密钥管理服务

SDK

产品文档

腾讯云

文档目录

SDK
SDK使用
C++ SDK

最近更新时间：2018-11-09 11:14:56

开发准备

相关资源

-GitHub地址 ,欢迎贡献代码以及反馈问题。

开发环境

1. 安装openssl的库和头文件
2. 安装libcurl
3. 安装cmake工具
4. 从控制台获取 AppID, SecretID,SecretKey。

SDK配置

下载 github 上提供的源码，集成到您的开发环境。

执行下面的命令：

```
cd ${kms-cpp-sdk}
mkdir -p build
cd build
cmake ..
make
```

sample/kms_sample.cpp 里面有常见的 API 例子，生成的 kms_sample 可以直接运行，生成的 libKMS.a 和 libKMS.so 文件可以放到自己的 lib 文件夹下，inc目录拷贝到自己工程的 include 路径下。

生成客户端对象

```
string secretId="xxxxxx"; #替换为用户的 secretId
string secretKey = "xxxxxx"; #替换为用户的 secretKey
string endpoint = "https://kms-region.api.tencentyun.com"; # 替换为用户的 region，例如 sh 表示上
```

```
海， gz表示广州，bj表示北京
KMSAccount account(endpoint,secretId,secretKey);
```

初始化客户端配置

客户端默认使用 sha1 签名算法，可以调用签名算法修改签名方式

```
account.set_sign_method("sha256");
```

密钥管理操作

创建主密钥

方法原型

```
void create_key(KeyMetadata & meta,const string &Description="",const string & Alias = "" , const string & KeyUsage="ENCRYPT/DECRYPT");
```

参数说明

参数名	类型	默认值	参数描述
KeyMetadata	struct		主密钥属性结构体，该参数返回创建的主密钥属性结构
Description	string	None	主密钥描述
Alias	string	空字符串	主密钥别名
KeyUsage	string	'ENCRYPT/DECRYPT'	主密钥用途：默认是加解密

返回值 KeyMetadata 结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途

属性名称	类型	含义
Alias	string	密钥别名

使用示例

```
KeyMetadata meta ;
string description ="test";
string alias = "kms_test";
string KeyUsage="ENCRYPT/DECRYPT";
account.create_key(meta,Description,Alias,KeyUsage);
```

获取主密钥属性

方法原型

```
void get_key_attributes(const string & KeyId, KeyMetadata & meta);
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
KeyMetadata	struct		主密钥属性结构体，该参数返回创建的主密钥属性结构

返回值 KeyMetadata 结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
KeyMetadata meta;
string keyId="" # 请填写您的keyId
account.get_key_attributes(meta.KeyId,meta);
```

设置主密钥属性

方法原型

```
void set_key_attributes(const string & KeyId, const string & Alias);
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
Alias	string	无	主密钥属性结构体，该参数返回创建的主密钥属性结构

使用示例

```
Alias = "For test";
account.set_key_attributes(KeyId, Alias);
```

获取主密钥列表

方法原型

```
void list_key(vector<string> & keyIds, const int offset= 0 , const int limit = 10);
```

参数说明

参数名	类型	默认值	参数描述
keyIds	vector	无	返回 keyid vector
offset	int	0	返回列表偏移值
limit	int	10	本次返回列表限制个数，不填写默认为返回 10 个

使用示例

```
vector<string> KeyIds;
account.list_key(KeyIds);
for(unsigned int i = 0 ; i < KeyIds.size(); ++i)
cout<<"the "<<i<<" key id is :"<<KeyIds[i]<<endl;
```

生成数据密钥

方法原型

```
void generate_data_key( string &KeyId, const string & KeySpace, int NumberOfBytes,const string & EncryptionContext,string & Plaintext,string &CiphertextBlob);
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
KeySpec	string	None	生成数据密钥算法
NumberOfBytes	int	None	生成指定长度的数据密钥
EncryptionContext	json string	无	生成数据密钥时提供的额外的 json key-value
Plaintext	string	无	生成的数据密钥明文
CiphertextBlob	string	无	生成的数据密钥密文

返回值(入参中)

参数名	类型	参数描述
plaintext	string	表示生成的数据密钥明文
ciphertextBlob	string	表示生成的数据密钥密文

使用示例

```
string KeySpec="AES_128";
string Plaintext,CiphertextBlob;
account.generate_data_key(meta.KeyId,KeySpec,1024,"",Plaintext, CiphertextBlob);
```

启用主密钥

方法原型

```
void enable_key(const string & KeyId);
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
string KeyId= "" // 请填写您的keyId;
account.enable_key(KeyId)
```

禁用主密钥

方法原型

```
void disable_key(const string & KeyId);
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
string KeyId= "" // 请填写您的keyId;
account.disable_key(KeyId)
```

加解密操作

加密

方法原型

```
string encrypt(const string &KeyId , const string & plaintext, const string & EncryptionContext);
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
Plaintext	string	空字符串	明文
EncryptionContext	string	None	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数

返回值

参数名	类型	参数描述
ciphertextBlob	string	表示生成的密文

使用示例

```
string KeyId = "" ; // 请填写您的keyId;
string Plaintest = "test message data"
string CiphertextBlob = account.encrypt(KeyId,Plaintest,"");
```

解密

方法原型

```
string decrypt(const string & CiphertextBlob, const string & EncryptionContext);
```

参数说明

参数名	类型	默认值	参数描述
CiphertextBlob	string	空字符串	密文
EncryptionContext	string	None	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数。

返回值

参数名	类型	参数描述
plaintext	string	表示通过密文解密得到的明文

使用示例

```
Plaintext = account.decrypt(CiphertextBlob,"");
```

Java SDK

最近更新时间：2018-11-09 11:13:40

开发准备

相关资源

GitHub 地址 ,欢迎贡献代码以及反馈问题。

环境依赖

JDK1.7

生成客户端对象

```
//从腾讯云官网查询的云 API 密钥信息
String secretId="";
String secretKey="";
String endpoint = "";
KMSAccount account = new KMSAccount(endpoint,secretId, secretKey);
```

初始化客户端配置

客户端默认使用 sha1 签名算法，可以调用签名算法修改签名方式。

```
account.setSignMethod("sha256");
```

密钥管理操作

创建主密钥

方法原型

```
public KeyMetadata create_key(String Description,String Alias ,String KeyUsage) throws Exception
```

参数说明

参数名	类型	默认值	参数描述
Description	string	无	主密钥描述
Alias	string	无	主密钥别名
KeyUsage	string	无	主密钥用途：默认是加解密

返回值 KeyMetadata结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
String Description = "test";
String Alias= "test";
String KeyUsage = "ENCRYPT/DECRYPT";
KeyMetadata meta = account.create_key(Description , Alias , KeyUsage);
```

获取主密钥属性

方法原型

```
public KeyMetadata get_key_attributes(String KeyId) throws Exception
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	无	主密钥 Id

返回值 KeyMetadata结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
meta = account.get_key_attributes(KeyId);
```

设置主密钥属性

方法原型

```
public void set_key_attributes(String KeyId , String Alias) throws Exception
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	无	主密钥 Id
Alias	string	无	主密钥别名

返回值 无

使用示例

```
Alias = "for test";
account.set_key_attributes(KeyId, Alias);
```

获取主密钥列表

方法原型

```
public void list_key(int offset, int limit,List<String> KeyList) throws Exception
```

参数说明

参数名	类型	默认值	参数描述
offset	int	0	返回列表偏移值。
limit	int	10	本次返回列表限制个数，不填写默认为返回 10 个。
KeyList	list	无	本次返回的 KeyId 列表。

使用示例

```
ArrayList<String> KeyId = new ArrayList<String>();
account.list_key(-1,-1,KeyId);
for(int i = 0 ; i < KeyId.size(); ++i)
System.out.println("the " +Integer.toString(i) + "Key id is " + KeyId.get(i));
```

生成数据密钥

方法原型

```
public String generate_data_key(String KeyId, String KeySpec, int NumberOfBytes , String EncryptionContext,String Plaintext ) throws Exception
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id。
KeySpec	string	None	生成数据密钥算法。
NumberOfBytes	int	None	生成指定长度的数据密钥。
EncryptionContext	string	None	生成数据密钥时提供的额外的 json key-value。
Plaintext	string	无	生成的数据密钥明文。

返回值

参数名	类型	参数描述

参数名	类型	参数描述
plaintext	string	表示生成的数据密钥明文(输入参数返回)
ciphertextBlob	string	表示生成的数据密钥密文
#### 使用示例		

```
String KeySpec = "AES_128";
String Plaintext = "";
String CiphertextBlob = account.generate_data_key(meta.KeyId, KeySpec,1024,"",Plaintext);
System.out.println("the data key string is " + Plaintext);
System.out.println("the encrypted data key string is "+CiphertextBlob);
```

启用主密钥

方法原型

```
public void enable_key(String KeyId) throws Exception
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
account.enable_key(KeyId);
```

禁用主密钥

方法原型

```
public void disable_key(String KeyId) throws Exception
```

参数说明

参数名	类型	默认值	参数描述

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
account.disable_key(KeyId);
```

加解密操作

加密

方法原型

```
def encrypt(self, KeyId=None, Plaintext="", EncryptionContext=None)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
Plaintext	string	空字符串	明文
EncryptionContext	string	None	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数。

返回值

参数名	类型	参数描述
ciphertextBlob	string	表示生成的密文
#### 使用示例		

```
CiphertextBlob = account.encrypt(KeyId, Plaintext,"");
System.out.println("the encrypted data is " + CiphertextBlob);
```

解密

方法原型

```
public String decrypt(String CiphertextBlob , String EncryptionContext)throws Exception
```

参数说明

参数名	类型	默认值	参数描述
CiphertextBlob	string	空字符串	密文
EncryptionContext	string	None	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数。

返回值

参数名	类型	参数描述
plaintext	string	表示通过密文解密得到的明文

使用示例

```
Plaintext = account.decrypt( Plaintext,"");
System.out.println("the decrypted data is " + Plaintext);
```

Python SDK

最近更新时间：2018-11-09 11:26:52

开发准备

相关资源

GitHub 地址 ,欢迎贡献代码以及反馈问题。

环境依赖

Python2.7 目前不支持 Python3。

获取 Python 版本的方法：

Linux shell

```
$python -V

Python 2.7.11
```

Windows cmd

```
D:>python -V
Python 2.7.11
```

如果提示不是内部或者外部命令，请先在 Window 环境变量 PATH 里面添加上 Python 的绝对路径。

生成客户端对象

```
secretId='xxxxxx' #替换为用户的 secretId
secretKey = 'xxxxxx' #替换为用户的 secretKey
endpoint = 'https://kms-region.api.tencentyun.com' # 替换为用户的 region，例如 sh 表示上海， gz 表示广州，bj 表示北京
kms_account = KMSAccount(endpoint,secretId,secretKey)
```

初始化客户端配置

客户端默认使用 sha1 签名算法，可以调用签名算法修改签名方式。

```
kms_account.set_sign_method('sha256')
```

密钥管理操作

创建主密钥

方法原型

```
def create_key(self, Description=None, Alias="", KeyUsage='ENCRYPT/DECRYPT')
```

参数说明

参数名	类型	默认值	参数描述
Description	string	None	主密钥描述
Alias	string	空字符串	主密钥别名
KeyUsage	string	'ENCRYPT/DECRYPT'	主密钥用途：默认是加解密

返回值 KeyMetadata结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
description ='for test'
alias = 'kms_test'
kms_meta = kms_account.create_key(description,alias)
```

获取主密钥属性

方法原型

```
def get_key_attributes(self, KeyId=None)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 KeyMetadata结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
keyId='' # 请填写您的 keyId
key_meta = kms_account.get_key_attributes("kms-awy8dndb")
print key_meta
```

设置主密钥属性

方法原型

```
def set_key_attributes(self, KeyId=None, Alias=None)
```

参数说明

参数名	类型	默认值	参数描述

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
Alias	string	None	主密钥别名

返回值 无

使用示例

```
keyId='' # 请填写您的 keyId
Alias='' # 请填写您的主密钥别名
kms_account.get_key_attributes(keyId,Alias)
```

获取主密钥列表

方法原型

```
def list_key(self, offset=0, limit=10)
```

参数说明

参数名	类型	默认值	参数描述
offset	int	0	返回列表偏移值。
limit	int	10	本次返回列表限制个数，不填写默认为返回 10 个。

返回值 KeyMetadata 结构体 描述如下：

属性名称	类型	含义
totalCount	int	表示所有的密钥个数。
keys	array	key 数组。

使用示例

```
totalCount, keys = kms_account.list_key()
print keys
```

生成数据密钥

方法原型

```
def generate_data_key(self, KeyId=None, KeySpec=None, NumberOfBytes=None, EncryptionContext=None)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id。
KeySpec	string	None	生成数据密钥算法。
NumberOfBytes	int	None	生成指定长度的数据密钥。
EncryptionContext	string	None	生成数据密钥时提供的额外的 json key-value。

返回值

参数名	类型	参数描述
plaintext	string	表示生成的数据密钥明文
ciphertextBlob	string	表示生成的数据密钥密文

使用示例

```
KeySpec = "AES_128"
Plaintext, CiphertextBlob = kms_account.generate_data_key(KeyId, KeySpec)
print "the data key : %s \n the encrypted data key :%s\n" % (Plaintext, CiphertextBlob)
```

启用主密钥

方法原型

```
def enable_key(self, KeyId=None)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
kms_account.enable_key(KeyId)
```

禁用主密钥

方法原型

```
def disable_key(self, KeyId=None)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
kms_account.disable_key(KeyId)
```

加解密操作

加密

方法原型

```
def encrypt(self, KeyId=None, Plaintext="", EncryptionContext=None)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
Plaintext	string	空字符串	明文

参数名	类型	默认值	参数描述
EncryptionContext	string	None	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数。

返回值：

参数名	类型	参数描述
ciphertextBlob	string	表示生成的密文
#### 使用示例		

```
Plaintest = "test message data"
CiphertextBlob = kms_account.encrypt(kms_meta.KeyId, Plaintest)
print "the encrypted data is :%s \n" % CiphertextBlob
```

解密

方法原型

```
def decrypt(self, CiphertextBlob="", EncryptionContext=None)
```

参数说明

参数名	类型	默认值	参数描述
CiphertextBlob	string	空字符串	密文
EncryptionContext	string	None	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数。

返回值 ：

参数名	类型	参数描述
plaintext	string	表示通过密文解密得到的明文
#### 使用示例		

```
Plaintest = kms_account.decrypt(CiphertextBlob)
print "the decrypted data is :%s\n" % Plaintest
```

PHP SDK

最近更新时间：2018-11-09 11:31:15

开发准备

相关资源

GitHub 地址 ,欢迎贡献代码以及反馈问题。

开发环境

1. 依赖环境：PHP5.3.0 版本及以上
2. 从控制台获取 AppID，SecretID，SecretKey。

生成客户端对象

```
// 从腾讯云官网查看云 API 的密钥信息
$secretId = "";
$secretKey = "";
$endPoint = "";
$kms_account = new KMSAccount($endPoint,$secretId,$secretKey);
```

初始化客户端配置

客户端默认使用 sha1 签名算法，可以调用签名算法修改签名方式。

```
account.set_sign_method("sha256");
```

密钥管理操作

创建主密钥

方法原型

```
public function create_key($Alias = NULL, $Description = NULL, $KeyUsage="ENCRYPT/DECRYPT")
```

参数说明

参数名	类型	默认值	参数描述
Description	string	NULL	主密钥描述
Alias	string	NULL	主密钥别名
KeyUsage	string	'ENCRYPT/DECRYPT'	主密钥用途：默认是加解密

返回值 KeyMetadata 结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
$Description = "test";
$Alias = "test";
$KeyUsage= "ENCRYPT/DECRYPT";
$kms_meta = $kms_account->create_key($Alias,$Description,$KeyUsage);
```

获取主密钥属性

方法原型

```
public function get_key_attributes($KeyId = NULL)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 KeyMetadata 结构体 描述如下：

属性名称	类型	含义
KeyId	string	密钥 Id
CreateTime	uinx time	创建时间
Description	string	密钥描述
KeyState	string	密钥状态
KeyUsage	string	密钥用途
Alias	string	密钥别名

使用示例

```
KeyMetadata meta;
string keyId="" # 请填写您的 keyId
$kms_meta = $kms_account->get_key_attributes($keyId);
```

设置主密钥属性

方法原型

```
public function set_key_attributes($KeyId = NULL, $Alias)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
Alias	string	无	设置的主密钥别名

返回值 无

使用示例

```
$Alias = "for test" ;
$kms_account->set_key_attributes($kms_meta->KeyId,$Alias);
```

获取主密钥列表

方法原型

```
public function list_key($offset = 0, $limit = 10)
```

参数说明

参数名	类型	默认值	参数描述
keyIds	vector	无	返回 keyid vector
offset	int	0	返回列表偏移值
limit	int	10	本次返回列表限制个数，不填写默认为返回10个

使用示例

```
$ret_pkg = $kms_account->list_key();
```

生成数据密钥

方法原型

```
public function generate_data_key($KeyId = NULL, $KeySpec = "", $NumberOfBytes = 1024,$EncryptionContext =NULL)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id
KeySpec	string	""	生成数据密钥算法
NumberOfBytes	int	1024	生成指定长度的数据密钥
EncryptionContext	string	NULL	生成数据密钥时提供的额外的json key-value

返回字典中 ：

参数名	类型	参数描述
plaintext	string	表示生成的数据密钥明文(输入参数返回)
ciphertextBlob	string	表示生成的数据密钥密文

使用示例

```
$KeySpec = "AES_128";
$ret_pkg = $kms_account->generate_data_key($kms_meta->KeyId,$KeySpec,1024,"");
$Plaintext = $ret_pkg['plaintext'];
$CiphertextBlob = $ret_pkg['ciphertextBlob'];
```

启用主密钥

方法原型

```
public function enable_key($KeyId = NULL)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
$KeyId= "" // 请填写您的keyId;
$kms_account->enable_key($KeyId);
```

禁用主密钥

方法原型

```
public function disable_key($KeyId= NULL)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	None	主密钥 Id

返回值 无

使用示例

```
$KeyId= "" // 请填写您的 keyId;
$kms_account->disable_key($KeyId);
```

加解密操作

加密

方法原型

```
public function encrypt($KeyId = NULL, $Plaintext=NULL,$EncryptionContext =NULL)
```

参数说明

参数名	类型	默认值	参数描述
KeyId	string	NULL	主密钥 Id
Plaintext	string	NULL	明文
EncryptionContext	string	NULL	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数

返回值

参数名	类型	参数描述
ciphertextBlob	string	表示生成的密文
#### 使用示例		

```
$CiphertextBlob = $kms_account->encrypt($KeyId,$Plaintext);
```

解密

方法原型

```
public function decrypt($CiphertextBlob = NULL,$EncryptionContext = NULL)
```

参数说明

参数名	类型	默认值	参数描述
CiphertextBlob	string	NULL	密文
EncryptionContext	string	NULL	key/value 对的 json 字符串，如果指定了该参数，则在调用 Decrypt API 时需要提供同样的参数。

返回值

参数名	类型	参数描述
plaintext	string	表示通过密文解密得到的明文
#### 使用示例		

```
$Plaintext = $kms_account->decrypt($CiphertextBlob);
```

SDK下载

最近更新时间：2018-11-09 11:35:44

SDK下载

腾讯云KMS目前支持 java、python、php 及 C++ SDK, 后续会支持更多语言。也欢迎广大开发者根据 API 说明开发更多语言的 SDK 版本。

github地址如下：

- Java sdk
- Python sdk
- PHP sdk
- C++ sdk

SDK 使用注意事项

使用 SDK 前至少要获取 secretId， secretKey，endpoint（即请求发到哪个地域，通过内网还是外网访问）。

endpoint 说明：

内网 endpoint： `https://kms-region.api.tencentyun.com`

公网 endpoint： `https://kms-region.api.qcloud.com`

- 如果业务进程也部署在腾讯云的 CVM 子机上，强烈建议使用同地域的内网 endpoint。例如在腾讯云北京地域的 CVM 子机则建议您使用 `https://kms-bj.api.tencentyun.com` 。
  原因是：1）同地域内网时延更低；2）目前 KMS 对于公网下行流量是要收取流量费用的，用内网可以节省这部分的费用。
- region 需用具体地域替换：gz（广州），sh（上海），bj（北京）。公共参数中的 region 值要与域名的 region 值保持一致，如果出现不一致的情况，以域名的 region 值为准，将请求发往域名 region 所指定的地域。

SDK文档

最近更新时间：2017-11-30 20:41:07

KMS SDK 使用说明文档

为了方便开发者更好地使用 KMS 的 SDK，腾讯云提供以下使用说明文档：

示例：Python SDK 使用简介

环境依赖

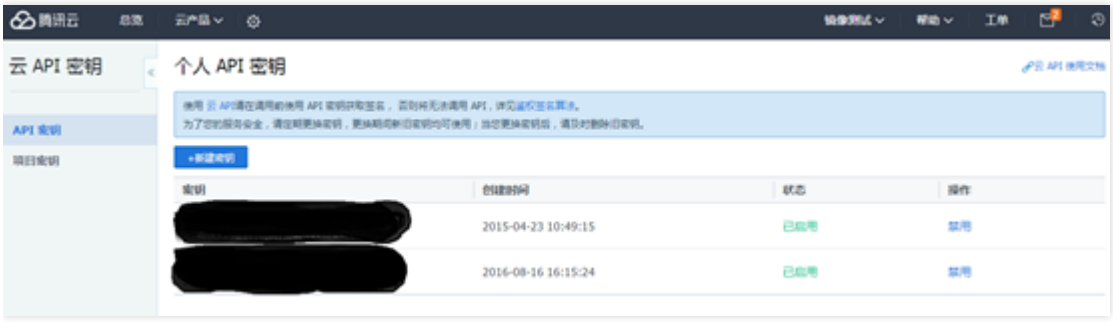
请确保已经安装了Python环境

KMS Python SDK 下载与配置

云 API 密钥使用说明

使用 Python SDK 时，首先需要用户的云 API 密钥，云 API 密钥是对用户身份的合法性验证。获取云 API 密钥的方法如下：登录腾讯云控制台，选择【云产品】-【云 API 密钥】选项

用户可在此新建新的云 API 密钥或使用现有密钥。点击密钥 ID 进入详情页获取使用的密钥 secretId 和对应的 secretKey。



endpoint 说明

endpoint 是使用 KMS 服务的访问地址，同时 endpoint 中也包含了使用的协议，endpoint的格式如下：

- 内网： `https://kms-region.api.tencentyun.com`
- 外网： `https://kms-region.api.qcloud.com`

region 说明

region 需要使用具体地域进行替换，有如下三个地区：gz(广州)，sh(上海)，bj(北京)。划分不同地域有助于不同地域的用户就近选择，提供更好的服务。公共参数中的 region 值要与域名的 region 值保持一致，如果出现不一致的情况，以域名的 region 值为准，将请求发往域名 region 所指定的地域。

内外网区别

如果业务进程也部署在腾讯云的 CVM 子机上，强烈建议使用同地域的内网endpoint：
1) 同地域内网的时延更低；
2) 目前KMS对于公网下行流量是要收取流量费用的，用内网可以节省这部分的费用。

Python SDK下载

下载最新版KMS SDK。

使用 KMS Python SDK

下面的代码也是 Python SDK 中的sample，从创建主密钥、生成数据密钥，加解密，启用禁用密钥等操作来示例密钥管理的操作。

```
#首先从控制台获取对应的secretId和secretKey。和对应的endpoint
try:
secretId = "your secret id"
secretKey = "your secret key"
endpoint = "your endpoint "

kms_account = KMSAccount(endpoint, secretId, secretKey)
```

```
# create a custom master key
Description = "test"
Alias = "test"
KeyUsage = "ENCRYPT/DECRYPT"
kms_meta = kms_account.create_key(Description, Alias, KeyUsage)
print kms_meta

# create a data key
KeySpec = "AES_128"
Plaintext , CiphertextBlob = kms_account.generate_data_key(kms_meta.KeyId, KeySpec)
print "the data key : %s \n the encrypted data key :%s\n" % (Plaintext, CiphertextBlob)

# encrypt the data string
Plaintest = "test message data"
CiphertextBlob = kms_account.encrypt(kms_meta.KeyId, Plaintest)
print "the encrypted data is :%s \n" % CiphertextBlob

# decrypt the encrypted data string
Plaintest = kms_account.decrypt(CiphertextBlob)
print "the decrypted data is :%s\n" % Plaintest

# get key attributes
key_meta = kms_account.get_key_attributes(key_meta.KeyId)
print key_meta

# set key attributes
Alias = "ForTest"
kms_account.set_key_attributes(key_meta.KeyId, Alias)

# disabke a custom key
kms_account.disable_key(key_meta.KeyId)
# enable a custom key
kms_account.enable_key(key_meta.KeyId)

# list key
totalCount, keys = kms_account.list_key()
print keys

except KMSExceptionBase, e:
print "Exception:%s\n" % e
```

SDK更新日志

最近更新时间：2017-11-30 20:42:01

KMS SDK1.0.0

2017-3-13

1. 同时上线c++，java，python，php四个语言版本；
2. sdk封装了密钥生成，加密，解密操作。
3. 支持sha256签名