数据加密服务

常见问题

文档版本 18

发布日期 2025-01-23

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 密钥管理类

1.1 什么是密钥管理?

密钥管理，即密钥管理服务（Key Management Service, KMS），是一种安全、可靠、简单易用的密钥托管服务，帮助您轻松创建和管理密钥，保护密钥的安全。

KMS通过使用硬件安全模块HSM（Hardware Security Module, HSM）保护密钥的安全，所有的用户密钥都由HSM中的根密钥保护，避免密钥泄露。并且HSM模块满足FIPS 140-2 Level 3安全要求。

KMS对密钥的所有操作都会进行访问控制及日志跟踪，提供所有密钥的使用记录，满足审计和合规性要求。

1.2 什么是用户主密钥?

用户主密钥（Customer Master Key，CMK），是用户或云服务通过密钥管理创建的密钥，是一种密钥加密密钥，主要用于加密并保护数据加密密钥。一个用户主密钥可以加密多个数据加密密钥。

用户主密钥分为自定义密钥和默认密钥。

- 自定义密钥

  用户通过密钥管理界面自行创建或导入的密钥。

- 默认密钥

  在用户第一次通过对应云服务使用KMS加密时，云服务自动通过密钥管理为用户创建的密钥，其别名后缀为“/default”。

  默认密钥可通过密钥管理界面进行查询，不支持禁用、计划删除操作。

  表 1-1 默认主密钥列表

密钥别名	对应云服务
obs/default	对象存储服务（Object Storage Service，OBS）
evs/default	云硬盘（Elastic Volume Service，EVS）

密钥别名	对应云服务
ims/default	镜像服务（Image Management Service，IMS）
kps/default	密钥对管理服务（Key Pair Service，KPS）
csms/default	云凭据管理服务（Cloud Secret Management Service，CSMS）
dlf/default	数据治理中心（DataArts Studio）

1.3 什么是默认密钥?

默认密钥，是对象存储服务（Object Storage Service，OBS）等其他云服务自动通过密钥管理为用户创建的密钥，其别名后缀为“/default”。

默认密钥可通过密钥管理界面进行查询，不支持禁用、计划删除操作。

默认密钥托管不计费，仅收取API请求次数费用，超出免费请求次数后，超出部分会进行计费。

表 1-2 默认主密钥列表

密钥别名	对应云服务
obs/default	对象存储服务（Object Storage Service，OBS）
evs/default	云硬盘（Elastic Volume Service，EVS）
ims/default	镜像服务（Image Management Service，IMS）
vbs/default	云硬盘备份（Volume Backup Service，VBS）
sfs/default	弹性文件服务（Scalable File Service，SFS）
kps/default	密钥对管理服务（Key Pair Service，KPS）
csms/default	云凭据管理服务（Cloud Secret Management Service，CSMS）
dlf/default	数据治理中心（DataArts Studio）

说明

默认密钥是在用户第一次通过对应云服务使用KMS加密时自动生成的。

1.4 自定义密钥与默认密钥有什么区别?

自定义密钥和默认密钥的区别，如**表 自定义密钥和默认密钥的区别**所示。

表 1-3 自定义密钥和默认密钥的区别

名称	概念	区别
自定义密钥	是用户自行通过KMS创建或导入的密钥，是一种密钥加密密钥，主要用于加密并保护DEK。 一个自定义密钥可以加密多个DEK。	• 支持禁用、计划删除等操作。 • 创建或导入成功后进行按需计费。
默认密钥	是用户第一次通过对应云服务使用KMS加密时，系统自动生成的，其名称后缀为“/default”。 例如：evs/default	• 不支持禁用、计划删除等操作。 • 使用对应云服务系统自动生成时不计费，调用API请求次数超过20000次后，收取请求费用。

1.5 什么是数据加密密钥?

数据加密密钥是用于加密数据的密钥。

您可以通过KMS创建、加密和解密数据加密密钥。KMS不会存储、管理、跟踪您的数据加密密钥，也不会使用数据加密密钥执行加解密操作。

创建数据加密密钥

KMS仅支持通过调用API接口的方式创建、加密和解密数据加密密钥。创建数据加密密钥有两种方式，如下：

- 调用**create-datakey**接口，返回数据加密密钥的明文和使用您指定的CMK加密后的数据加密密钥的密文。
- 调用**create-datakey-without-plaintext**接口，返回使用您指定的CMK加密后的数据加密密钥的密文。当您需要获取数据加密密钥的明文时，请调用**decrypt-datakey**接口对该密文进行解密。

使用数据加密密钥加密数据

KMS无法使用数据加密密钥加密数据。您可以利用KMS之外的加密库（例如：OpenSSL），使用数据加密密钥对数据进行加密。

1. 根据**创建数据加密密钥**获取数据加密密钥的明文。
2. 使用数据加密密钥的明文加密数据。
3. 删除数据加密密钥的明文，将数据加密密钥的密文和加密后的数据一起存储到安全的存储设备。

使用数据加密密钥解密数据

KMS无法使用数据加密密钥解密数据。您可以利用KMS之外的加密库（例如：OpenSSL），使用数据加密密钥对数据进行解密。

1. 获取您已加密的数据和加密该数据时使用的数据加密密钥的密文。

2. 调用**decrypt-datakey**接口，获取您加密该数据时使用的数据加密密钥的明文。
3. 使用数据加密密钥的明文解密数据。
4. 删除数据加密密钥的明文。

1.6 为什么不能立即删除用户主密钥?

删除密钥是一个需要非常谨慎的操作。操作前，用户需确保使用该密钥加密的相关数据都已完成迁移。因为密钥一旦被删除，所有使用该密钥加密的相关数据都无法解密。因此在删除密钥时，KMS会将该操作推迟7天到1096天执行，推迟时间由用户指定。超过推迟时间，密钥才会被真正删除。在密钥被真正删除之前，如果用户发现该密钥仍然有用，可取消删除操作。KMS通过这种方式来减少用户误操作所带来的损失。

1.7 哪些云服务使用 KMS 加密数据?

对象存储服务、云硬盘、镜像服务、弹性文件服务、文档数据库服务和云数据库借助KMS实现了加密特性。

表 1-4 使用 KMS 加密的云服务列表

服务名称	如何使用
对象存储服务	对象存储服务支持普通方式和服务端加密方式上传和下载对象。当用户使用服务端加密方式上传对象时，数据会在服务端加密成密文后安全地存储在对象存储服务中；用户下载加密对象时，存储的密文会先在服务端解密为明文，再提供给用户。对象存储服务支持KMS托管密钥的服务端加密方式（即SSE-KMS加密方式），该加密方式是通过KMS提供密钥的方式进行服务端加密。 用户如何使用对象存储服务的SSE-KMS加密方式上传对象，具体操作请参见**《对象存储服务控制台指南》**。
云硬盘	在创建云硬盘时，用户启用云硬盘的加密功能，系统将使用用户主密钥产生的数据密钥对磁盘进行加密，则在使用该云硬盘时，存储到云硬盘的数据将会自动加密。 用户如何使用云硬盘加密功能，具体操作请参见**《云硬盘用户指南》**。
镜像服务	用户通过外部镜像文件创建私有镜像时，可启用私有镜像加密功能，选择KMS提供的用户主密钥对镜像进行加密。 用户如何使用镜像服务的私有镜像加密功能，具体操作请参见**《镜像服务用户指南》**。
弹性文件服务	用户通过弹性文件服务创建文件系统时，选择KMS提供的用户主密钥对文件系统进行加密，当使用该文件系统时，存储到文件系统的文件将会自动加密。 用户如何使用弹性文件服务的文件系统加密功能，具体操作请参见**《弹性文件服务用户指南》**。

服务名称	如何使用
云数据库RDS	在购买数据库实例时，用户启用数据库实例的磁盘加密功能，选择KMS提供的用户主密钥对数据库实例的磁盘进行加密，选择磁盘加密后会提高数据的安全性。 用户如何使用云数据库RDS的磁盘加密功能，具体操作请参见《云数据库RDS用户指南》。
文档数据库服务	在购买文档数据库实例时，用户启用文档数据库实例的磁盘加密功能，选择KMS提供的用户主密钥对文档数据库实例的磁盘进行加密，选择磁盘加密后会提高数据的安全性。 用户如何使用文档数据库的磁盘加密功能，具体操作请参见《文档数据库服务用户指南》。

1.8 华为云服务如何使用 KMS 加密数据?

华为云服务一般使用KMS的**信封加密方式**来保护用户的数据。

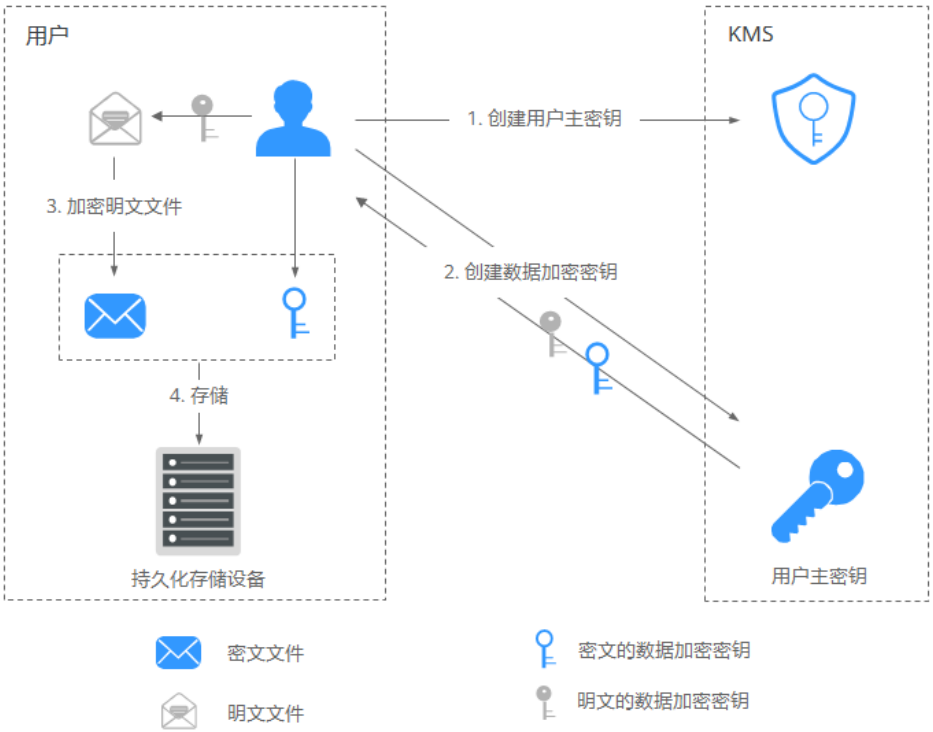
说明

信封加密方式，是一种加密手段，将加密数据的数据密钥封入信封中存储、传递和使用，不再使用用户主密钥直接加解密数据。

信封加密方式加解密原理

- 加密本地文件流程，如图1-1所示。

图 1-1 加密本地文件



流程说明如下：

a. 用户需要在KMS中创建一个**用户主密钥**。

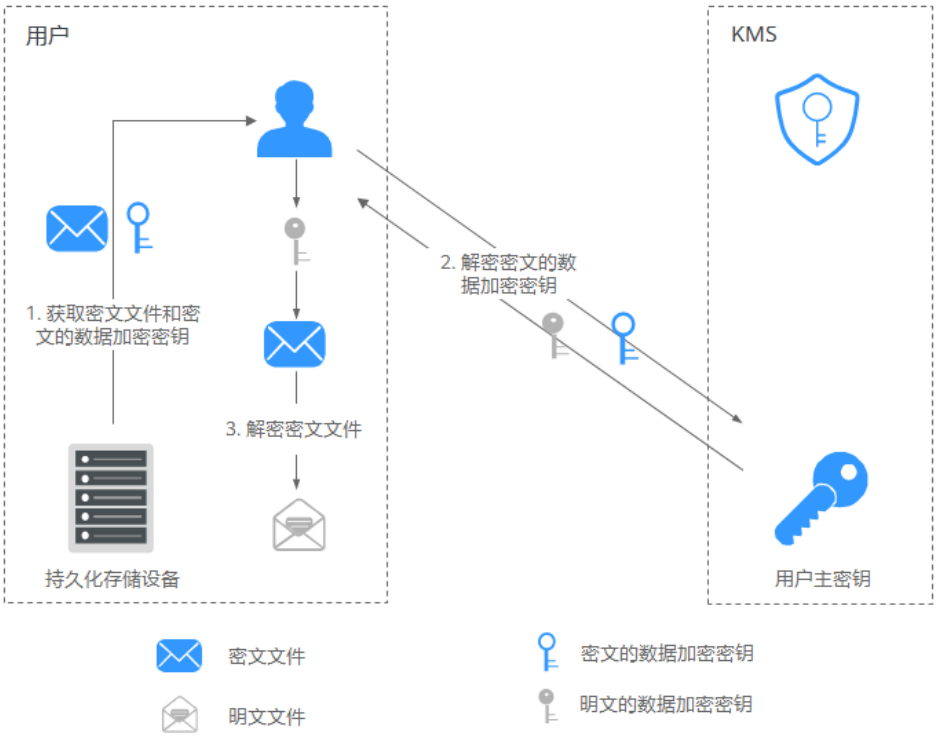
b. 用户调用KMS的“create-datakey”接口创建**数据加密密钥**。用户得到一个**明文的数据加密密钥**和一个**密文的数据加密密钥**。其中**密文的数据加密密钥**是由指定的**用户主密钥**加密**明文的数据加密密钥**生成的。

c. 用户使用**明文的数据加密密钥**来加密明文文件，生成密文文件。

d. 用户将**密文的数据加密密钥**和**密文文件**一同存储到持久化存储设备或服务中。

- 解密本地文件流程，如图1-2所示。

图 1-2 解密本地文件



流程说明如下：

a. 用户从持久化存储设备或服务中读取**密文的数据加密密钥**和**密文文件**。

b. 用户调用KMS的“decrypt-datakey”接口，使用对应的**用户主密钥**（即生成密文的数据加密密钥时所使用的用户主密钥）来解密**密文的数据加密密钥**，取得**明文的数据加密密钥**。

如果对应的用户主密钥被误删除，会导致解密失败。因此，需要妥善管理好用户主密钥。

c. 用户使用**明文的数据加密密钥**来解密密文文件。

关于使用KMS加解密数据的实践教程，请参见**云服务使用KMS加密数据教程**。

1.9 信封加密方式有什么优势？

信封加密方式，是一种加密手段，将加密数据的数据密钥封入信封中存储、传递和使用，不再使用用户主密钥直接加解密数据。

信封加密方式优势如下：

- 相对于KMS提供的另一种加密方式：KMS用户主密钥直接加密

  使用KMS用户主密钥直接加密：是通过KMS界面使用在线工具加解密数据，或者调用KMS的API接口使用指定的用户主密钥直接加密、解密数据。

  使用KMS用户主密钥直接加解密数据仅适用于不大于4KB的小数据加解密场景；而信封加密方式可以在本地对大量数据进行加解密。

  信封加密方式加解密数据，只需要传输数据加密密钥到KMS服务端，无需通过网络传输大量数据。

- 相对于直接加解密的云服务
  - 安全性

    由云服务直接为用户加解密数据：通过因特网将敏感信息从客户手中传递到服务的过程中会存在诸多风险，例如：窃听、钓鱼。

    信封加密方式：KMS通过使用硬件安全模块HSM保护密钥的安全，所有的用户密钥都由HSM中的根密钥保护，避免密钥泄露。
  - 信任和可信证明

    由云服务直接为用户加解密数据：信任和可信证明较难做。用户不一定信任云服务，愿意上传如此敏感的数据；云服务也难以证明自己不会误用和泄露这些数据。

    信封加密方式：KMS对密钥的所有操作都会进行访问控制及日志跟踪，提供所有密钥的使用记录，满足审计和合规性要求。
  - 性能、成本

    由云服务直接为用户加解密数据：大量数据需要通过安全信道传递到服务端，加密后再返回给用户，这一过程，对用户服务的性能影响很大。另外，大量的移动数据会带来巨大的成本。

    信封加密方式：可以通过KMS的密码运算API在线生成数据密钥，用离线数据密钥在本地加密大量数据。

1.10 在 KMS 中创建的自定义密钥的个数是否有限制?

在KMS中创建的自定义密钥的个数是有限制的。

用户最多可以创建100个自定义密钥。启用、禁用和计划删除状态的用户主密钥都会被计入该限制，默认密钥不计入该限制。

1.11 是否可以从 KMS 中导出用户主密钥?

不可以。

为确保用户主密钥的安全，用户只能在KMS中创建和使用用户主密钥，无法导出用户主密钥。

1.12 如果自定义密钥被彻底删除，用户数据是否还可以解密?

不可以。

如果自定义密钥被彻底删除，KMS将不再保留任何该密钥的数据，使用该密钥加密的数据将无法解密；如果自定义密钥没有被彻底删除，则可以通过KMS界面取消删除自定义密钥。

如果自定义密钥是通过KMS导入的密钥，且仅删除了密钥材料，则可以将本地备份的密钥材料再次导入原来的空密钥，回收用户数据。如果密钥材料没有在本地备份，则无法回收用户数据。

1.13 如何使用在线工具加解密数据?

使用在线工具加解密小数据的操作步骤如下所示：

加密数据

步骤1 登录管理控制台。

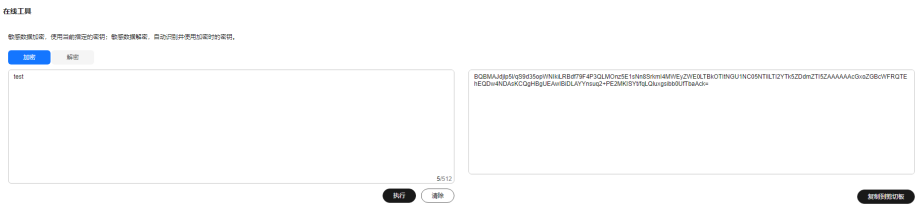
步骤2 单击管理控制台左上角，选择区域或项目。

步骤3 单击页面左侧，选择“安全与合规 > 数据加密服务”，默认进入“密钥管理”界面。

步骤4 单击目标自定义密钥的名称，进入密钥信息页面后，单击“工具”页签。

步骤5 在“加密”文本框中输入待加密的数据，如图1-3所示。

图 1-3 加密数据



步骤6 单击“执行”，右侧文本框显示加密后的密文数据。

说明

- 加密数据时，使用当前指定的密钥加密数据。
- 用户可单击“清除”，清除已输入的数据。
- 用户可单击“复制到剪切板”拷贝加密后的密文数据，并保存到本地文件中。

----结束

说明

在控制台输入的明文，会进行base64编码得到加密后的字符。

如果直接调用API接口进行解密，得到的解密结果是进行了base64编码的内容。需再进行一次base64解码才能得到与控制台输入明文一致的内容。

解密数据

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角，选择区域或项目。

步骤3 单击页面左侧 ，选择“安全与合规 > 数据加密服务”，默认进入“密钥管理”界面。

步骤4 解密数据时，可单击任意“启用”状态的非默认密钥别名，进入该密钥的在线工具页面。

步骤5 单击“解密”，在左侧文本框中输入待解密的密文数据，如图1-4所示。

说明

- 在线工具自动识别并使用数据被加密时使用的密钥解密数据。
- 如果该密钥已被删除，会导致解密失败。

图 1-4 解密数据

步骤6 单击“执行”，右侧文本框中显示解密后的明文数据。

说明

- 用户可直接单击“复制到剪切板”拷贝解密后的明文数据，并保存到本地文件中。
- 在控制台输入的明文，会进行base64编码得到加密后的字符。

  如果直接调用API接口进行解密，得到的解密结果是进行了base64编码的内容。需再进行一次base64解码才能得到与控制台输入明文一致的内容。

----结束

1.14 是否可以更新 KMS 管理的密钥?

不可以。

通过KMS创建的密钥无法更新，用户只能通过KMS创建新密钥，使用新的密钥加解密数据。

1.15 在什么场景下推荐使用导入的密钥?

- 如果用户不想使用KMS中创建的密钥材料，而使用自己的密钥材料，并且可以随时删除密钥材料，或者密钥材料被意外删除，用户可以重新导入相同的密钥材料的情况下，推荐用户使用导入的密钥。
- 当用户把本地的加密数据迁移到云上时，想在云上云下共用一个密钥材料时，可以把云下的密钥材料导入到KMS。

1.16 密钥材料被意外删除时如何处理?

如果密钥材料被意外删除，用户可以在原自定义密钥下将备份的密钥材料重新导入KMS。

须知

导入密钥材料时需要及时备份，重新导入的密钥材料必须与被意外删除的密钥材料保持一致，否则导入会失败。

1.17 默认密钥如何生成?

默认密钥是自动生成的。

在用户第一次通过对应云服务使用KMS加密时，云服务自动通过密钥管理为用户创建的密钥，其别名后缀为“/default”。

默认密钥可通过密钥管理界面进行查询，不支持禁用、计划删除操作。

默认密钥托管不计费，仅收取API请求次数费用，超出免费请求次数后，超出部分会进行计费。

表 1-5 默认主密钥列表

密钥别名	对应云服务
obs/default	对象存储服务（Object Storage Service，OBS）
evs/default	云硬盘（Elastic Volume Service，EVS）
ims/default	镜像服务（Image Management Service，IMS）
kps/default	密钥对管理服务（Key Pair Service，KPS）
csms/default	云凭据管理服务（Cloud Secret Management Service，CSMS）
dlf/default	数据治理中心（DataArts Studio）

1.18 没有权限操作 KMS，该如何处理?

问题描述

用户在KMS中执行查看密钥信息、创建密钥、导入密钥等操作时，显示无法操作KMS。

可能原因

该用户没有KMS系统策略，导致没有权限操作KMS。

解决方法

步骤1 检查该用户是否具有KMS系统策略，KMS Administrator和KMS CMKFullAccess权限。

查看用户所属用户组以及用户组已有的权限。具体操作请参见**用户组及授权**。

如无KMS系统策略，则继续执行**步骤2**。

步骤2 如无系统策略，则为该用户添加系统策略。

- 如需添加管理员权限，则请参见**创建用户并授权使用DEW**进行处理。
- 如需添加自定义策略，则请参见**DEW自定义策略**进行处理。

----结束

1.19 如何修补 OpenSSL 以使用-id-aes256-wrap-pad 包装非对称密钥?

问题描述

默认情况下，OpenSSL命令行工具中未启用包装密码算法-id-aes256-wrap-pad。您可以下载并安装最新版本的OpenSSL，然后对其进行修补，以完成导入非对称密钥所需的信封包装。

解决方法

按照以下说明，使用bash命令创建已有OpenSSL的本地副本，而无需删除或更改OpenSSL的客户端默认安装。

步骤1 以**root**用户完成以下操作步骤，以确保您对使用此命令的目录和二进制文件拥有正确的权限。

sudo su -

步骤2 运行此命令并记下OpenSSL版本。

openssl version

步骤3 在/root/build目录中下载最新的OpenSSL二进制文件。运行以下命令以设置目录。

mkdir $HOME/build

mkdir -p $HOME/local/ssl

cd $HOME/build

步骤4 记住从下载页面(https://www.openssl.org/source/)下载的最新OpenSSL版本。

步骤5 使用以下命令下载并解压缩二进制文件。

步骤6 将openssl-1.1.1d.tar.gz替换为**步骤4**中的最新OpenSSL版本。

curl -O https://www.openssl.org/source/openssl-1.1.1d.tar.gz

tar -zxf openssl-1.1.1d.tar.gz

步骤7 安装修补程序，使gcc工具进行修补，然后编译已下载的二进制文件。

yum install patch make gcc -y

说明

如果您使用的版本与OpenSSL-1.1.1d不同，您可能需要更改目录。您可能需要为更新的OpenSSL的版本更新这些命令，否则此修补程序可能无法正常工作。

步骤8 复制并粘贴此数据块，然后在您的设备上选择输入。

```
sed -i "/BIO_get_cipher_ctx(benc, &ctx);/a\ EVP_CIPHER_CTX_set_flags(ctx,
EVP_CIPHER_CTX_FLAG_WRAP_ALLOW);" $HOME/build/openssl-1.1.1d/apps/enc.c
```

步骤9 运行此命令来编译OpenSSL enc.c文件。

cd $HOME/build/openssl-1.1.1d/

./config --prefix=$HOME/local --openssldir=$HOME/local/ssl

make -j$(grep -c ^processor /proc/cpuinfo)

make install

步骤10 成功安装最新版本的OpenSSL后，此版本的OpenSSL已与$HOME/local/ssl/lib/目录中的二进制文件动态链接，您的shell无法直接运行它。设置环境变量LD_LIBRARY_PATH，以确保有相关的库可用于OpenSSL。

步骤11 由于您需要多次运行修复版本的OpenSSL，请创建一个名为openssl.sh的脚本，以在运行二进制文件之前加载$HOME/local/ssl/lib/路径。

cd $HOME/local/bin/

echo -e '#!/bin/bash \nenv LD_LIBRARY_PATH=$HOME/local/lib/ $HOME/local/bin/openssl "$@"' > ./openssl.sh

步骤12 使用以下命令在脚本上设置执行位。

chmod 755 ./openssl.sh

步骤13 要启动修复版本的OpenSSL，请执行此命令。

$HOME/local/bin/openssl.sh

----结束

1.20 如何修补 GmSSL 以使用-sms4-wrap-pad 包装非对称密钥?

问题描述

默认情况下，GmSSL命令行工具中未启用包装密码算法-sms4-wrap-pad。您可以下载并安装最新版本的GmSSL，然后对其进行修补，以完成导入非对称密钥所需的信封包装。

解决方法

按照以下说明，在没有任何业务的机器，或创建临时机器，使用bash命令安装修补版本的GmSSL。

须知

请使用没有任何业务的机器，或创建临时机器执行以下操作。

步骤1 以**root**用户完成以下操作步骤，以确保您对使用此命令的目录和二进制文件拥有正确的权限。

sudo su -

步骤2 下载的最新GmSSL版本。

下载地址：https://www.github.com/guanzhi/GmSSL/

1. 使用以下命令下载并解压缩二进制文件。
2. 下载最新版本的GmSSL版本。

   curl -LO https://github.com/guanzhi/GmSSL/archive/GmSSL-v2.zip

   unzip GmSSL-GmSSL-v2.zip

   cd GmSSL-GmSSL-v2

步骤3 复制并粘贴此数据块，然后在您的设备上选择输入。

sed -i "/BIO_get_cipher_ctx(benc, &ctx);/a\ EVP_CIPHER_CTX_set_flags(ctx, EVP_CIPHER_CTX_FLAG_WRAP_ALLOW);" apps/enc.c

步骤4 重新编译并安装GmSSL。

./config && make && make install

说明

如果执行GmSSL命令报如下错误：

“gmssl: relocation error: gmssl: symbol PBEPARAM_it, version OPENSSL_xxxx not defined in file libcrypto.so.1.1 with link time reference”

则在当前Shell会话中，执行如下命令:

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$(pwd)

如果您执行此命令时更改过当前路径，请先cd到编译结果的目录。

cd GmSSL-GmSSL-v2 再执行上面的export命令。

----结束

1.21 KMS 支持的密钥算法类型

表 1-6 KMS 支持的密钥算法类型

密钥类型	算法类型	密钥规格	说明	用途
对称密钥	AES	• AES_256	AES对称密钥	小量数据的加解密或用于加解密数据密钥。

密钥类型	算法类型	密钥规格	说明	用途
对称密钥	SM4	• SM4	国密SM4对称密钥	小量数据的加解密或用于加解密数据密钥。
摘要密钥	SHA	• HMAC_256 • HMAC_384 • HMAC_512	SHA摘要密钥	• 数据防篡改 • 数据完整性校验
摘要密钥	SM3	• HMAC_SM3	国密SM3摘要密钥	• 数据防篡改 • 数据完整性校验
非对称密钥	RSA	• RSA_2048 • RSA_3072 • RSA_4096	RSA非对称密钥	小量数据的加解密或数字签名。
	ECC	• EC_P256 • EC_P384	椭圆曲线密码，使用NIST推荐的椭圆曲线	数字签名
非对称密钥	SM2	• SM2	国密SM2非对称密钥	小量数据的加解密或数字签名。

1.22 请求 KMS 异常，错误码 401，应该如何处理?

问题现象

请求KMS报错或使用云服务加密功能报错。

报错信息为：httpcode=401,code=APIGW.0301,Msg=Incorrect IAM authentication information: current ip:xx.xx.xx.xx refused。

可能原因

用户在IAM服务中设置了访问控制。

IAM控制策略默认范围为全地址访问，如果用户设置了允许访问的IP地址或者网段，则未允许的IP地址/网段均无法访问KMS，或无法使用云服务加密特性。

解决方法

- 通过云服务控制台访问KMS（如OBS加密）：需要开放10.0.0.0/8、11.0.0.0/8、26.0.0.0/8网段。
- 通过API调用KMS接口：根据访问的源IP地址设置开放。

开放IP地址操作步骤

步骤1 登录管理控制台。

步骤2 单击页面左侧 ☰，选择“管理与监督 > 统一身份认证服务 IAM”，默认进入“用户”界面。

步骤3 选择“安全设置 > 访问控制”，查看“允许访问的IP地址区间”和“允许访问的IP地址或网段”是否包含请求的源IP地址。

说明

需在“控制台访问”和“API访问”中都包含请求的源IP地址。

----结束

1.23 进行 SM2 签名时，如何计算 SM3 摘要?

使用SM2密钥签名时，仅支持对消息摘要签名。

根据GBT32918国家标准，计算SM2签名值时，消息摘要不是对原始消息直接计算SM3摘要，而是对Z(A)和M的拼接值计算的摘要。其中M是待签名的原始消息，Z(A)是GBT32918中定义的用户A的杂凑值。

以JAVA为例，参考如下示例代码：

```
public class Sm2SignDataPreprocessing {

    private static final String ECC_A =
"FFFFFFFEFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF00000000FFFFFFFFFFFFFFFC";
    private static final String ECC_B =
"28E9FA9E9D9F5E344D5A9E4BCF6509A7F39789F515AB8F92DDBCBD414D940E93";
    private static final String ECC_GX =
"32C4AE2C1F1981195F9904466A39C9948FE30BBFF2660BE1715A4589334C74C7";
    private static final String ECC_GY =
"BC3736A2F4F6779C59BDCEE36B692153D0A9877CC62A474002DF32E52139F0A0";

   // 签名者ID，本示例使用国密标准定义的缺省值"1234567812345678"
   private static final String SM2_ID = "31323334353637383132333435363738";

   public static void main(String[] args) throws Exception {

      // SM2公钥，BASE64编码格式，仅做示例，实际使用请替换
      String sm2PubKey = "MFkwEwYHKoZIzj0CAQYIKoEcz1UBgi0DQgAEsuOq/
EjQeYUD9h7lIyqi3pQ6SWL7hTXjUJWmSIZAcnj" +
            "h9c0QcdbwzaCfI8iyyPCetX0QZl5NHrBoYLYxJpvbFg==";

      byte[] pubKeyBytes = Base64.decodeBase64(sm2PubKey.getBytes(StandardCharsets.UTF_8));
      X509EncodedKeySpec keySpec = new X509EncodedKeySpec(pubKeyBytes);
      KeyFactory keyFactory = KeyFactory.getInstance("EC", new BouncyCastleProvider());
      ECPublicKey ecPublicKey = (ECPublicKey) keyFactory.generatePublic(keySpec);

      // 待签名的原始消息，仅做示例，实际使用请替换
      byte[] dataToDigest = new byte[]{1, 2, 3, 4};

      // Z(A)是GBT32918中定义的用户A的杂凑值
      byte[] zsm = getSm2Z(ecPublicKey.getQ().getAffineXCoord().getEncoded(),
            ecPublicKey.getQ().getAffineYCoord().getEncoded());

      // Z(A)和M的拼接值计算的摘要，并打印
      byte[] dataToSign = getSm3SignData(zsm, dataToDigest);
      System.out.println(Base64.encodeBase64String(dataToSign));

      // 其他语言实现或执行此示例应有如下输出：He+qiM2MmtNxlV/EB4vqkcP60XgG08z/8nWQdp/IS5c=
   }

   // 计算Z(A)
```



```
    private static byte[] getSm2Z(byte[] x, byte[] y) throws DecoderException {
        SM3Digest sm3 = new SM3Digest();
        byte[] userId = Hex.decodeHex(SM2_ID.toCharArray());
        byte[] byteEccA = Hex.decodeHex(ECC_A);
        byte[] byteEccB = Hex.decodeHex(ECC_B);
        byte[] byteEccGx = Hex.decodeHex(ECC_GX);
        byte[] byteEccGy = Hex.decodeHex(ECC_GY);
        int len = userId.length * 8;
        sm3.update((byte) (len >> 8 & 255));
        sm3.update((byte) (len & 255));
        sm3.update(userId, 0, userId.length);
        sm3.update(byteEccA, 0, byteEccA.length);
        sm3.update(byteEccB, 0, byteEccB.length);
        sm3.update(byteEccGx, 0, byteEccGx.length);
        sm3.update(byteEccGy, 0, byteEccGy.length);
        sm3.update(x, 0, x.length);
        sm3.update(y, 0, y.length);
        byte[] md = new byte[sm3.getDigestSize()];
        sm3.doFinal(md, 0);
        return md;
    }

    // 计算Z(A)和M的拼接值的摘要
    private static byte[] getSm3SignData(byte[] z, byte[] sourceData) {
        SM3Digest sm3 = new SM3Digest();
        sm3.update(z, 0, z.length);
        sm3.update(sourceData, 0, sourceData.length);
        byte[] md = new byte[sm3.getDigestSize()];
        sm3.doFinal(md, 0);
        return md;
    }

}
```

1.24 调用 encrypt-data 接口，返回的密文和明文有什么关系?

encrypt-data接口返回的密文数据基础长度为124字节。密文数据由“密钥ID”、“加密算法”、“密钥版本”、“密文摘要”等字段拼接组成。

明文按照每个分组16个字节进行处理，不足16字节的，补码至16字节。所以密文长度为124+Ceil(明文长度/16)*16，并将结果进行Base64编码。

以4字节明文输入为例，先计算结果124+Ceil(4/16)*16=140。140字节进行Base64编码后为188字节。

说明

Ceil为向上取整函数。 Ceil(a) = 1, a的取值范围是(0,1]。

1.25 KMS 如何保护创建的密钥?

KMS的开发机制能够预防任何人以明文形式访问您的密钥。KMS使用加密机（HSM）确保密钥的机密性和完整性。明文KMS密钥由加密机加密并保护。仅在处理您的加密请求时，KMS才会在存储器中使用这些密钥。

1.26 如何使用非对称密钥对公钥对签名结果进行验签?

在公私钥对的使用场景中，通常使用私钥进行签名，使用公钥进行验签。公钥可以分发给需要使用的业务主体，业务主体对关键数据进行验签。密钥管理服务提供获取公钥的接口get-publickey。

本示例使用RSA_3072主密钥，密钥用途为SIGN_VERIFY。通过调用密钥管理服务sign接口。请求体如下：

```
{
 "key_id": "key_id_value",
 "message": "MTIzNA==",
 "signing_algorithm": "RSASSA_PSS_SHA_256",
 "message_type": "RAW"
}
```

获取的结果如下：

```
{
 "key_id": "key_id_value",
 "signature": "xxx"
}
```

通过get-publickey获取公钥后，对signature进行验签。

```
public class RawDataVerifyExample {

    /**
     * 基础认证信息：
     * - ACCESS_KEY: 华为云账号Access Key
     * - SECRET_ACCESS_KEY: 华为云账号Secret Access Key, 敏感信息，建议密文存储
     * - IAM_ENDPOINT: 华为云IAM服务访问终端地址,详情见https://developer.huaweicloud.com/endpoint?IAM
     * - KMS_REGION_ID: 华为云KMS支持的地域,详情见https://developer.huaweicloud.com/endpoint?DEW
     * - KMS_ENDPOINT: 华为云KMS服务访问终端地址,详情见https://developer.huaweicloud.com/endpoint?
DEW
     */
    private static final String ACCESS_KEY = System.getenv("HUAWEICLOUD_SDK_AK");
    private static final String SECRET_ACCESS_KEY = System.getenv("HUAWEICLOUD_SDK_SK");
    private static final String IAM_ENDPOINT = "https://<IamEndpoint>";
    private static final String KMS_REGION_ID = "<RegionId>";
    private static final String KMS_ENDPOINT = "https://<KmsEndpoint>";

    private static final int SALT_LENGTH = 32;
    private static final int TRAILER_FIELD = 1;
    public static final String RSA_PUBLIC_KEY_BEGIN = "-----BEGIN PUBLIC KEY-----\n";
    public static final String RSA_PUBLIC_KEY_END = "-----END PUBLIC KEY-----";

    // 示例签名数据，BASE64编码格式，原文信息1234
    private static final String RWA_DATA = "MTIzNA==";

    // 通过密钥管理服务sign接口获取的签名值
    private static final String SIGN = "xxx";
    public static void main(String[] args) throws Exception {

        final String keyId = args[0];

        publicKeyVerify(keyId);
    }
   public static void publicKeyVerify(String keyId) throws Exception {

        // 1.准备访问华为云的认证信息
        final BasicCredentials auth = new BasicCredentials()
                .withIamEndpoint(IAM_ENDPOINT).withAk(ACCESS_KEY).withSk(SECRET_ACCESS_KEY);

        // 2.初始化SDK，传入认证信息及KMS访问终端地址
        final KmsClient kmsClient = KmsClient.newBuilder()
```



```
            .withRegion(new Region(KMS_REGION_ID, KMS_ENDPOINT)).withCredential(auth).build();

        // 3.获取公钥信息,返回的是PKCS8格式
        final ShowPublicKeyRequest showPublicKeyRequest = new ShowPublicKeyRequest()
                .withBody(new OperateKeyRequestBody().withKeyId(keyId));
        final ShowPublicKeyResponse showPublicKeyResponse =
kmsClient.showPublicKey(showPublicKeyRequest);

        // 4.获取公钥字符串
        final String publicKeyStr = showPublicKeyResponse.getPublicKey().replace(RSA_PUBLIC_KEY_BEGIN, "")
                .replaceAll("\n", "").replace(RSA_PUBLIC_KEY_END, "");

        // 5.解析公钥
        final X509EncodedKeySpec keySpec = new
X509EncodedKeySpec(Base64.getDecoder().decode(publicKeyStr));
        final KeyFactory keyFactory = KeyFactory.getInstance("RSA", new BouncyCastleProvider());
        final PublicKey publicKey = keyFactory.generatePublic(keySpec);

        // 6.进行验签
        final Signature signature = getSignature();
        signature.initVerify(publicKey);
        signature.update(commonHash(Base64.getDecoder().decode(RWA_DATA)));

        // 7.验签结果
        assert signature.verify(Base64.getDecoder().decode(SIGN));

    }
    private static Signature getSignature() throws Exception {
        Signature signature= Signature.getInstance("NONEwithRSASSA-PSS", new BouncyCastleProvider());
        MGF1ParameterSpec mgfParam = new MGF1ParameterSpec("SHA256");
        PSSParameterSpec pssParam = new PSSParameterSpec("SHA256", "MGF1", mgfParam, SALT_LENGTH,
TRAILER_FIELD);
        signature.setParameter(pssParam);
        return signature;
    }
    private static byte[] commonHash(byte[] data) {
        byte[] digest;
        try {
            MessageDigest md = MessageDigest.getInstance("SHA256",
BouncyCastleProvider.PROVIDER_NAME);
            md.update(data);
            digest = md.digest();
        } catch (Exception e) {
            throw new RuntimeException("Digest failed.");
        }
        return digest;
    }
}
```

1.27 外部导入的密钥支持轮转吗?

外部导入的密钥不支持轮转，且当用户清除外部导入的密钥材料时，再次导入的密钥材料，须与上次导入的密钥材料保持一致。

1.28 密钥管理服务支持离线加解密数据吗?

正常来说，密钥管理服务提供小数据加解密的公开API“encrypt-data”和“decrypt-data”。该接口的运算基于密钥管理服务，密钥管理服务对密文进行一定的包装。因此是不支持离线加解密数据的。

但是针对非对称密钥场景，密钥管理服务遵从通用规范，不会对密文进行重新包装。因此是支持公钥离线加密，私钥在线解密的。

本文提供如下示例：

使用RSA_3072主密钥，密钥用途为ENCRYPT_DECRYPT。使用公钥离线加密"hello world!"，调用decrypt-data使用私钥进行解密，加解密使用的算法为"RSA/ECB/OAEPWithSHA-256AndMGF1Padding"：

```
public class RsaEncryptDataExample {
    /**
     * 基础认证信息：
     * - ACCESS_KEY: 华为云账号Access Key，获取方式请参见获取AK/SK。
     * - SECRET_ACCESS_KEY: 华为云账号Secret Access Key, 敏感信息，建议密文存储，获取方式请参见获取AK/SK。
     * - IAM_ENDPOINT: 华为云IAM服务访问终端地址，获取方式请参见IAM终端节点。
     * - KMS_REGION_ID: 华为云KMS支持的地域,，获取方式请参见地区和终端节点。
     * - KMS_ENDPOINT: 华为云KMS服务访问终端地址，获取方式请参见终端节点。
     * - 认证用的ak和sk直接写到代码中有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。
     * - 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
     */
    private static final String ACCESS_KEY = System.getenv("HUAWEICLOUD_SDK_AK");
    private static final String SECRET_ACCESS_KEY = System.getenv("HUAWEICLOUD_SDK_SK");
    private static final String IAM_ENDPOINT = "https://<IamEndpoint>";
    private static final String KMS_REGION_ID = "<RegionId>";
    private static final String KMS_ENDPOINT = "https://<KmsEndpoint>";

    private static final String RSA_PUBLIC_KEY_BEGIN = "-----BEGIN PUBLIC KEY-----\n";
    private static final String RSA_PUBLIC_KEY_END = "-----END PUBLIC KEY-----";

    private static final String RSAES_OAEP_SHA_256 = "RSA/ECB/OAEPWithSHA-256AndMGF1Padding";

    private static final String SHA_256 = "SHA-256";

    private static final String MGF1 = "MGF1";

    private static final String HELLO_WORLD = "hello world!";

    public static void main(String[] args) throws Exception {

        final String keyId = args[0];

        publicKeyEncrypt(keyId);
    }

    private static void publicKeyEncrypt(String keyId) throws NoSuchAlgorithmException,
InvalidKeySpecException,
            NoSuchPaddingException, InvalidAlgorithmParameterException, InvalidKeyException,
            IllegalBlockSizeException, BadPaddingException {

        // 1.准备访问华为云的认证信息
        final BasicCredentials auth = new BasicCredentials()
                .withIamEndpoint(IAM_ENDPOINT).withAk(ACCESS_KEY).withSk(SECRET_ACCESS_KEY);

        // 2.初始化SDK，传入认证信息及KMS访问终端地址
        final KmsClient kmsClient = KmsClient.newBuilder()
                .withRegion(new Region(KMS_REGION_ID, KMS_ENDPOINT)).withHttpConfig(new HttpConfig()
                        .withIgnoreSSLVerification(true)).withCredential(auth).build();

        // 3.获取公钥信息,返回的是PKCS8格式
        final ShowPublicKeyRequest showPublicKeyRequest = new ShowPublicKeyRequest()
                .withBody(new OperateKeyRequestBody().withKeyId(keyId));
        final ShowPublicKeyResponse showPublicKeyResponse =
kmsClient.showPublicKey(showPublicKeyRequest);

        // 4.获取公钥字符串
        final String publicKeyStr = showPublicKeyResponse.getPublicKey().replace(RSA_PUBLIC_KEY_BEGIN, "")
                .replaceAll("\n", "").replace(RSA_PUBLIC_KEY_END, "");

        // 5.获取公钥二进制
        final X509EncodedKeySpec keySpec = new
X509EncodedKeySpec(Base64.getDecoder().decode(publicKeyStr));
```



```
        final KeyFactory keyFactory = KeyFactory.getInstance("RSA", new BouncyCastleProvider());
        final PublicKey publicKey = keyFactory.generatePublic(keySpec);

        // 6.公钥离线加密字符串"hello world!"
        final Cipher cipher = Cipher.getInstance(RSAES_OAEP_SHA_256);
        final OAEPParameterSpec oaepParameterSpec = new OAEPParameterSpec(SHA_256, MGF1,
                new MGF1ParameterSpec(SHA_256), PSource.PSpecified.DEFAULT);
        cipher.init(Cipher.ENCRYPT_MODE, publicKey, oaepParameterSpec);
        final byte[] cipherData = cipher.doFinal(HELLO_WORLD.getBytes(StandardCharsets.UTF_8));

        // 7.私钥在线解密
        final DecryptDataRequest decryptDataRequest = new DecryptDataRequest()
                .withBody(new DecryptDataRequestBody().withKeyId(keyId)
                        .withEncryptionAlgorithm(DecryptDataRequestBody.EncryptionAlgorithmEnum.RSAES_OAEP
_SHA_256)
                        .withCipherText(Base64.getEncoder().encodeToString(cipherData)));

        final DecryptDataResponse decryptDataResponse = kmsClient.decryptData(decryptDataRequest);

        assert HELLO_WORLD.equals(decryptDataResponse.getPlainText());
    }

}
```

1.29 为什么 SM2 算法签名结果不是 64 字节?

SM2算法正常不进行编码的签名长度为64个字节，即为R+S，各32个字节；密钥管理服务(KMS)针对签名结果使用ASN.1进行编码。

根据SM2密码算法使用规范，SM2算法签名数据格式的ASN.1定义为：

```
SM2Signature ::= SEQUENCE{
    R    INTEGER,-- 签名值的第一部分
    S    INTEGER    -- 签名值的第二部分}
```

其中R和S的长度各为256位。但是在大数INTEGER做der编码时，如果首字节的第一个二进制位为1时，前面需要补00字节，所以导致der编码长度多一个字节，这种情况下SM2签名值的编码长度最大会有两个字节的差距。

其中R和S分别对应：

70个字节，R值，S值均不补00：3044+0220+32个字节R+0220+32个字节S

71个字节，(1)R值补00：3045+022100+32个字节R+0220+32个字节S (2)S值补00：3045+0220+32个字节R+022100+32个字节S

72个字节，R值，S值均补00：3046+022100 +32个字节R+022100+32个字节S

部分特殊场景会存在69字节。R或S的数据存在前导0时，在实际编码过码中，会删掉前导00的长度。但不是所有情况都会删除前导0。是否删除，取决于其后一个字节是否存在补位（即第一个字节最高位是否为1）。

说明

ASN.1整体按照TLV三元组<Type,Length,Value>进行存储。

关于0x30和0x02，是ASN.1的tag标志域，指明数据类型，占用一个字节。BER_TYPE_INTEGER 0x02 BER_TYPE_SEQUENCE 0x30

0x44，0x45，0x46表明SEQUENCE的长度，分别对应70，71和72字节。

0x20 表明INTEGER的长度，对应32字节。

1.30 如何将原始 EC 私钥转换成 PKCS8 格式的私钥对象?

背景

EC私钥是个大整数，但是在密钥对导入场景，要求私钥对象是ASN.1编码之后，再对数据进行二进制编码，得到DER格式。仅通过OPENSSL命令无法获得。

本示例介绍如何将一个原始的256长度的EC私钥转换成PKCS8格式的私钥。

环境准备

- 搭建JAVA环境，并引入bouncy castle 1.78及其以后的版本。
- 环境安装OpenSSL 1.1.1m及以后的版本。

将私钥转换为 PKCS8 对象

我们有一个secp256k1的私钥，原始私钥16进制表示如下：

```DC23DA6E913444ABADCE2F42A3B7DC3958569948633EE80AEC46ACCA02523495```

说明

示例的私钥仅作演示使用，请勿用到实际生产环境。

通过如下代码，将私钥转换成PKCS8对象：

```java
import org.bouncycastle.jcajce.provider.asymmetric.ec.BCECPrivateKey;
import org.bouncycastle.jcajce.provider.asymmetric.ec.BCECPublicKey;
import org.bouncycastle.jce.ECNamedCurveTable;
import org.bouncycastle.jce.interfaces.ECPrivateKey;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.bouncycastle.jce.spec.ECNamedCurveParameterSpec;
import org.bouncycastle.jce.spec.ECPrivateKeySpec;
import org.bouncycastle.jce.spec.ECPublicKeySpec;
import org.bouncycastle.math.ec.ECPoint;

import java.math.BigInteger;
import java.security.KeyFactory;
import java.security.NoSuchAlgorithmException;
import java.security.PublicKey;
import java.security.Security;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.InvalidParameterSpecException;
import java.util.Base64;

public class RawEcPrivateKeyToPKCS8Object {
    public static void main(String[] args)
        throws InvalidParameterSpecException, NoSuchAlgorithmException, InvalidKeySpecException {

        Security.addProvider(new BouncyCastleProvider());

        KeyFactory keyFactory = KeyFactory.getInstance("ECDSA", new BouncyCastleProvider());

        ECNamedCurveParameterSpec ecSpec = ECNamedCurveTable.getParameterSpec("secp256k1");
        BigInteger d = new
BigInteger("DC23DA6E913444ABADCE2F42A3B7DC3958569948633EE80AEC46ACCA02523495", 16);
        ECPrivateKeySpec ecPrivateKeySpec = new ECPrivateKeySpec(d, ecSpec);
        BCECPrivateKey ec = new BCECPrivateKey("EC", ecPrivateKeySpec,
BouncyCastleProvider.CONFIGURATION);
```


```

```
        ECPoint q = ecSpec.getG().multiply(((ECPrivateKey) ec).getD());
        ECPublicKeySpec pubSpec = new ECPublicKeySpec(q, ecSpec);
        PublicKey publicKey = keyFactory.generatePublic(pubSpec);

        BCECPrivateKey ec2 = new BCECPrivateKey("EC", ec.engineGetKeyParameters(), (BCECPublicKey)
publicKey,
            ecPrivateKeySpec.getParams(), BouncyCastleProvider.CONFIGURATION);

        System.out.println(Base64.getEncoder().encodeToString(ec2.getEncoded()));
    }
}
```

执行上述代码，获得如下输出:

```ignorelang
MIGNAgEAMBAGByqGSM49AgEGBSuBBAAKBHYwdAIBAQQg3CPabpE0RKutzi9Co7fcOVhWmUhjPugK7Easyg
JSNJWgBwYFK4EEAAqhRANCAAQWiYvQT8cyVJx3wN85fXw0c2Ppv3SEsgnDaB96rWlz6G2bf2WhBJVD/jF5zb
+5/oxgVIOYDe8EwqYtBwhIJ3Yh
```

使用ASN.1解码工具查看

```
 <SEQUENCE>
  <INTEGER/>
  <SEQUENCE>
   <OBJECT_IDENTIFIER Comment="ANSI X9.62 public key type"
Description="ecPublicKey">1.2.840.10045.2.1</OBJECT_IDENTIFIER>
   <OBJECT_IDENTIFIER Comment="SECG (Certicom) named elliptic curve"
Description="secp256k1">1.3.132.0.10</OBJECT_IDENTIFIER>
  </SEQUENCE>
  <OCTET_STRING>
   <SEQUENCE>
    <INTEGER>1</INTEGER>
    <OCTET_STRING>0xDC23DA6E913444ABADCE2F42A3B7DC3958569948633EE80AEC46ACCA02523495</
OCTET_STRING>
    <NODE Sign="a0">
     <OBJECT_IDENTIFIER Comment="SECG (Certicom) named elliptic curve"
Description="secp256k1">1.3.132.0.10</OBJECT_IDENTIFIER>
    </NODE>
    <NODE Sign="a1">
     <BIT_STRING
Bits="520">0x000416898BD04FC732549C77C0DF397D7C347363E9BF7484B209C3681F7AAD6973E86D9B7F6
5A1049543FE3179CDBFB9FE8C605483980DEF04C2A62D070848277621</BIT_STRING>
    </NODE>
   </SEQUENCE>
  </OCTET_STRING>
 </SEQUENCE>
```

将如下内容写入到命名为“ec_private_key.pem”的文件中:

```ignorelang
-----BEGIN PRIVATE KEY-----
MIGNAgEAMBAGByqGSM49AgEGBSuBBAAKBHYwdAIBAQQg3CPabpE0RKutzi9Co7fcOVhWmUhjPugK7Easyg
JSNJWgBwYFK4EEAAqhRANCAAQWiYvQT8cyVJx3wN85fXw0c2Ppv3SEsgnDaB96rWlz6G2bf2WhBJVD/jF5zb
+5/oxgVIOYDe8EwqYtBwhIJ3Yh
-----END PRIVATE KEY-----
```

执行如下命令：查看EC密钥的信息：

```shell
openssl ec -in ec_private_key.pem -text
```

```ignorelang
read EC key
Private-Key: (256 bit)
```



```
priv:
    dc:23:da:6e:91:34:44:ab:ad:ce:2f:42:a3:b7:dc:
    39:58:56:99:48:63:3e:e8:0a:ec:46:ac:ca:02:52:
    34:95
pub:
    04:16:89:8b:d0:4f:c7:32:54:9c:77:c0:df:39:7d:
    7c:34:73:63:e9:bf:74:84:b2:09:c3:68:1f:7a:ad:
    69:73:e8:6d:9b:7f:65:a1:04:95:43:fe:31:79:cd:
    bf:b9:fe:8c:60:54:83:98:0d:ef:04:c2:a6:2d:07:
    08:48:27:76:21
ASN1 OID: secp256k1
writing EC key
-----BEGIN EC PRIVATE KEY-----
MHQCAQEEINwj2m6RNESrrc4vQqO33DlYVpllYz7oCuxGrMoCUjSVoAcGBSuBBAAK
oUQDQgAEFomL0E/HMlScd8DfOX18NHNj6b90hLIJw2gfeq1pc+htm39loQSVQ/4x
ec2/uf6MYFSDmA3vBMKmLQcISCd2lQ==
-----END EC PRIVATE KEY-----

```

后续可以正常执行转成DER命令：

```shell
openssl pkcs8 -topk8 -inform PEM -outform DER -in ec_private_key.pem -out ec_private_key.der -nocrypt```

# 1.31 如何将原始 SM2 私钥转换成 PKCS8 格式的私钥对象?

## 背景

SM2私钥是一个256bits的大整数，但是在密钥对导入场景，要求私钥对象是ASN.1编码之后，再对数据进行二进制编码，得到DER格式。仅通过OPENSSL命令无法获得。

本示例介绍如何将一个原始的SM2私钥转换成PKCS8格式的私钥。

## 环境准备

- 搭建JAVA环境，并引入bouncy castle 1.78及其以后的版本。
- 环境安装OpenSSL 1.1.1m及以后的版本。

## 将私钥转换为 PKCS8 对象

sm2p256v1的私钥示例， 原始私钥16进制表示如下：

```5AF8F37036DFF6C303B65DF52674B6B8269BCDF1D70DC6305D93F975DCA2469A```

**说明**

- 示例的私钥仅作演示使用，请勿用到实际生产环境。
- 本示例不包含修补GmSSL启用包装密钥算法。

通过如下代码，将私钥转换成PKCS8对象：

```java
import org.bouncycastle.asn1.gm.GMNamedCurves;
import org.bouncycastle.asn1.x9.X9ECParameters;
import org.bouncycastle.jcajce.provider.asymmetric.ec.BCECPrivateKey;
import org.bouncycastle.jcajce.provider.asymmetric.ec.BCECPublicKey;
import org.bouncycastle.jce.ECNamedCurveTable;
import org.bouncycastle.jce.interfaces.ECPrivateKey;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
```



```
import org.bouncycastle.jce.spec.ECNamedCurveParameterSpec;
import org.bouncycastle.jce.spec.ECParameterSpec;
import org.bouncycastle.jce.spec.ECPrivateKeySpec;
import org.bouncycastle.jce.spec.ECPublicKeySpec;
import org.bouncycastle.math.ec.ECPoint;

import java.math.BigInteger;
import java.security.KeyFactory;
import java.security.NoSuchAlgorithmException;
import java.security.PublicKey;
import java.security.Security;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.InvalidParameterSpecException;
import java.util.Base64;

public class RawSm2PrivateKeyToPKCS8Object {

    private static final String CURVE_NAME = "sm2p256v1";

    private static final X9ECParameters X9_EC_PARAMETERS = GMNamedCurves.getByName(CURVE_NAME);

    private static final ECParameterSpec EC_PARAMETER_SPEC = new
ECNamedCurveParameterSpec(CURVE_NAME,
        X9_EC_PARAMETERS.getCurve(),
        X9_EC_PARAMETERS.getG(), X9_EC_PARAMETERS.getN());

    public static void main(String[] args)
        throws InvalidParameterSpecException, NoSuchAlgorithmException, InvalidKeySpecException {
        Security.addProvider(new BouncyCastleProvider());

        KeyFactory keyFactory = KeyFactory.getInstance("ECDSA", new BouncyCastleProvider());

        // 替换成待导入的私钥,如下内容仅为示例，实际使用时请替换
        String hexPrivateKey =
"5AF8F37036DFF6C303B65DF52674B6B8269BCDF1D70DC6305D93F975DCA2469A";

        BigInteger d = new BigInteger(hexPrivateKey, 16);
        ECPrivateKeySpec ecPrivateKeySpec = new ECPrivateKeySpec(d, EC_PARAMETER_SPEC);
        BCECPrivateKey ec = new BCECPrivateKey("EC", ecPrivateKeySpec,
BouncyCastleProvider.CONFIGURATION);

        ECNamedCurveParameterSpec ecSpec = ECNamedCurveTable.getParameterSpec(CURVE_NAME);
        ECPoint q = ecSpec.getG().multiply(((ECPrivateKey) ec).getD());
        ECPublicKeySpec pubSpec = new ECPublicKeySpec(q, ecSpec);
        PublicKey publicKey = keyFactory.generatePublic(pubSpec);

        BCECPrivateKey ec2 = new BCECPrivateKey("EC", ec.engineGetKeyParameters(), (BCECPublicKey)
publicKey,
            ecPrivateKeySpec.getParams(), BouncyCastleProvider.CONFIGURATION);

        System.out.println(Base64.getEncoder().encodeToString(ec2.getEncoded()));
    }

}
```

执行上述代码，获得如下输出：

```ignorelang
MIGTAgEAMBMGByqGSM49AgEGCCqBHM9VAYItBHkwdwIBAQQgWvjzcDbf9sMDtl31JnS2uCabzfHXDcYwXZ
P5ddyiRpqgCgYIKoEcz1UBgi2hRANCAATOxASdjgJXIhnJOF/bTZwE0mnK6BEGoJIOmFHQzpMbrlu+hoRIXOGX/
pEtsZqJsKZLD99/iDjB5bM9y/f9GaEC
```

使用ASN.1解码工具查看：

```
 <SEQUENCE>
  <INTEGER/>
  <SEQUENCE>
```



```
  <OBJECT_IDENTIFIER Comment="ANSI X9.62 public key type"
Description="ecPublicKey">1.2.840.10045.2.1</OBJECT_IDENTIFIER>
  <OBJECT_IDENTIFIER Comment="China GM Standards Committee"
Description="sm2ECC">1.2.156.10197.1.301</OBJECT_IDENTIFIER>
 </SEQUENCE>
 <OCTET_STRING>
  <SEQUENCE>
   <INTEGER>1</INTEGER>
   <OCTET_STRING>0x5AF8F37036DFF6C303B65DF52674B6B8269BCDF1D70DC6305D93F975DCA2469A</
OCTET_STRING>
   <NODE Sign="a0">
    <OBJECT_IDENTIFIER Comment="China GM Standards Committee"
Description="sm2ECC">1.2.156.10197.1.301</OBJECT_IDENTIFIER>
   </NODE>
   <NODE Sign="a1">
    <BIT_STRING
Bits="520">0x0004CEC4049D8E02572219C9385FDB4D9C04D269CAE81106A0920E9851D0CE931BAE5BBE86
84485CE197FE912DB19A89B0A64B0FDF7F8838C1E5B33DCBF7FD19A102</BIT_STRING>
   </NODE>
  </SEQUENCE>
 </OCTET_STRING>
</SEQUENCE>
```

将如下内容写入到命名为“sm2_private_key.pem”的文件中：

```ignorelang
-----BEGIN PRIVATE KEY-----
MIGTAgEAMBMGByqGSM49AgEGCCqBHM9VAYItBHkwdwIBAQQgWvjzcDbf9sMDtl31JnS2uCabzfHXDcYwXZ
P5ddyiRpqgCgYIKoEcz1UBgi2hRANCAATOxASdjgJXIhnJOF/bTZwE0mnK6BEGoJIOmFHQzpMbrlu+hoRIXOGX/
pEtsZqJsKZLD99/iDjB5bM9y/f9GaEC
-----END PRIVATE KEY-----
```

执行如下命令，查看EC密钥的信息：

```shell
openssl ec -in sm2_private_key.pem -text
```

```ignorelang
read EC key
Private-Key: (256 bit)
priv:
    5a:f8:f3:70:36:df:f6:c3:03:b6:5d:f5:26:74:b6:
    b8:26:9b:cd:f1:d7:0d:c6:30:5d:93:f9:75:dc:a2:
    46:9a
pub:
    04:ce:c4:04:9d:8e:02:57:22:19:c9:38:5f:db:4d:
    9c:04:d2:69:ca:e8:11:06:a0:92:0e:98:51:d0:ce:
    93:1b:ae:5b:be:86:84:48:5c:e1:97:fe:91:2d:b1:
    9a:89:b0:a6:4b:0f:df:7f:88:38:c1:e5:b3:3d:cb:
    f7:fd:19:a1:02
ASN1 OID: sm2p256v1
NIST CURVE: SM2
writing EC key
-----BEGIN EC PRIVATE KEY-----
MHcCAQEEIFr483A23/bDA7Zd9SZ0trgmm83x1w3GMF2T+XXcokaaoAoGCCqBHM9V
AYItoUQDQgAEzsQEnY4CVyIZyThf202cBNJpyugRBqCSDphR0M6TG65bvoaESFzh
l/6RLbGaibCmSw/ff4g4weWzPcv3/RmhAg==
-----END EC PRIVATE KEY-----

```

后续可以正常执行转成DER命令：

```shell
openssl pkcs8 -topk8 -inform PEM -outform DER -in sm2_private_key.pem -out sm2_private_key.der -nocrypt
```


2 凭据管理类

2.1 TaurusDB 凭据是新增的轮转凭据类型吗?

不是。**TaurusDB凭据**就是原来的**GaussDB凭据**。

GaussDB(for MySQL)服务更名为TaurusDB。所以，自动托管华为云TaurusDB数据库的**GaussDB凭据**，更名为**TaurusDB凭据。**

更名公告请参见：**GaussDB(for MySQL)服务更名为TaurusDB**

2.2 为什么凭据版本状态不能删除?

“SYSCURRENT”和“SYSPREVIOUS”为服务内建的凭据状态，不支持删除。

2.3 RDS 凭据设置轮转周期为什么与实际轮转周期不一致?

RDS凭据新增轮转时，由于定时任务一个小时触发一次，不会即时生效，所以轮转时间有可能会比预期时间晚一小时。

2.4 轮转 TaurusDB 凭据时，失败记录提示“The API does not exist or has not been published in the environment”如何处理?

问题描述

“TaurusDB凭据”开启轮转时，失败记录提示“The API does not exist or has not been published in the environment”。

问题原因

FunctionGraph轮转函数中的域名地址不正确。

解决方法

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角，选择区域或项目。

步骤3 单击页面左侧，选择“安全与合规 > 数据加密服务”。

步骤4 在左侧导航树，选择“凭据管理 > 凭据列表”。

步骤5 在凭据列表，定位到目标凭据，单击凭据名称，进入凭据详情页。

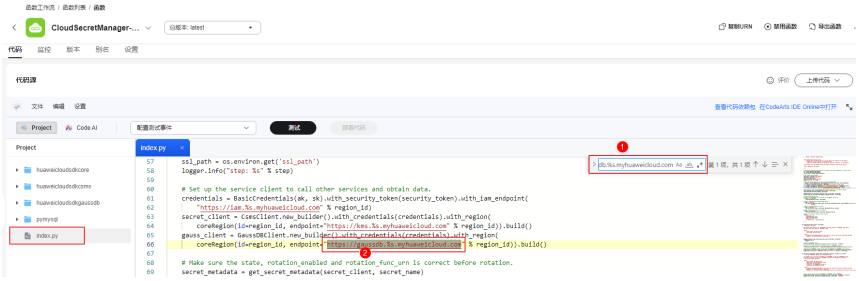
步骤6 在“当前版本”区域，单击“轮转失败”后的数字，查看凭据轮转失败记录是否包含“The API does not exist or has not been published in the environment”。若是，执行后续步骤。

图 2-1 查看凭据轮转失败原因

步骤7 返回凭据详情页，在“凭据信息”区域，单击FunctionGraph轮转函数，跳转到FunctionGraph服务控制台。

步骤8 定位到“index.py”文件，在代码源中搜索“https://gaussdb.%s.myhuaweicloud.com”。

图 2-2 编辑 index.py 文件



步骤9 调用云数据库TaurusDB的API接口查询数据库引擎的版本，查询并复制当前“Region”的TaurusDB数据库域名地址。

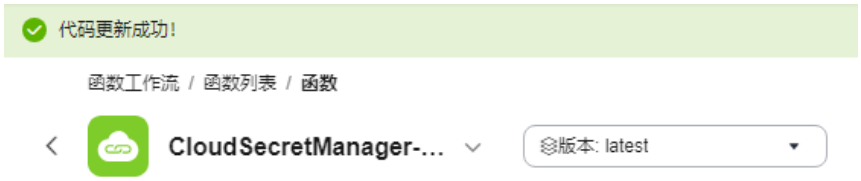
图 2-3 查询 TaurusDB 数据库域名地址

步骤10 替换**步骤8**“index.py”文件中的“https://gaussdb.%s.myhuaweicloud.com”为复制的域名地址，单击“部署代码”。

图 2-4 修改域名地址

当FunctionGraph服务控制台页面，显示横幅“代码更新成功”，表示本次修改已成功。

图 2-5 代码更新成



----结束

后续操作

登录管理控制台，选择“凭据管理 > 凭据列表”，定位到目标“TaurusDB凭据”，再次开启轮转，验证是否成功。

3 密钥对管理类

3.1 密钥对的配额是多少?

密钥对当前免费使用，但是有配额限制，每个用户最多可创建1999个密钥对。

3.2 如何创建密钥对?

通过管理控制台创建密钥对

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角的，选择区域或项目。

步骤3 单击页面左上方的，选择“安全与合规 > 数据加密服务”。

步骤4 在左侧导航树中，选择“密钥对管理”，进入“密钥对管理”页面。

步骤5 默认进入“账号密钥对”页签，根据用户使用需求，自主选择创建私有密钥对或者账号密钥对。

步骤6 单击“创建密钥对”，进入“创建密钥对”页面，输入密钥对名称，如图3-1所示。

图 3-1 创建密钥对

步骤7 （可选）选择密钥对类型。当您账号未开通账号密钥对时，默认创建SSH_RSA_2048的密钥对。

说明

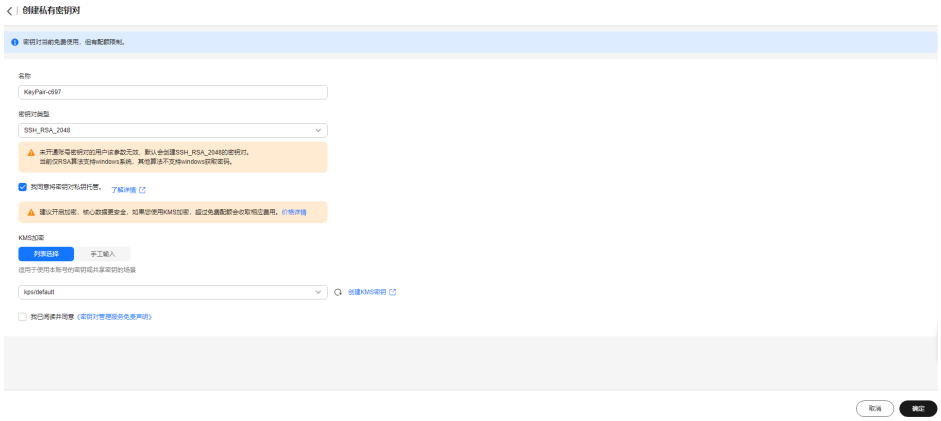
当前仅RSA算法支持windows系统。

步骤8 如果需要托管私钥，请阅读并勾选“我同意将密钥对私钥托管”。在“KMS加密”下拉列表中选择加密密钥。如果不需要托管私钥，请跳过此步骤。

说明

- KPS采用KMS提供的加密密钥对私钥进行加密，用户使用密钥对的KMS加密功能时，可选择KMS创建的默认密钥“kps/default”。
- 用户使用KMS创建的自定义密钥，具体操作请参见**创建密钥**。
- 用户使用授权密钥，创建授权后，用户可以通过切换手工输入方式，输入密钥ID后使用被授权密钥加密。授权密钥操作可参见**创建授权**。

图 3-2 托管私钥



步骤9 请阅读《密钥对管理服务免责声明》并勾选“我已阅读并同意《密钥对管理服务免责声明》”。

步骤10 单击“确定”，浏览器自动执行下载任务，下载私钥文件，并弹出提示对话框。

步骤11 用户需要根据提示对话框的提示信息，保存私钥文件。

须知

- 如果用户没有进行私钥托管，为保证安全，私钥只能下载一次，请妥善保管。如果不慎遗失，您可以通过重置密码或重置密钥对的方式，重新给弹性云服务器绑定密钥对，具体可参照**解绑密钥对后用户无法登录ECS时如何处理?** 进行处理。
- 如果用户已授权华为云托管私钥，可根据需要将托管的私钥导出使用。

步骤12 单击“确定”，密钥对创建成功。密钥对创建成功后，用户可以在密钥对列表里看到新创建的密钥对信息，包括密钥对的“名称”、“指纹”、“状态”、“私钥”等。

说明

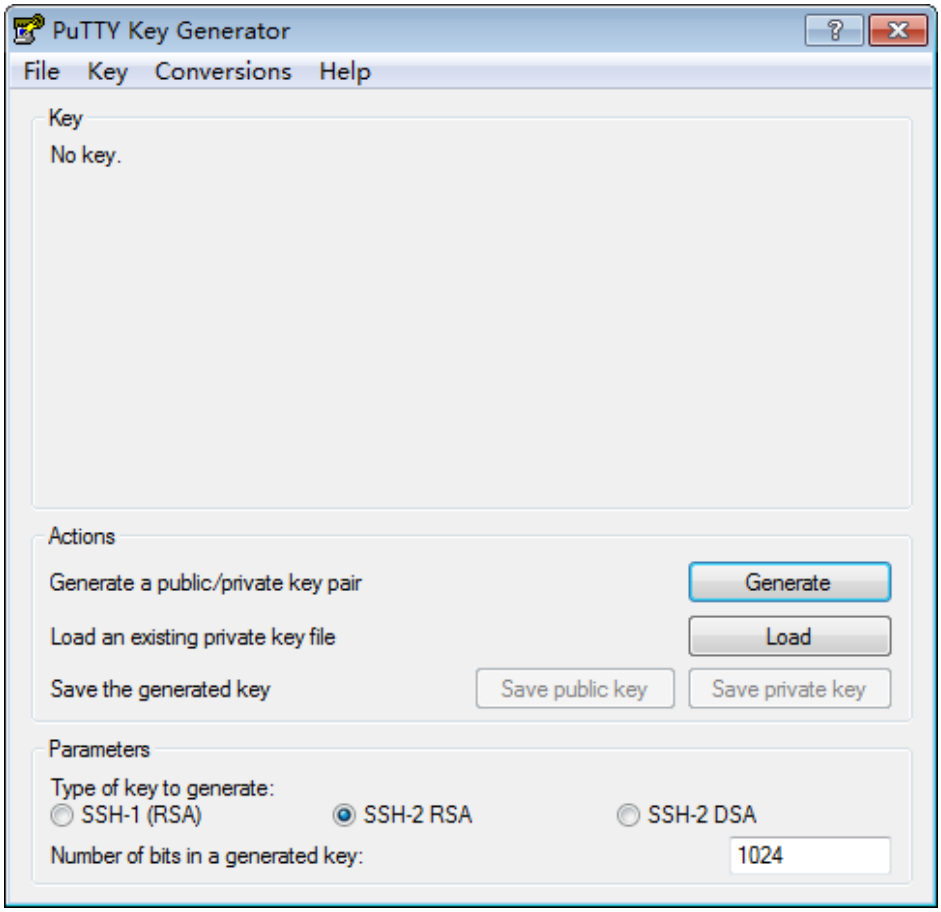
创建密钥对成功后，请确认私钥已成功下载到本地，并注意妥善保管私钥。

----结束

通过 PuTTYgen 工具创建密钥对

步骤1 生成公钥和私钥文件，双击“PUTTYGEN.exe”，打开“PuTTY Key Generator”。如图3-3所示。

图 3-3 PuTTY Key Generator



步骤2 请根据表3-1设置参数。

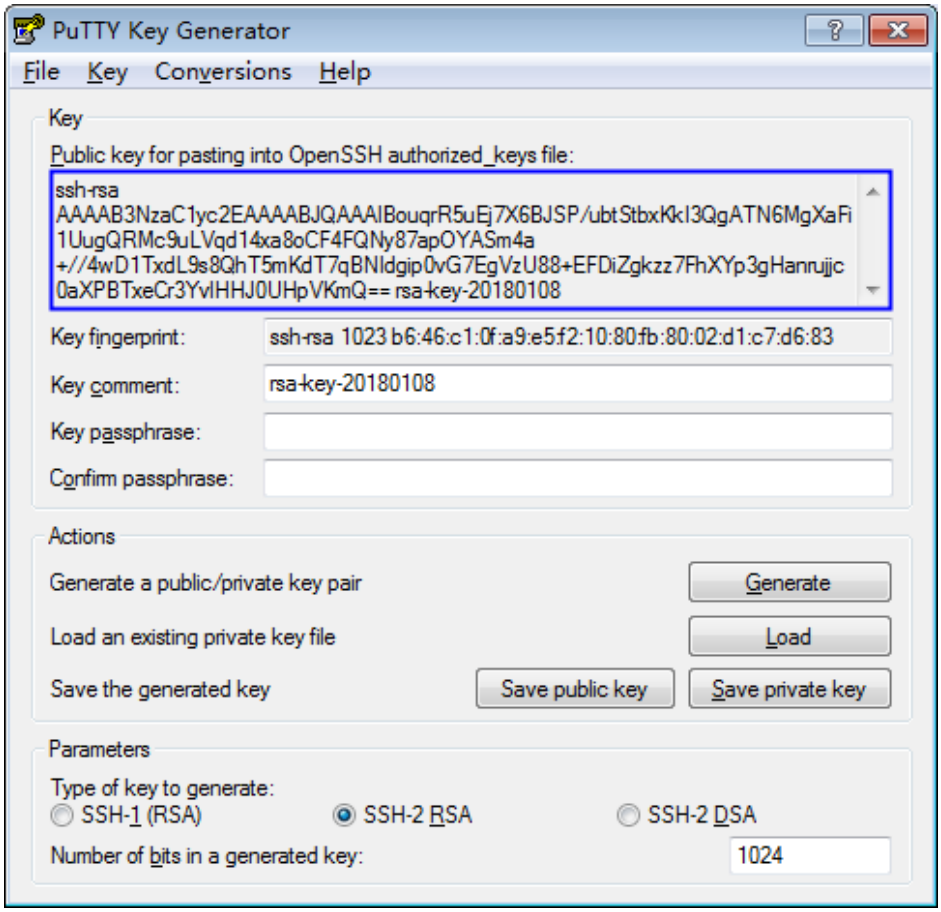
表 3-1 生成密钥对参数说明

参数	参数说明
Type of key to generate	当前导入管理控制台的密钥对的加解密算法，仅支持“SSH-2 RSA”。
Number of bits in a generated key	当前支持导入管理控制台的密钥对的算法长度为：1024、2048、4096。

步骤3 单击“Generate”，生成一个公钥和一个私钥，如图3-4所示。

蓝框中标记的内容为生成的公钥内容。

图 3-4 生成公钥和私钥文件



步骤4 复制蓝框中的公钥内容，并将其粘贴在文本文档中，以“.txt”格式保存在本地。

须知

请勿直接单击“Save public key”保存公钥文件。如果用户使用“Save public key”保存公钥，公钥内容的格式会发生变化，不能直接导入管理控制台使用。

步骤5 根据以下方式，选择保存私钥的格式，可保存为“.ppk”或者“.pem”格式的私钥。

须知

为保证安全，私钥只能下载一次，请妥善保管。

表 3-2 私钥文件格式

私钥文件格式	私钥使用场景	保存方法
“.pem”格式	• 使用**Xshell**工具登录Linux操作系统云服务器 • 将私钥托管在管理控制台	1. 选择“Conversions > Export OpenSSH key”。 2. 保存私钥到本地。例如：kp-123.pem。
	获取Windows操作系统云服务器的密码	1. 选择“Conversions > Export OpenSSH key”。 **说明** 请勿填写“Key passphrase”信息，否则会导致获取密码失败。 2. 保存私钥到本地。例如：kp-123.pem。
“.ppk”	使用**PuTTY**工具登录Linux操作系统云服务器	1. 在“PuTTY Key Generator”界面，选择“File > Save private key”。 2. 保存私钥到本地。例如：kp-123.ppk。

根据需要正确保存公钥和私钥文件后，可将密钥对导入管理控制台使用。

----结束

3.3 什么是私有密钥对和账号密钥对?

私有密钥对是仅支持当前账号查看或使用的密钥对。

账号密钥对是支持本账号下所有用户查看或使用的密钥对。

私有密钥对可以升级为账号密钥对，具体操作请参见**升级密钥对**。

3.4 导入通过 PuTTYgen 工具创建的密钥对失败如何处理?

问题描述

通过**PuTTYgen**工具创建的密钥对，在导入管理控制台使用时，系统提示导入公钥文件失败。

可能原因

公钥内容的格式不符合系统要求：

当用户使用**PuTTYgen**工具创建密钥对时，使用**PuTTYgen**工具的“Save public key”保存公钥，公钥内容的格式会发生变化。当用户将公钥内容导入管理控制台时，系统会校对公钥内容的格式，如果校对不成功，则会导致导入失败。

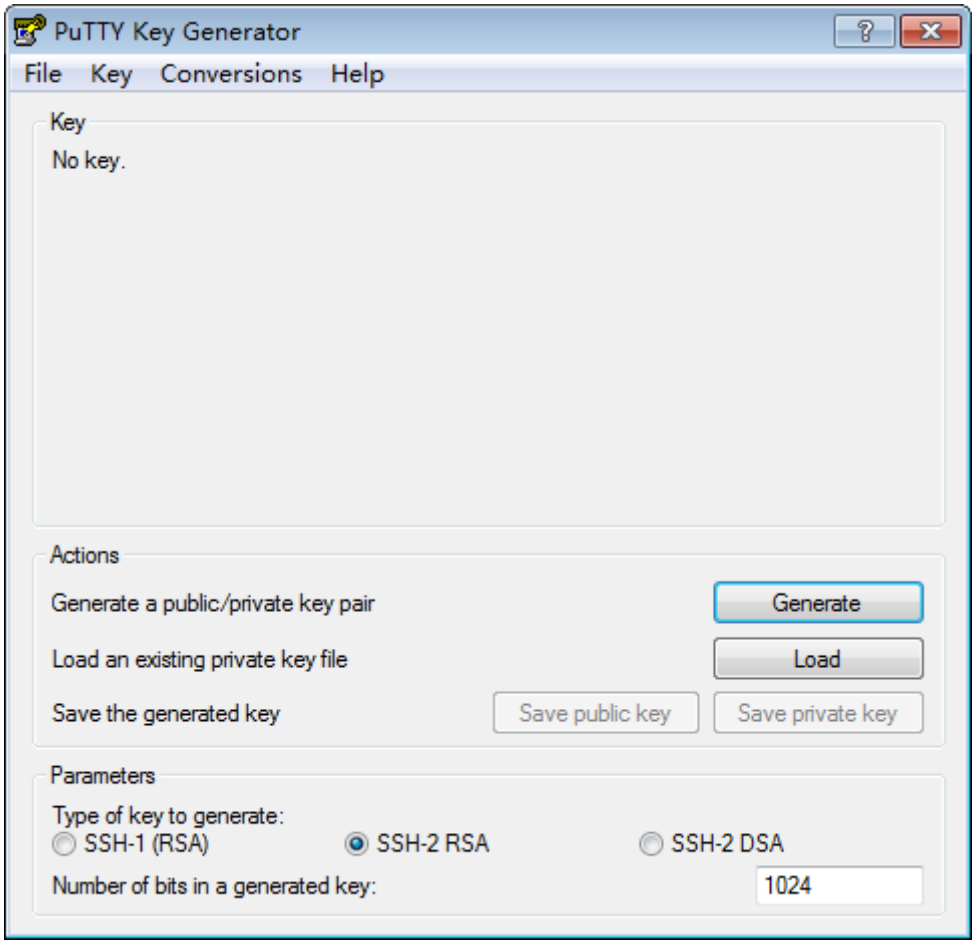
处理方法

使用本地保存的私钥文件，在“PuTTY Key Generator”中恢复内容格式正确的公钥文件，然后再将该公钥文件导入管理控制台。

步骤1 恢复内容格式正确的公钥文件。

1. 双击“PUTTYGEN.exe”，打开“PuTTY Key Generator”，如图3-5所示。

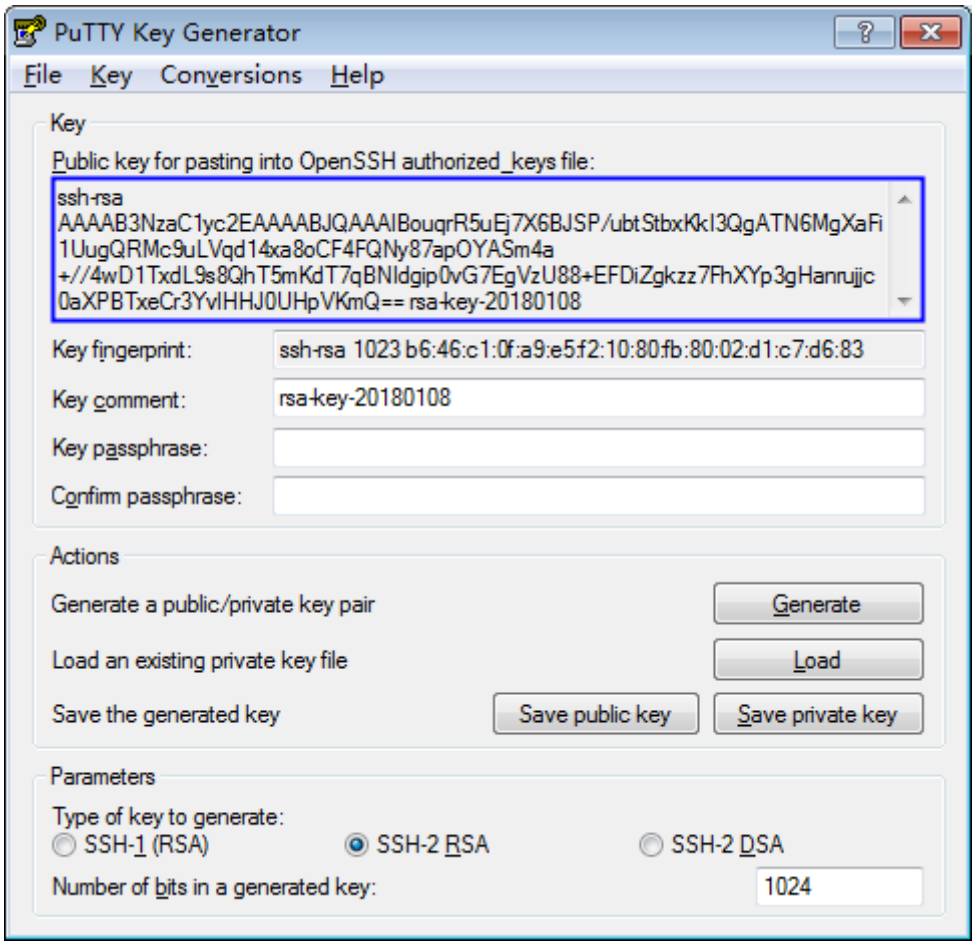
图 3-5 PuTTY Key Generator 主界面



2. 单击“Load”，并在本地选择该密钥对的私钥文件。

   系统将自动加载该私钥文件，并在“PuTTY Key Generator”中恢复格式正确的公钥文件内容，如图3-6所示，蓝框中的内容即为符合系统要求的公钥文件。

图 3-6 恢复公钥文件内容



3. 复制蓝框中的公钥内容，并将其粘贴在文本文档中，以“.txt”格式保存在本地。

须知

请勿直接单击“Save public key”保存公钥文件。如果用户使用“Save public key”保存公钥，公钥内容的格式会发生变化，不能直接导入管理控制台使用。

步骤2 将内容格式正确的公钥文件导入管理控制台。

1. 登录管理控制台。
2. 单击页面左上方的 ☰，选择“安全与合规 > 数据加密服务”。
3. 在左侧导航树中，选择“密钥对管理”。
4. 在密钥对列表页面，单击“导入密钥对”。
5. 单击“选择文件”，选择保存的“.txt”格式的公钥文件，或将公钥内容复制并粘贴至“公钥内容”文本框中。
6. 单击“确定”，导入公钥文件。

----结束

3.5 使用 IE9 浏览器无法导入密钥对如何处理?

问题描述

当使用的是IE9浏览器时，无法导入密钥对。

处理方法

步骤1 在浏览器主界面，单击  。

步骤2 选择“Internet选项”。

步骤3 在Inernet选项对话框中，单击“安全”。

步骤4 单击“Internet”。

步骤5 如果安全级别显示为“自定义”，单击“默认级别”，把设置还原为默认级别。

步骤6 滑动安全级别滑块，把安全级别调至“中”，单击“应用”。

步骤7 选择“自定义级别”。

步骤8 将“对未标记为可安全执行脚本的ActiveX控件初始化并执行脚本”设置为“提示”。

步骤9 单击“确定”。

----结束

3.6 如何使用私钥登录 Linux 弹性云服务器?

操作场景

用户通过管理控制台创建或者导入密钥对后，在购买弹性云服务器时，“登录方式”选择“密钥对”，并选择创建或者导入的密钥对。

用户购买弹性云服务器成功后，可使用密钥对的私钥登录弹性云服务器。

前提条件

- 使用的登录工具（如PuTTY、Xshell）与待登录的弹性云服务器之间网络连通。
- 弹性云服务器已经绑定弹性IP地址。
- 已获取该弹性云服务器的私钥文件。

本地使用 Windows 系统

如果您本地使用Windows操作系统登录Linux弹性云服务器，可以按照以下方式登录弹性云服务器。

方式一：使用PuTTY登录

以PuTTY为例介绍如何登录弹性云服务器，使用PuTTY登录弹性云服务器前，需要获取“.ppk”格式的私钥文件。

步骤1 双击“PuTTY.EXE”，打开“PuTTY Configuration”。

步骤2 选择“Connection > data”，在“Auto-login username”处输入镜像的用户名。

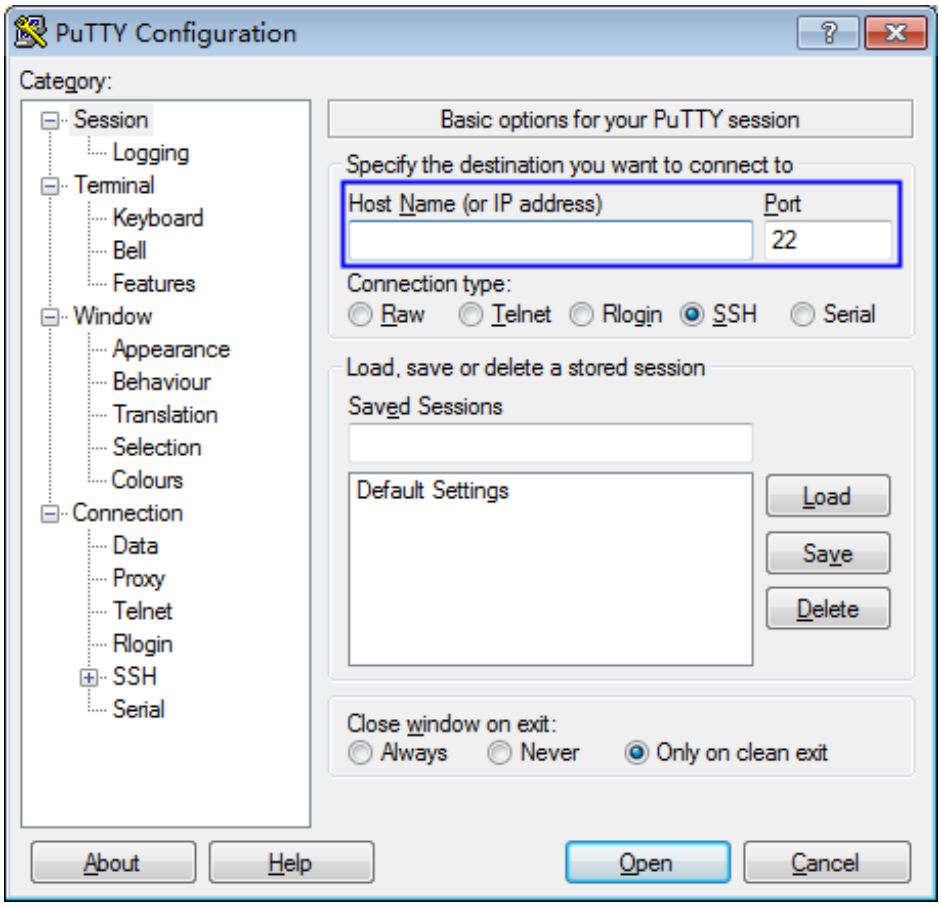
说明

- 如果是“CoreOS”的公共镜像，镜像的用户名为“core”。
- 如果是“非CoreOS”的公共镜像，镜像的用户名为“root”。

步骤3 选择“Connection > SSH > Auth”，在“Private key file forauthentication”配置项中，单击“Browse”，选择私钥文件（“.ppk”格式）。

步骤4 单击“Session”，在“Host Name (or IP address)”下的输入框中输入弹性云服务器的弹性IP地址。

图 3-7 配置弹性 IP



步骤5 单击“Open”，登录弹性云服务器。

----结束

方式二：使用Xshell登录

步骤1 打开Xshell工具。

步骤2 执行以下命令，SSH远程连接弹性云服务器。

ssh *用户名@弹性IP*

示例：

ssh *root@192.168.1.1*

步骤3 （可选）如果系统弹窗提示“SSH安全警告”，此时，需要单击“接受并保存”。

步骤4 选择“Public Key”，并单击“用户密钥(K)”栏的“浏览”。

步骤5 在“用户密钥”窗口中，单击“导入”。

步骤6 选择本地保存的私钥文件（“.pem”格式），并单击“打开”。

步骤7 单击“确定”，登录弹性云服务器。

----结束

本地使用 Linux 操作系统

如果您是在Linux操作系统上登录Linu弹性云服务器，可以按照下面方式登录。下面步骤以私钥文件是“kp-123.ppk”为例进行介绍。

步骤1 在您的Linux计算机的命令行中执行以下命令，变更权限。

chmod 600 ***/path/kp-123.ppk***

说明

*path*为密钥文件的存放路径。

步骤2 执行以下命令登录弹性云服务器。

ssh -i ***/path/kp-123*** **root@*弹性IP地址***

说明

- *path*为密钥文件的存放路径。
- *弹性IP地址*为弹性云服务器绑定的弹性IP地址。

----结束

3.7 如何通过私钥获取 Windows 弹性云服务器的登录密码?

操作场景

登录Windows操作系统的弹性云服务器时，需要使用密码方式登录。此时，用户需要先根据购买弹性云服务器时下载的私钥文件，获取该弹性云服务器初始安装时系统生成的管理员密码（Administrator账号或Cloudbase-init设置的账号）。该密码为随机密码，安全性高，请放心使用。

用户可以通过管理控制台获取Windows弹性云服务器的登录密码。

说明

- 为安全起见，建议用户获取初始密码后，执行清除密码操作，清除系统中记录的初始密码信息。

  该操作不会影响弹性云服务器的正常登录与运行。清除密码后，系统不能恢复获取密码功能，因此，请在执行清除密码操作前，记录弹性云服务器密码信息。详细信息请参见《弹性云服务器用户指南》。
- 用户也可以通过调用API接口的方式获取Windows弹性云服务器的初始密码，请参考《弹性云服务器API参考》。

前提条件

已获取登录弹性云服务器的私钥文件（“.pem”格式）。

获取密码

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角的，选择区域或项目。

步骤3 单击，选择“计算 > 弹性云服务器”。

步骤4 在弹性云服务器列表，选择待获取密码的弹性云服务器。

步骤5 选择“操作 > 更多”，单击“获取密码”。

步骤6 通过密钥文件获取密码，有以下两种方式：

- 单击“选择文件”，从本地上传密钥文件。
- 将密钥文件内容复制粘贴在空白文本框中。

步骤7 单击“获取密码”，获取随机密码。

----结束

3.8 绑定密钥对失败如何处理?

问题描述

当对弹性云服务器执行绑定密钥对操作时失败。

说明

- 管理控制台上“密钥对执行失败记录”对话框中的失败记录只记录了弹性云服务器的操作历史，不会影响弹性云服务器的状态及后续操作，可单击失败记录所在行的“删除”，直接删除失败记录，或者单击“删除所有失败记录”，删除所有执行失败的记录。
- 单击“了解更多”，查看相关文档。

可能原因

- 用户提供了错误或者失效的密码。
- 用户修改了公钥文件权限或属组。
- 用户修改了弹性云服务器的SSH配置。
- 弹性云服务器安全组22端口入方向未对100.125.0.0/16开放。
- 在弹性云服务器执行密钥对绑定期间，用户对弹性云服务器进行关机、开启或者卸载磁盘等操作。
- 网络发生故障。
- 弹性云服务器设置了防火墙规则。

处理方法

步骤1 查看弹性云服务器的状态。

- 运行中，请执行**步骤2**。
- 关机，请执行**步骤5**。

步骤2 使用密码登录弹性云服务器，检查密码是否正确。

- 正确，请执行**步骤4**。
- 错误，请使用正确的密码再次执行绑定密钥对操作。

步骤3 检查弹性云服务器的“/root/.ssh/authorized_keys”文件权限路径和属组是否被修改过。

- 是，请恢复至以下权限：
  - 每级属组都是root:root权限。
  - .ssh权限为700。
  - authorized_keys权限为600。
- 否，请执行**步骤4**。

步骤4 检查弹性云服务器的“/root/.ssh/authorized_keys”文件是否被修改过。

- 是，请根据实际情况恢复“/root/.ssh/authorized_keys”文件的原始内容。
- 否，请执行**步骤5**。

步骤5 检查ECS安全组22端口入方向是否对100.125.0.0/16开放，即允许100.125.0.0/16地址通过SSH远程连接到Linux弹性云服务器。

- 是，请执行**步骤6**。
- 否，请添加如下安全组规则后再次执行绑定密钥对操作。添加安全组规则具体操作请参见**添加安全组规则**。

方向	协议/应用	端口	源地址
入方向	SSH（22）	22	100.125.0.0/16

步骤6 请检查执行密钥对绑定操作的弹性云服务器是否可以正常开机、关机和登录使用等操作。

- 是，请再次执行绑定密钥对操作。
- 否，请执行**步骤7**。

步骤7 检查网络是否发生故障。

- 是，请联系技术支持工程师查看并定位原因。
- 否，请再次执行绑定密钥对操作。

----结束

3.9 替换密钥对失败如何处理?

问题描述

当对弹性云服务器执行替换密钥对操作时失败。

说明

管理控制台上“密钥对执行失败记录”对话框中的失败记录只记录了弹性云服务器的操作历史，不会影响弹性云服务器的状态及后续操作，可单击失败记录所在行的“删除”，直接删除失败记录，或者单击“删除所有失败记录”，删除所有执行失败的记录。

可能原因

- 用户提供了错误或者失效的私钥。
- ECS安全组22端口入方向未对100.125.0.0/16开放。
- 用户修改了服务器的SSH配置。
- 在弹性云服务器执行密钥对替换操作期间，用户对弹性云服务器进行关机、开启或者卸载磁盘等操作。
- 网络发生故障。
- 弹性云服务器设置了防火墙规则。

处理方法

步骤1 使用SSH密钥对登录弹性云服务器，检查私钥是否正确。

- 正确，请执行**步骤2**。
- 错误，请使用正确的私钥再次执行替换密钥对操作。

步骤2 检查弹性云服务器的“/root/.ssh/authorized_keys”文件是否被修改过。

- 是，请根据实际情况恢复“/root/.ssh/authorized_keys”文件的原始内容。
- 否，请执行**步骤3**。

步骤3 查看ECS安全组22端口入方向是否对100.125.0.0/16开放，即允许100.125.0.0/16地址通过SSH远程连接到Linux弹性云服务器。

- 是，请执行**步骤4**。
- 否，请添加如下安全组规则后再次执行替换密钥对操作。

方向	协议/应用	端口	源地址
入方向	SSH（22）	22	100.125.0.0/16

步骤4 请检查执行密钥对替换操作的弹性云服务器是否可以正常开机、关机和登录使用等操作。

- 是，请再次执行替换密钥对操作。
- 否，请执行**步骤5**。

步骤5 检查网络是否发生故障。

- 是，请联系技术支持工程师查看并定位原因。
- 否，请再次执行替换密钥对操作。

----结束

3.10 重置密钥对失败如何处理?

问题描述

当对弹性云服务器执行重置密钥对操作时失败。

说明

管理控制台上“密钥对执行失败记录”对话框中的失败记录只记录了弹性云服务器的操作历史，不会影响弹性云服务器的状态及后续操作，可单击失败记录所在行的“删除”，直接删除失败记录，或者单击“删除所有失败记录”，删除所有执行失败的记录。

可能原因

- ECS安全组22端口入方向未对100.125.0.0/16开放。
- 在弹性云服务器执行密钥对重置操作期间，用户对弹性云服务器进行关机、开启或者卸载磁盘等操作。
- 网络发生故障。
- 弹性云服务器设置了防火墙规则。

处理方法

步骤1 查看ECS安全组22端口入方向是否对100.125.0.0/16开放，即允许100.125.0.0/16地址通过SSH远程连接到Linux弹性云服务器。

- 是，请执行**步骤2**。
- 否，请添加如下安全组规则后再次执行重置密钥对操作。

方向	协议/应用	端口	源地址
入方向	SSH（22）	22	100.125.0.0/16

步骤2 请检查执行密钥对重置操作的弹性云服务器是否可以正常开机、关机和登录使用等操作。

- 是，请再次执行重置密钥对操作。
- 否，请执行**步骤3**。

步骤3 检查网络是否发生故障。

- 是，请联系技术支持工程师查看并定位原因。
- 否，请再次执行重置密钥对操作。

----结束

3.11 解绑密钥对失败如何处理?

问题描述

当对弹性云服务器执行解绑密钥对操作时失败。

说明

管理控制台上“密钥对执行失败记录”对话框中的失败记录只记录了弹性云服务器的操作历史，不会影响弹性云服务器的状态及后续操作，可单击失败记录所在行的“删除”，直接删除失败记录，或者单击“删除所有失败记录”，删除所有执行失败的记录。

可能原因

- 用户提供了错误或者失效的私钥。
- ECS安全组22端口入方向未对100.125.0.0/16开放。
- 用户修改了服务器的SSH配置。
- 在弹性云服务器执行密钥对解绑期间，用户对弹性云服务器进行关机、开启或者卸载磁盘等操作。
- 网络发生故障。
- 弹性云服务器设置了防火墙规则。

处理方法

步骤1 查看弹性云服务器的状态。

- 运行中，请执行**步骤2**。
- 关机，请执行**步骤4**。

步骤2 使用SSH密钥对登录弹性云服务器，检查私钥是否正确。

- 正确，请执行**步骤4**。
- 错误，请使用正确的私钥再次执行解绑密钥对操作。

步骤3 检查弹性云服务器的“/root/.ssh/authorized_keys”文件是否被修改过。

- 是，请恢复“/root/.ssh/authorized_keys”文件的原始内容。
- 否，请执行**步骤4**。

步骤4 查看ECS安全组22端口入方向是否对100.125.0.0/16开放，即允许100.125.0.0/16地址通过SSH远程连接到Linux弹性云服务器。

- 是，请执行**步骤5**。
- 否，请添加如下安全组规则后再次执行解绑密钥对操作。

方向	协议/应用	端口	源地址
入方向	SSH（22）	22	100.125.0.0/16

步骤5 请检查执行密钥对解绑操作的弹性云服务器是否可以正常开机、关机和登录使用等操作。

- 是，请再次执行解绑密钥对操作。
- 否，请执行**步骤6**。

步骤6 检查网络是否发生故障。

- 是，请联系技术支持工程师查看并定位原因。
- 否，请再次执行解绑密钥对操作。

----结束

3.12 替换密钥对后，服务器需要重启吗?

不需要重启，替换密钥对操作对业务无影响。

3.13 关闭弹性云服务器的密码登录方式后如何重新开启?

当用户将密钥对绑定到弹性云服务器时，关闭了密码登录方式，如果仍然需要使用密码登录弹性云服务器，可重新开启密码登录方式。

操作步骤

如下以PuTTY方式登录弹性云服务器开启密码登录方式为例进行说明。

步骤1 双击“PuTTY.EXE”，打开“PuTTY Configuration”。

步骤2 选择“Connection > data”，在“Auto-login username”处输入镜像的用户名。

说明

- 如果是“CoreOS”的公共镜像，镜像的用户名为“core”。
- 如果是“非CoreOS”的公共镜像，镜像的用户名为“root”。

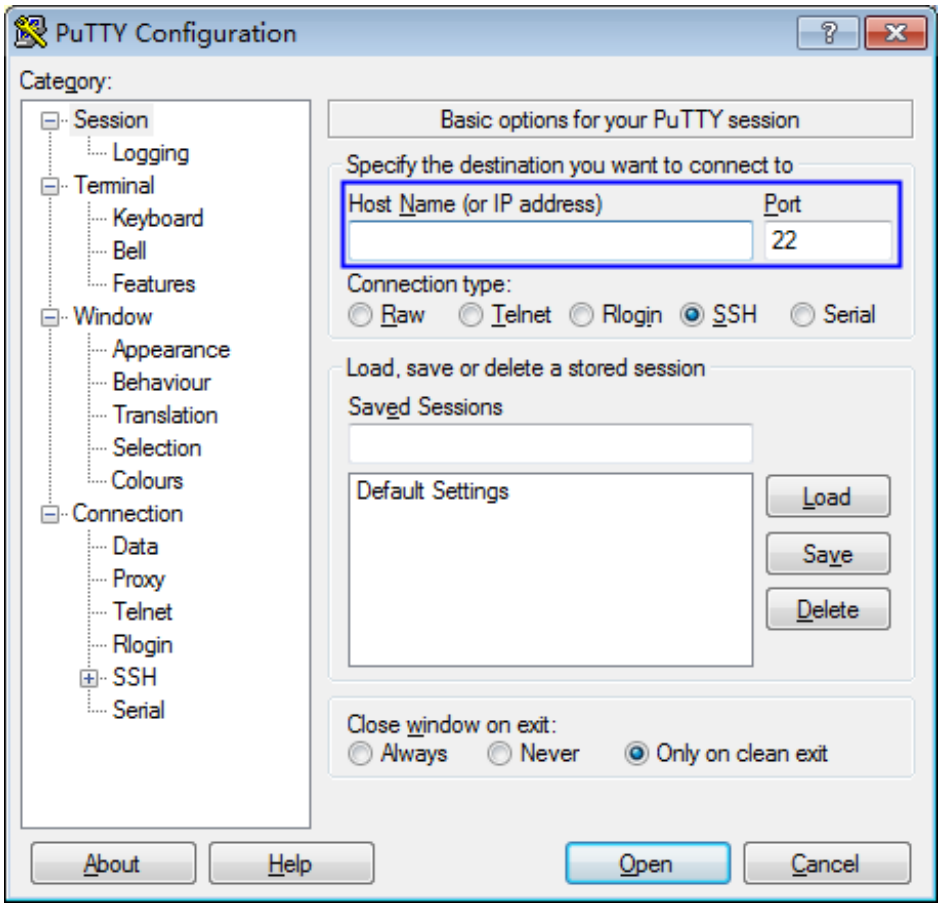
步骤3 选择“Connection > SSH > Auth”，在“Private key file forauthentication”配置项中，单击“Browse”，选择私钥文件（“.ppk”格式）。

说明

如果是“.pem”格式文件，请参考**将“.pem”格式的私钥文件转化为“.ppk”格式**进行转换。

步骤4 单击“Session”，在“Host Name (or IP address)”下的输入框中输入弹性云服务器的弹性IP地址。

图 3-8 配置弹性 IP



步骤5 单击“Open”，登录弹性云服务器。

步骤6 执行以下命令，打开“/etc/ssh/sshd_config”文件。

vi /etc/ssh/sshd_config

步骤7 按“i”进入编辑模式，开启密码方式登录。

- 非SUSE操作系统，将“PasswordAuthentication”字段值修改为“yes”。
  ```
  PasswordAuthentication yes
  ```
- SUSE操作系统，将“PasswordAuthentication”和“UsePAM”字段值修改为“yes”。
  ```
  PasswordAuthentication yes
  UsePAM yes
  ```

说明

- 非SUSE操作系统

  关闭密码方式登录需要将“PasswordAuthentication”字段值修改为“no”。如果“/etc/ssh/sshd_config”文件中没有“PasswordAuthentication”参数，新增该参数并配置为“no”。
- SUSE操作系统

  关闭密码登录需要将“PasswordAuthentication”和“UsePAM”字段值均修改为“no”。如果文件中没有“PasswordAuthentication”和“UsePAM”参数，新增该参数并配置为“no”。

步骤8 按“Esc”，退出编辑模式。

步骤9 输入“:wq”，按“Enter”，保存退出。

步骤10 执行以下命令，重启SSH服务，使配置生效。

- 非*Ubuntu14.xx*版本的操作系统。
  service sshd restart
- *Ubuntu14.xx*版本的操作系统。
  service ssh restart

----结束

3.14 解绑密钥对后用户无法登录 ECS 时如何处理?

问题描述

- 用户购买弹性云服务器时，选择的是“密钥对方式”登录弹性云服务器，解绑初始密钥对后，用户没有密码和密钥对，无法登录弹性云服务器
- 用户在KPS管理控制台给弹性云服务器绑定密钥对时，勾选了“关闭密码登录方式”，解绑密钥对后，用户没有密码和密钥对，无法登录弹性云服务器。

处理方法

方式一：重置密码

通过弹性云服务器界面重置密码，使用密码登录弹性云服务器，详细信息请参见《弹性云服务器用户指南》。

方式二：重置密钥对

将弹性云服务器关机，然后通过KPS管理控制台重新绑定密钥对，使用密钥对登录弹性云服务器，操作步骤如下：

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角的，选择区域或项目。

步骤3 单击页面左上方的，选择“安全与合规 > 数据加密服务”。

步骤4 单击“云服务器列表”，显示云服务器列表页面，如图3-9所示。

图 3-9 弹性云服务器列表

步骤5 单击目标弹性云服务器的名称，进入弹性云服务器详细信息界面。

步骤6 单击右上角“关机”，将弹性云服务器关机。

步骤7 参照**步骤5**，回到云服务器列表页面。

步骤8 单击目标弹性云服务器所在行的“绑定”，弹出绑定密钥对的对话框。

步骤9 在“新密钥对”下拉列表中，选择新的密钥对。

图 3-10 绑定密钥对

步骤10 用户可根据自己的需要选择是否勾选“关闭密码登录方式”，默认勾选“关闭密码登录方式”。

说明

- 如果不关闭密码登录方式，用户既可使用密码登录弹性云服务器，也可以使用密钥对登录弹性云服务器。
- 如果关闭了密码登录方式，用户只能使用密钥对登录弹性云服务器，如果用户仍然需要使用密码登录弹性云服务器，可再次开启密码登录方式，具体操作请参见**关闭弹性云服务器的密码登录方式后如何重新开启？**。

步骤11 请阅读并勾选“我已阅读并同意《密钥对管理服务免责声明》”。

步骤12 单击“确定”，完成密钥对绑定操作，绑定完成后，可使用密钥对登录弹性云服务器。

----结束

3.15 私钥不慎遗失怎么办？

私钥托管在 KPS

私钥托管在KPS，您可根据需要将私钥多次导出使用。

私钥未托管在 KPS

私钥未托管在KPS，私钥遗失后，将无法找回。

您可以通过重置密码或重置密钥对的方式，重新给弹性云服务器绑定密钥对，可参照**解绑密钥对后用户无法登录ECS时如何处理?**进行处理。

3.16 如何转换私钥文件格式?

将“.ppk”格式的私钥文件转化为“.pem”格式

上传或者拷贝至文本框的私钥必须是“.pem”格式文件，如果是“.ppk”格式文件，请执行以下步骤进行转换。

步骤1 在以下路径中下载PuTTY和PuTTYgen。

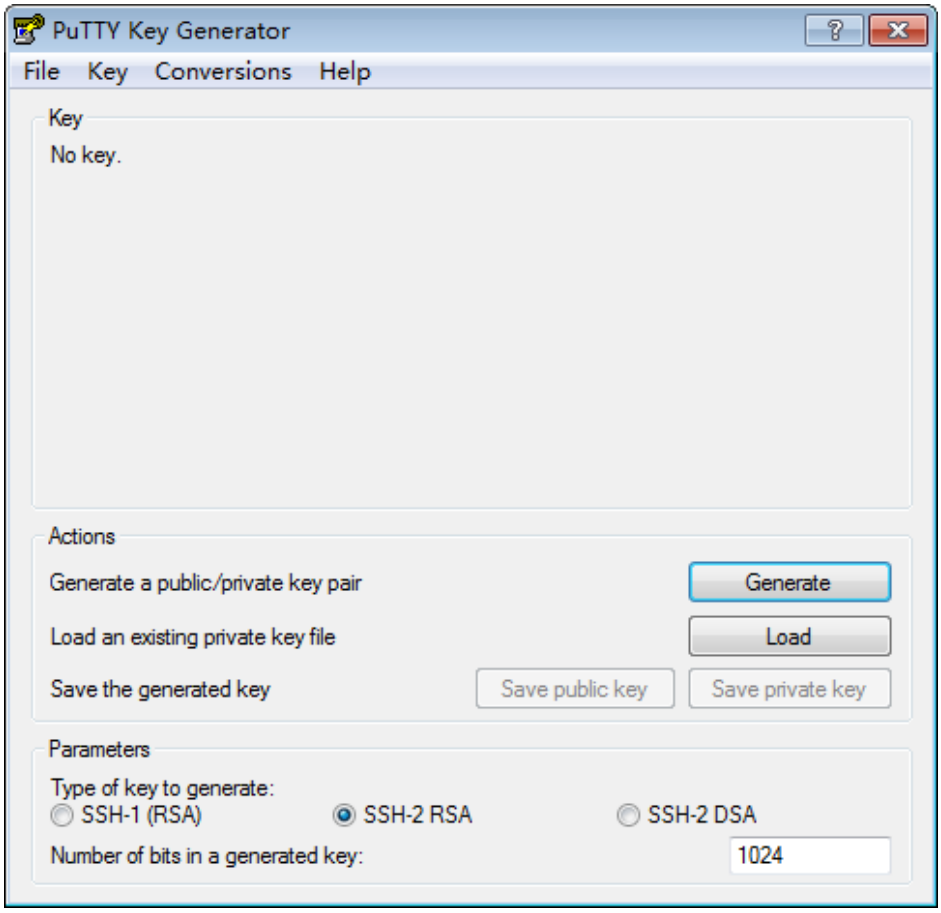
http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

说明

PuTTYgen是密钥生成器，用于创建SSH密钥对，生成一个公钥和私钥供PuTTY使用。

步骤2 双击“PUTTYGEN.exe”，打开“PuTTY Key Generator”，如图3-11所示。

图 3-11 PuTTY Key Generator



步骤3 选择“Conversions > Import Key”导入格式为“.ppk”的私钥文件。

步骤4 选择“Conversions > Export OpenSSH Key”，弹出“PuTTYgen Warning”对话框。

步骤5 单击“是”，将文件保存为“.pem”格式文件。

----结束

将“.pem”格式的私钥文件转化为“.ppk”格式

使用**PuTTY**工具登录Linux操作系统云服务器时，私钥必须是“.ppk”格式文件，如果是“.pem”格式文件，请执行以下步骤进行转换。

步骤1 在以下路径中下载PuTTY和PuTTYgen。

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

说明

PuTTYgen是密钥生成器，用于创建SSH密钥对，生成一个公钥和私钥供PuTTY使用。

步骤2 双击“PUTTYGEN.exe”，打开“PuTTY Key Generator”

步骤3 在“Actions”区域，单击“Load”，并导入购买弹性云服务器时保存的私钥文件。

导入时注意确保导入的格式要求为“All files(*.*)”。

步骤4 单击“Save private key”。

步骤5 保存转化后的私钥到本地。例如：kp-123.ppk。

----结束

3.17 密钥对在创建主机成功之后可以更改吗?

可以。

您可以根据需要对弹性云服务器绑定的密钥对进行解绑、重置、替换等操作，更多详细操作请参见**管理密钥对**。

3.18 密钥对是否支持多用户共享?

密钥对不支持跨账号共享，但您可以通过以下方法实现密钥对在同一账号下的IAM用户之间共享：

- 通过导入密钥对的方式实现共享。如果多个IAM用户需要使用相同的密钥对，您可以先通过其他工具（例如，**PuTTYgen**工具）创建密钥对，然后分别在IAM用户的资源中导入您创建的密钥对，具体操作请参见**导入密钥对**。
- 通过将密钥对升级为账号密钥对的方式实现共享。**通过管理控制台创建的密钥对**或者已导入到控制台的密钥对，您可以参考**升级密钥对**章节将已创建的密钥对升级为账号密钥对。

3.19 如何获取密钥对的私钥或公钥文件?

获取私钥文件

在**创建密钥对**时，浏览器自动执行下载任务，下载私钥文件。

- 如果您没有进行私钥托管，为保证安全，私钥只能下载这一次，请妥善保管。
- 如果您已授权华为云托管私钥，可根据需要将托管的私钥导出使用，具体操作请参见**导出私钥**。

获取公钥文件

- 通过管理控制台创建的密钥对，公钥自动保存在华为云中，可按F12刷新密钥对列表，查看密钥对列表返回值中的“public_key”字段，获取公钥。
- 通过**PuTTYgen**工具创建密钥对，公钥保存在用户本地，请自行在保存路径获取。

3.20 账号密钥首次创建、首次升级时系统报错如何处理?

首次创建账号密钥对

首次创建账号密钥对时，需要具有Tenant Administrator系统角色的用户完成一次账号密钥对创建。

首次升级账号密钥对

进行升级密钥对操作后，您选择的密钥对将会升级为账号密钥对，本账号下所有用户均能查看或使用该密钥对。密钥对名称如果与其他子用户私有密钥对重名，将无法升级。升级密钥对时，需要具有Tenant Administrator系统角色的用户至少执行一次升级，升级密钥对个数不限。

3.21 私有密钥对升级账号密钥对后，会占用账号密钥对配额吗?

会。

当私有密钥对升级为账号密钥对时，会占用账号密钥对配额。

3.22 用户联邦身份登录时，私有密钥对升级账号密钥对之后，为什么私有密钥对会不可见?

问题描述

用户认证联邦身份后，使用联邦身份登录场景，在私有密钥对页面进行升级账号密钥对操作后，私有密钥对列表内部分密钥对不可见。

可能原因

由于联邦登录账号的userid是虚拟ID，在升级场景无法获取，所以密钥对升级后，原私有密钥对列表内密钥对会存在不可见问题。

处理方法

在使用联邦账号前，先用主账号进行一次升级账号密钥对。

私有密钥对是根据账号的userid进行资源隔离，账号密钥对是domainid进行资源隔离。因此主账号、联邦账号、委托账号使用密钥对建议如下：

-	**主账号**	**联邦认证账号**	**委托账号**
私有密钥对	不推荐	禁止	禁止
账号密钥对	推荐	推荐	推荐

4 专属加密类

4.1 哪些区域提供专属加密服务?

以下区域提供专属加密服务。其他区域按需部署，由于涉及到第三方硬件采购部署，部署周期约2个月。

- 华北-北京一
- 华北-北京四
- 华东-上海一
- 华东-上海二
- 华南-广州

4.2 什么是专属加密?

专属加密（Dedicated Hardware Security Module，Dedicated HSM）是一种云上数据加密的服务，可处理加解密、签名、验签、产生密钥和密钥安全存储等操作。

Dedicated HSM为您提供经国家密码管理局检测认证的加密硬件，帮助您保护弹性云服务器上数据的安全性与完整性，满足FIPS 140-2安全要求。同时，您能够对专属加密实例生成的密钥进行安全可靠的管理，也能使用多种加密算法来对数据进行可靠的加解密运算。

4.3 如何获取身份识别卡（Ukey）?

购买专属加密实例后，需要使用身份识别卡（Ukey）来进行实例的管理。

- **标准版**：请在专属加密实例购买界面，通过提交工单的方式，反馈Ukey邮寄地址。专属加密服务专家会尽快将身份识别卡(USB key)邮寄给您。
- **铂金版**（国内）：
  - 购买铂金版（国内）专属加密实例成功后，华为云安全专家会联系您。您可以提供Ukey收件地址，华为云会通过您提供的地址将配套的Uke邮寄给您。

4.4 加密机是否支持明文通信?

加密机支持明文通信，也支持TLS单向和双向通信。为了安全起见，建议优选双向TLS通信。

4.5 专属加密如何保障密钥生成的安全性?

- 密钥是由用户自己远程创建，且创建过程需要仅用户持有的Ukey参与认证。
- 加密机的配置和内部密钥的准备，都必须要使用这一组Ukey作为鉴权凭证才能操作。

用户作为设备使用者完全控制密钥的产生、存储和访问授权，Dedicated HSM只负责监控和管理设备及其相关网络设施。

4.6 机房管理员是否有超级管理权限，在机房插入特权 Ukey 窃取信息?

机房管理员没有超级管理权限，Ukey是Dedicated HSM提供给您的身份识别卡，此卡仅购买专属加密实例的用户持有。

敏感数据（密钥）存储在国家规定的硬件加密卡中，即使加密机制造商也无法读取内部密钥信息。

4.7 专属加密采用的是什么云加密机?

专属加密采用的是符合国家密码局认证或FIPS 140-2第3级验证的硬件加密机，对高安全性要求的用户提供高性能专属加密服务，保障数据安全，规避风险。

4.8 专属加密是否支持切换密码机?

创建专属加密实例后，无论是否成功，均不支持切换密码机。如果您想切换密码机类型，则需要重新购买，具体操作请参见**购买专属加密实例**。

不同之处在于：

- 如果成功创建专属加密实例后，不支持切换密码机，也不支持退订。如果您想切换密码机类型，则需要重新购买。
- 如果创建专属加密实例失败，不支持切换密码机，但可申请退款。

  可以单击该专属加密实例所在行的“删除”，删除专属加密实例，并以工单的形式申请退款。

  然后，如果您想切换密码机类型，再重新购买。

⚠ 注意

切换加密机类型后，用户无法将原有的根密钥导入到新的加密机类型中。

4.9 专属加密的设备是哪个厂商的?

目前专属加密设备厂商包含“江南天安”和“三未信安”。

4.10 专属加密支持哪些接口?

专属加密提供与实体密码设备相同的功能与接口，方便向云端迁移，具体支持：PKCS#11接口，CSP接口，JCE接口，GM/T 0018-2012 SDF接口等。

更多详细内容请参见**专属加密版本说明**。

4.11 如何开通公网访问专属加密实例?

专属加密实例可以通过绑定弹性公网IP的方式实现公网访问。

前提条件

拥有可用于绑定专属加密实例的弹性公网IP。

说明

弹性公网IP的申请可参考**申请弹性公网IP**。

约束条件

- 专属加密实例绑定弹性公网IP后，存在公网攻击的风险，请谨慎使用。
- 弹性公网IP属于收费资源，请按需配置，如果不使用时请及时解绑，解绑操作参见**弹性公网IP与实例解绑**。解绑后如果不释放该弹性公网IP，华为云会收取IP保有费。同时，解绑弹性公网IP后，对于按带宽计费的EIP，会继续收取带宽费，具体可参见**为什么弹性公网IP已经解绑或者释放了，还在继续扣费?** 。

操作步骤

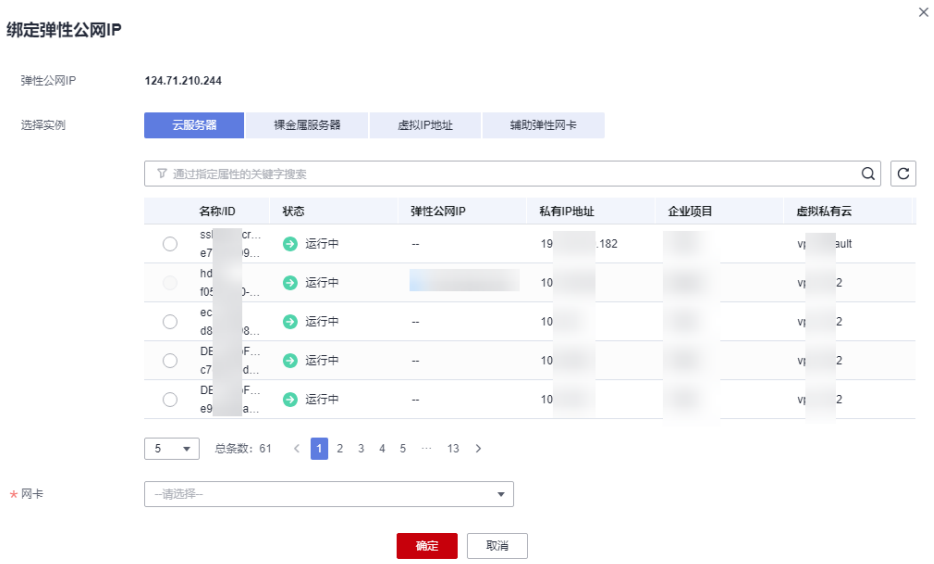
步骤1 **登录管理控制台**。

步骤2 单击管理控制台左上角，选择区域或项目。

步骤3 单击页面左侧，选择“网络 > 弹性公网IP”，默认进入“弹性公网IP”界面。

步骤4 选择已创建完成的目标弹性公网IP所在行，单击“绑定”，弹出“绑定弹性公网IP”页面，如**图 绑定弹性公网IP**所示。

图 4-1 绑定弹性公网 IP



步骤5 单击“虚拟IP地址”，在搜索栏中输入需绑定专属加密实例的IPV4地址，选择搜索结果进行绑定，如**图 虚拟IP地址绑定**所示。

图 4-2 虚拟 IP 地址绑定

步骤6 勾选对应IP地址后，单击“确定”，完成绑定操作。

----结束

5 计费类

5.1 数据加密服务如何收费和计费?

详细的服务资费和费率标准，请参见**产品价格详情**。

密钥管理

密钥管理实行按需计费，没有最低费用。用户创建密钥后，密钥会按小时计费。用户需要为自己创建的所有用户主密钥，以及超出免费次数的API请求支付费用。

密钥对管理

- 密钥对管理的私钥不托管在华为云时，密钥对管理免费使用。
- 私钥托管在华为云时，导入私钥成功后按照小时收费，当前阶段免费使用。

专属加密

专属加密根据您购买的专属加密实例版本和设备型号进行包年/包月收费。

凭据管理

根据您购买的凭据数量、使用时长和API请求次数进行收费。

5.2 如何为数据加密服务续费?

该任务指导用户如何在密钥管理或专属加密实例即将到期时进行续费。续费后，用户可以继续使用密钥管理或专属加密实例。

- 自动续费

  如果在升级密钥管理或购买专属加密实例时，您已勾选并同意“自动续费”，则在服务到期前，系统会自动按照购买周期生成续费订单并进行续费。
- 手动续费

  服务到期前，系统会以短信或邮件的形式提醒您服务即将到期，并提醒您续费。

  服务到期后，如果您没有及时续费，资源会进入保留期。

说明

服务到期后，如果没有按时续费，公有云平台会提供一定的保留期，保留期时长根据用户等级来定，具体请参见“**保留期**”。

表 5-1 到期说明

服务	版本	保留期
密钥管理	标准版	密钥被冻结。请通过充值的方式来激活被冻结的密钥。
专属加密	-	• 保留期内：无法使用专属加密实例，但资源予以保留。 • 保留期满：专属加密实例的资源将被释放。

说明

- 冻结状态的密钥无法用来执行加解密操作，为了防止造成不必要的损失，请您及时续费。
- 专属加密实例的资源释放后，您将失去与该实例相关的所有内容，为了防止造成不必要的损失，请您及时续费。

前提条件

已获取管理控制台的登录账号（拥有BSS Administrator权限与KMS Administrator权限）与密码。

说明

拥有BSS Administrator权限的账号，可以对账号中心、费用中心、资源中心中的所有菜单项执行任意操作。

操作步骤

步骤1 **登录管理控制台**。

步骤2 单击左上角的，选择区域或项目。

步骤3 在左侧导航树中，单击，选择“安全与合规 > 数据加密服务”。

步骤4 在界面右上角，单击“续费”。

步骤5 在“续费管理”界面，根据页面提示完成续费。

详细续费操作请参见**续费管理**。

----结束

5.3 如何退订数据加密服务?

数据加密服务不支持退订。

说明

如果您在使用专属加密时，创建专属加密实例失败，您可以单击创建失败的专属加密实例所在行的“删除”，删除专属加密实例，并以工单的形式申请退款。

相关链接

- 退订规则说明
- 不支持退订的云服务产品清单
- 如何提交工单

5.4 密钥被禁用后是否还计费?

计费。

密钥被禁用后，仍然会存储在KMS中，您可以根据需要随时启用该密钥。因此密钥被禁用后，仍然会计费。只有删除密钥，才会停止计费。

5.5 计划删除的凭据是否还计费?

不计费。

计划删除的凭据，从计划删除日期开始，直至凭据彻底被删除，凭据不会计费。

但是，如果您在凭据被彻底删除前的等待期内取消删除凭据，该凭据将恢复计费，并收取从计划删除开始到取消删除期间的费用。

5.6 计划删除的密钥是否还计费?

不计费。

计划删除的密钥，从计划删除日期开始，直至密钥彻底被删除，密钥不会计费。

但是，如果您在密钥被彻底删除前的等待期内取消删除密钥，该密钥将恢复计费，并收取从计划删除开始到取消删除期间的费用。

5.7 开通密钥轮转如何收费?

开通密钥轮转后，会收取相应的密钥存储费用，每个轮转的版本将作为一个独立的主密钥资源进行计算。

以1个密钥开通轮转，轮转周期为30天，单价0.015元（抹零后0.01元）每小时为例：

第一个月：密钥的轮转版本为0，收费为0.01*24*30 + 0*0.01*24*30 = 7.2元。

第二个月：密钥的轮转版本为1，收费为0.01*24*30 + 1*0.01*24*30 = 14.4元。

第三个月：密钥的轮转版本为2，收费为0.01*24*30 + 2*0.01*24*30 = 21.6元。

以此类推

第n个月：密钥的轮转版本为n，收费为0.01*24*30 + (n-1)*0.01*24*30 = 7.2*n元。

说明

调用次数计费与轮转次数计费无关。

5.8 副本密钥如何收费?

副本密钥与主密钥计费方式一致。副本密钥的账单发生在该密钥所在区域。

说明

如果您需要查看DEW服务的费用账单，您可以前往“费用中心 > 账单管理 > 流水和明细账单 > 费用账单”查看，详情可参考**费用账单**。

副本密钥的计费项为：密钥实例费用 * 时长+密钥请求API次数 * API请求费用。

6 通用类

须知

对接KMS时，必须有重试，包括不限于504、502、500、429等错误码，且推荐重试3~5次。针对502和504错误码，推荐超时时间5~8秒。不建议配置过长超时时间，否则会造成客户侧无法响应。

6.1 DEW 服务提供了哪些功能?

密钥管理

表 6-1 密钥管理

功能	服务内容
密钥全生命周期管理	• 创建、查看、启用、禁用、计划删除、取消删除自定义密钥 • 修改自定义密钥的别名和描述
用户自带密钥	导入密钥、删除密钥材料
小数据加解密	在线工具加解密小数据
签名验签	消息或消息摘要的签名、签名验证 **说明** 仅支持通过API调用。
密钥标签	添加、搜索、编辑、删除标签
密钥轮换	开启、修改、关闭密钥轮换周期
密钥授权	创建、撤销、查询授权
	退役授权 **说明** 仅支持通过API调用。

功能	服务内容
密钥区域性	跨区域创建副本密钥
云服务加密	对象存储服务OBS加密
	云硬盘服务EVS加密
	镜像服务IMS加密
	弹性文件服务SFS加密（SFS文件系统加密）
	弹性文件服务SFS加密（SFS Turbo文件系统加密）
	云数据库RDS（MySQL、PostgreSQL、SQL Server引擎）加密
	文档数据库服务DDS加密
	数据仓库服务DWS加密
数据加密密钥管理	创建、加密、解密数据加密密钥 **说明** 仅支持通过API调用。
生成硬件真随机数	生成512bit的随机数，为加密系统提供基于硬件真随机数的密钥材料和加密参数 **说明** 仅支持通过API调用。
消息认证码	生成、验证消息认证码 **说明** 仅支持通过API调用。
密钥库管理	创建、禁用、删除密钥库

密钥对管理

用户可通过密钥对管理界面或接口，对密钥对进行以下操作：

- 创建、导入、查看、删除密钥对
- 重置、替换、绑定、解绑密钥对
- 托管、导入、导出、清除私钥

专属加密

用户可通过专属加密界面，购买专属加密实例、实例化专属加密实例和查看专属加密实例信息。

6.2 DEW 采用的是什么加解密算法?

KPS 支持的密码算法

- 通过管理控制台创建的SSH密钥对支持的加解密算法为：
  - SSH-ED25519
  - ECDSA-SHA2-NISTP256
  - ECDSA-SHA2-NISTP384
  - ECDSA-SHA2-NISTP521
  - SSH_RSA有效长度为：2048，3072，4096
- 通过外部导入的SSH密钥对支持的加解密算法为：
  - SSH-DSS
  - SSH-ED25519
  - ECDSA-SHA2-NISTP256
  - ECDSA-SHA2-NISTP384
  - ECDSA-SHA2-NISTP521
  - SSH_RSA有效长度为：2048，3072，4096

Dedicated HSM 支持的密码算法

支持国密算法以及部分国际通用密码算法，满足用户各种加密算法需求。

表 6-2 Dedicated HSM 支持的密码算法

加密算法分类	通用密码算法	国密算法
对称密码算法	AES	SM1、SM4、SM7
非对称密码算法	RSA（1024-4096）	SM2
摘要算法	SHA1、SHA256、SHA384	SM3

6.3 什么是配额?

什么是配额?

为防止资源滥用，平台限定了各服务资源的配额，对用户的资源数量和容量做了限制。如您最多可以创建多少个用户主密钥。

如果当前资源配额限制无法满足使用需要，您可以申请扩大配额。

怎样查看我的配额?

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角，选择区域或项目。

步骤3 在页面右上角，选择“资源 > 我的配额”。

系统进入“服务配额”页面。

图 6-1 我的配额

步骤4 您可以在“服务配额”页面，查看各项资源的总配额、及使用情况。

步骤5 如果当前配额不能满足业务要求，请单击“申请扩大配额”。

----结束

如何申请扩大配额?

步骤1 登录管理控制台。

步骤2 单击管理控制台左上角，选择区域或项目。

步骤3 在页面右上角，选择“资源 > 我的配额”。

系统进入“服务配额”页面。

图 6-2 我的配额

步骤4 单击“申请扩大配额”。

步骤5 在“新建工单”页面，根据您的需求，填写相关参数。

其中，“问题描述”请填写需要调整的内容和申请原因。

步骤6 填写完毕后，勾选协议并单击“提交”。

----结束

6.4 DEW 服务资源分配的机制是什么?

DEW服务资源以region为大资源池，以每个客户单独的资源或服务为小资源池，后台有默认流量限制。对单个用户来说如果流量过大超过阈值会造成服务速度缓慢限制。对于有相应大流量需求的客户而言，可根据实际情况和需求进行后台资源变更。

如果客户业务量确实较大超出限制，可以通过提工单增加配额，DEW服务通过后台对客户限制进行调整，可为客户开通专属配置集群提供支撑，保证业务平稳运行。

6.5 什么是区域和可用区?

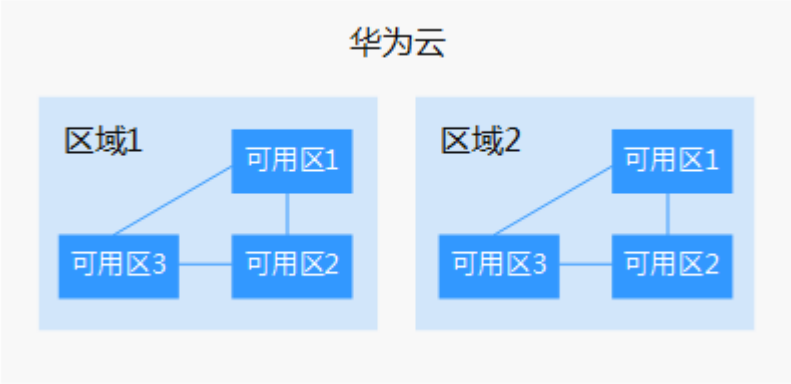
什么是区域、可用区?

通常使用区域和可用区来描述数据中心的位置，您可以在特定的区域、可用区创建资源。

- 区域（Region）：从地理位置和网络时延维度划分，同一个Region内共享弹性计算、块存储、对象存储、VPC网络、弹性公网IP、镜像等公共服务。Region分为通用Region和专属Region，通用Region指面向公共租户提供通用云服务的Region；专属Region指只承载同一类业务或只面向特定租户提供业务服务的专用Region。
- 可用区（AZ，Availability Zone）：一个AZ是一个或多个物理数据中心的集合，有独立的风火水电，AZ内逻辑上再将计算、网络、存储等资源划分成多个集群。一个Region中的多个AZ间通过高速光纤相连，以满足用户跨AZ构建高可用性系统的需求。

图6-3阐明了区域和可用区之间的关系。

图 6-3 区域和可用区



目前，华为云已在全球多个地域开放云服务，您可以根据需求选择适合自己的区域和可用区。

如何选择区域?

选择区域时，您需要考虑以下几个因素：

- 地理位置

  一般情况下，建议就近选择靠近您或者您的目标用户的区域，这样可以减少网络时延，提高访问速度。

  - 在除中国大陆以外的亚太地区有业务的用户，可以选择“中国-香港”、“亚太-曼谷”或“亚太-新加坡”区域。
  - 在非洲地区有业务的用户，可以选择“非洲-约翰内斯堡”区域。
  - 在拉丁美洲地区有业务的用户，可以选择“拉美-圣地亚哥”区域。

- 资源的价格

  不同区域的资源价格可能有差异，请参见**华为云服务价格详情**。

如何选择可用区?

是否将资源放在同一可用区内，主要取决于您对容灾能力和网络时延的要求。

- 如果您的应用需要较高的容灾能力，建议您将资源部署在同一区域的不同可用区内。
- 如果您的应用要求实例之间的网络延时较低，则建议您将资源创建在同一可用区内。

区域和终端节点

当您通过API使用资源时，您必须指定其区域终端节点。有关华为云的区域和终端节点的更多信息，请参阅**地区和终端节点**。

6.6 数据加密服务是否可跨账号使用?

数据加密服务暂不支持跨账号使用，每个用户只能使用并管理自己的密钥、密钥对。

6.7 数据加密服务支持通过哪些方式进行使用?

DEW提供了Web控制台管理方式和基于HTTPS请求的API（Application Programming Interface）管理方式。

- 管理控制台方式

  如果用户已注册公有云，可直接登录管理控制台，单击管理控制台左上角的，选择区域或项目后，单击页面左侧的，选择“安全与合规 > 数据加密服务”。

- API方式

  用户可通过接口方式访问数据加密服务，具体操作请参见《数据加密服务API参考》。

  数据加密服务提供了REST（Representational State Transfer）风格API，支持通过HTTPS请求调用。用户可使用提供的API对密钥和密钥对进行相关操作，如创建、查询、删除密钥等。

  在通过API调用数据加密服务时，API接口使用的是HTTPS协议，HTTPS为加密传输，可以保证传输通道的安全，不受中间人攻击。

6.8 为什么配置了数据加密服务的权限没有立即生效?

正常情况下，客户配置了数据加密服务相关的权限，如KMS Administrator， KMS CMKFullAccess等，权限是会立即生效的。

但如下情况会造成权限无法立即生效：

1. 控制台Console未及时登出，会有Session缓存造成权限不生效。建议重新登录控制台。
2. 客户使用对象存储服务OBS等具有权限缓存的服务。权限具体生效时间取决于对应服务权限缓存时长。
3. 客户编程调用APIG网关，权限生效时间取决于统一身份认证服务IAM广播权限变化到网关的时间。

