
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
keyIds	Array[string]	是	密钥编号集合
resEncode	string	是	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "keyId":"",
        "publicKey":"",
        "privateKey":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
keyId	string	密钥编号
publicKey	string	公钥
privateKey	string	私钥
resEncode	string	输出结果编码规则（HEX, BASE64）

SM2加密

接口说明

sm2算法加密

接口URL

/sm2/encrypt

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "publicKey": "",
    "keyEncode": "BASE64",
    "plaintext": "",
    "plaintextEncode": "UTF8",
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
publicKey	string	是	公钥
keyEncode	string	是	公钥编码规则（HEX, BASE64）
plaintext	string	是	原文（待加密数据）
plaintextEncode	string	是	原文编码规则（UTF8, HEX, BASE64）
resEncode	string	是	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "ciphertext":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
ciphertext	string	密文（已加密数据）
resEncode	string	密文编码规则（HEX, BASE64）

SM2解密

接口说明

sm2算法解密

接口URL

/sm2/decrypt

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "privateKey": "",
    "keyEncode": "BASE64",
    "ciphertext": "",
    "ciphertextEncode": "BASE64",
    "resEncode": "UTF8"
}
```

参数名	参数类型	是否必传	参数描述
privateKey	string	否	私钥
keyEncode	string	否	私钥编码规则（HEX, BASE64）
ciphertext	string	否	密文（待解密数据）
ciphertextEncode	string	否	密文编码规则（HEX, BASE64）
resEncode	string	否	输出结果编码规则（UTF8, HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "plaintext":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
plaintext	string	原文（已解密数据）
resEncode	string	输出结果编码规则（UTF8, HEX, BASE64）

SM2数字签名

接口说明

sm2算法数字签名

接口URL

/sm2/sign

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "publicKey": "",
    "privateKey": "",
    "keyEncode": "BASE64",
    "userID": "",
    "userIDEncode": "UTF8",
    "plaintext": "",
    "plaintextEncode": "UTF8",
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
publicKey	string	是	公钥
privateKey	string	是	私钥
keyEncode	string	是	公钥和私钥编码规则（HEX, BASE64）
userID	string	是	用户ID
userIDEncode	string	是	用户ID编码规则（UTF8, HEX, BASE64）
plaintext	string	是	待签名数据
plaintextEncode	string	是	待签名数据编码规则（UTF8, HEX, BASE64）
resEncode	string	是	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "signResult":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
signResult	string	签名结果
resEncode	string	签名结果编码规则（HEX, BASE64）

SM2验证签名

接口说明

> sm2算法数字签名验证

接口URL

> /sm2/verify

请求方式

> POST

Content-Type

> application/json

请求Body参数

```
{
    "publicKey": "",
    "keyEncode": "BASE64",
    "userID": "",
    "userIDEncode": "UTF8",
    "signData": "",
    "signDataEncode": "BASE64",
    "plaintext": "",
    "plaintextEncode": "UTF8"
}
```

参数名	参数类型	是否必传	参数描述
publicKey	string	是	公钥
keyEncode	string	是	公钥编码规则（HEX, BASE64）
userID	string	是	用户ID
userIDEncode	string	是	用户ID编码规则（UTF8,HEX, BASE64）
signData	string	是	签名数据
signDataEncode	string	是	签名数据编码规则（HEX, BASE64）
plaintext	string	是	验证数据
plaintextEncode	string	是	验证数据编码规则（UTF8, HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "verifyResult": true
    }
}
```

参数名	参数类型	参数描述
verifyResult	boolean	验签结果(true-成功，false-失败)

SM3算法

SM3杂凑计算

接口说明

sm3杂凑计算

接口URL

/sm3/hash

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "message": "",
    "messageEncode": "UTF8",
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
message	string	是	待计算消息
messageEncode	string	是	待计算消息编码规则（UTF8, HEX, BASE64）
resEncode	string	是	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "hashText":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
hashText	string	摘要计算结果
resEncode	string	结果编码规则（HEX, BASE64）

SM4算法

SM4通过KeyId获取密钥

接口说明

> 获取指定keyId对应的密钥值

接口URL

> /sm4/getKeyByKeyId

请求方式

> POST

Content-Type

> application/json

请求Body参数

```
{
    "keyIds": [""],
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
keyIds	Array[string]	否	密钥
resEncode	string	否	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "keyId":"",
        "keyValue":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
keyId	string	密钥编号
keyValue	string	密钥值
resEncode	string	输出结果编码规则（HEX, BASE64）

SM4根据数量获取密钥

接口说明

SM4获取指定数量的密钥

接口URL

/sm4/getKeyByNumber

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "number": 1,
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
number	number	是	请求数量
resEncode	string	是	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "keyId":"",
        "keyValue":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
keyId	string	密钥编号
keyValue	string	密钥值
resEncode	string	输出结果编码规则（HEX, BASE64）

SM4加密

接口说明

> sm4算法加密,支持ECB和CBC模式

接口URL

> /sm4/encrypt

请求方式

> POST

Content-Type

> application/json

请求Body参数

```
{
    "key": "",
    "keyEncode": "BASE64",
    "iv": "",
    "ivEncode": "BASE64",
    "mode": "",
    "plaintext": "",
    "plaintextEncode": "UTF8",
    "resEncode": "BASE64"
}
```

参数名	参数类型	是否必传	参数描述
key	string	否	密钥，16字节
keyEncode	string	否	密钥编码规则（HEX, BASE64）
iv	string	否	iv向量，16字节
ivEncode	string	否	iv向量编码规则（HEX, BASE64）
mode	string	否	模式（ECB/CBC）
plaintext	string	否	原文（待加密数据）
plaintextEncode	string	否	密文编码规则（UTF8, HEX, BASE64）
resEncode	string	否	输出结果编码规则（HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "ciphertext":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
ciphertext	string	密文
resEncode	string	输出结果编码规则（HEX, BASE64）

SM4解密

接口说明

sm4算法解密，支持ECB和CBC模式

接口URL

/sm4/decrypt

请求方式

POST

Content-Type

application/json

请求Body参数

```
{
    "key": "",
    "keyEncode": "BASE64",
    "iv": "",
    "ivEncode": "BASE64",
    "mode": "",
    "ciphertext": "",
    "ciphertextEncode": "BASE64",
    "resEncode": "UTF8"
}
```

参数名	参数类型	是否必传	参数描述
key	string	否	密钥
keyEncode	string	否	密钥编码规则（HEX, BASE64）
iv	string	否	iv向量，长度16字节
ivEncode	string	否	iv向量编码规则（HEX, BASE64）
mode	string	否	模式（ECB/CBC）
ciphertext	string	否	密文（待解密数据）
ciphertextEncode	string	否	密文编码规则（HEX, BASE64）
resEncode	string	否	输出结果编码规则（UTF8, HEX, BASE64）

响应示例

```
{
    "requestId": "",
    "code": 0,
    "msg": "",
    "data": {
        "plaintext":"",
        "resEncode":""
    }
}
```

参数名	参数类型	参数描述
plaintext	string	原文（已解密数据）
resEncode	string	输出结果编码规则（UTF8, HEX, BASE64）