**AWS动手实验Lambad for SSM**

**实验目的**

使用Lambda上传一个参数

**实验要求**

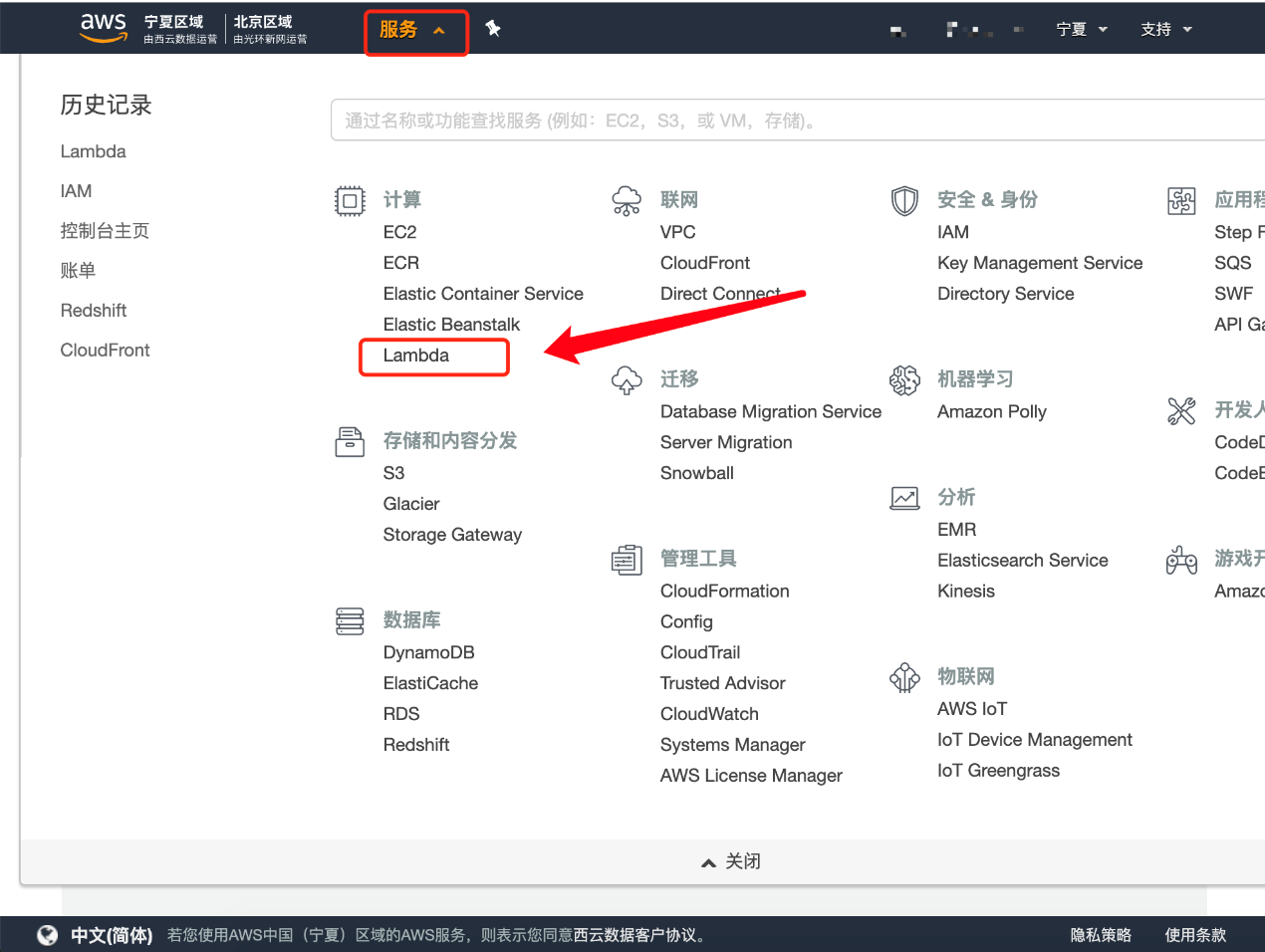
了解Amazon Web Services  
了解无服务器架构  
了解Python  
了解AWS SDK for Python  
准备一个拥有管理员权限的Lambda角色

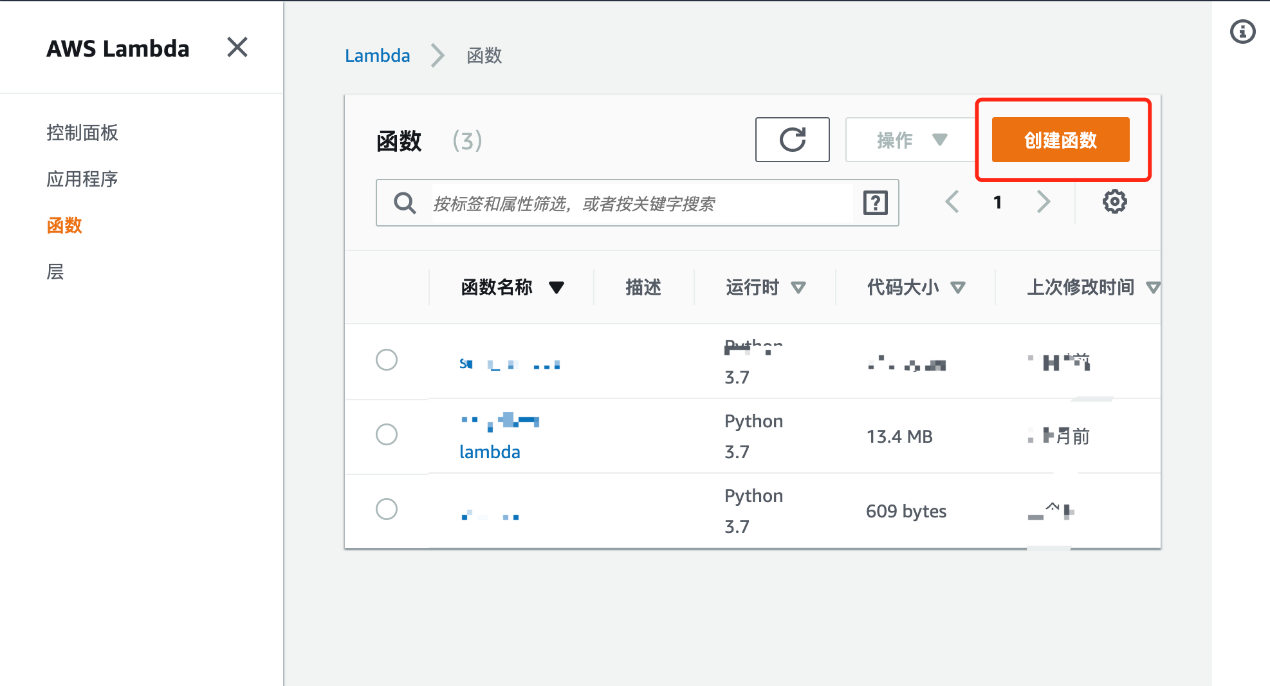
**实验原理**

AWS Lambda是一种计算服务，可让您在不配置或管理服务器的情况下运行代码。AWS Lambda仅在需要时执行您的代码并自动扩展，从每天几个请求到每秒数千个。您只需要为您消耗的计算时间付费，当您的代码未运行时不收取任何费用。借助AWS Lambda，您几乎可以为任何类型的应用程序或后端服务运行代码，并且不必进行任何管理。

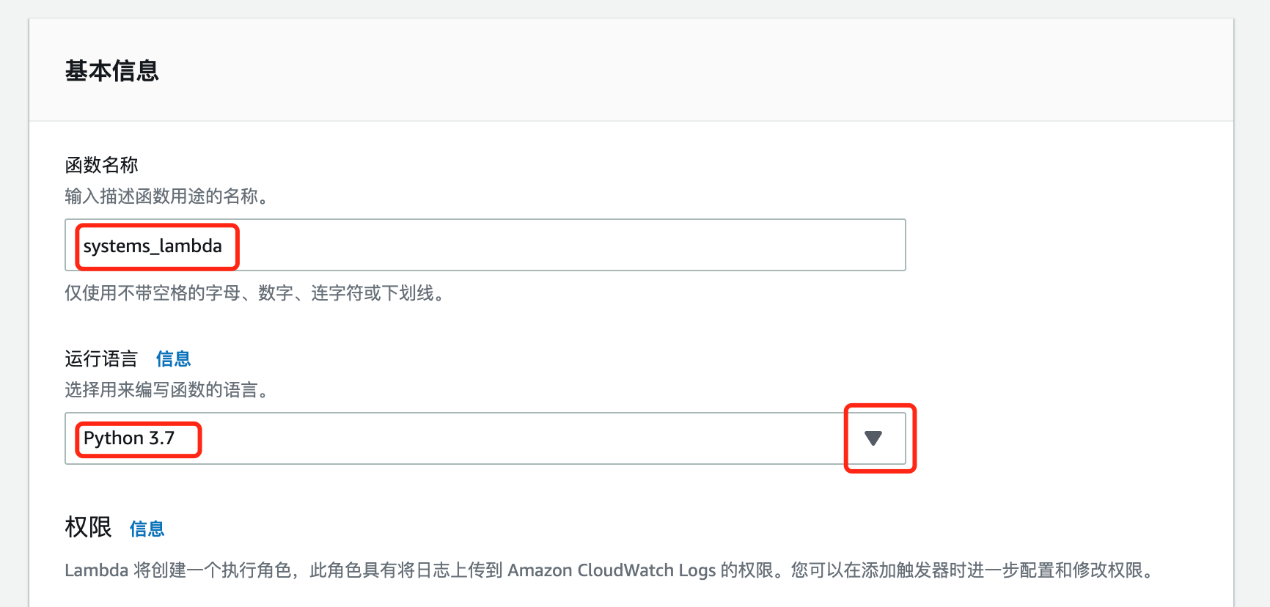
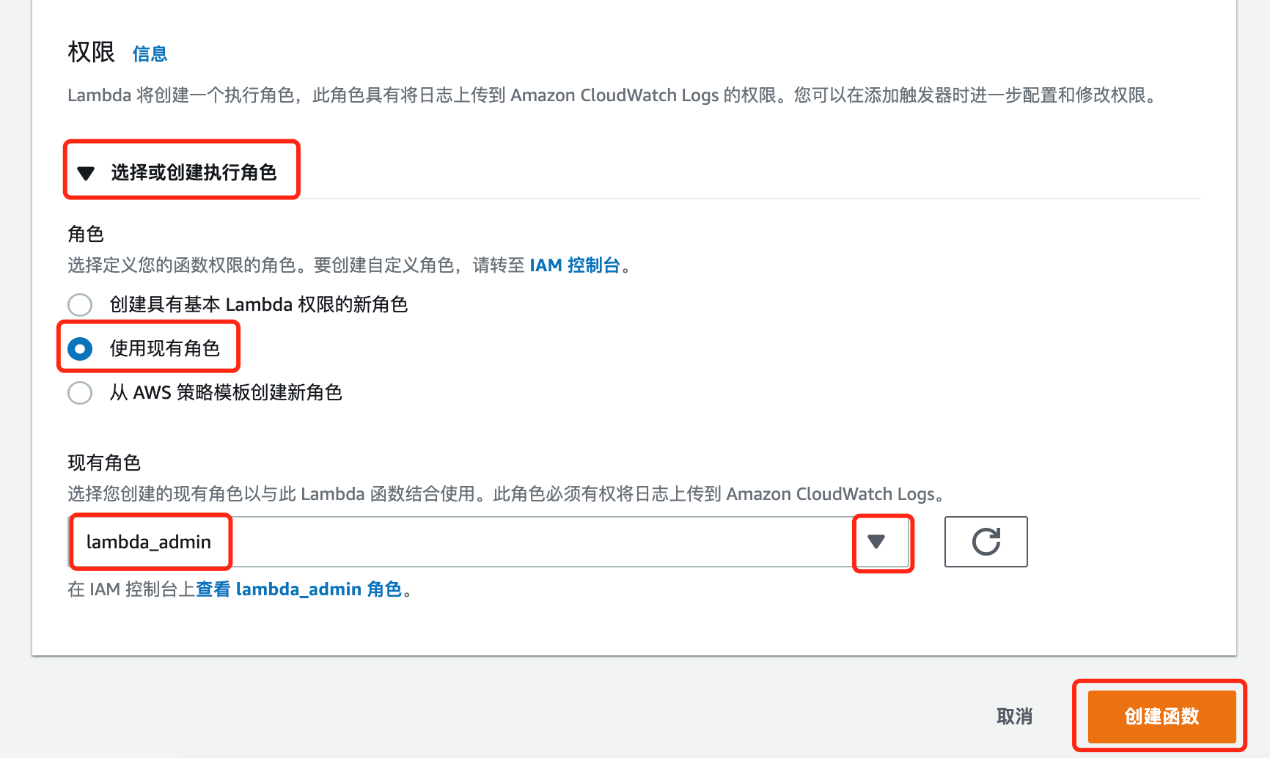
AWS systems manager是一个功能集合，可以帮助您自动管理任务，例如收集系统资源清单、应用操作系统（OS）补丁、自动创建Amazon Machine Image （AMI）以及大规模配置操作系统和应用程序。  
本实验使用Lambda为系统添加一个参数（一些开发场景可能使用）。

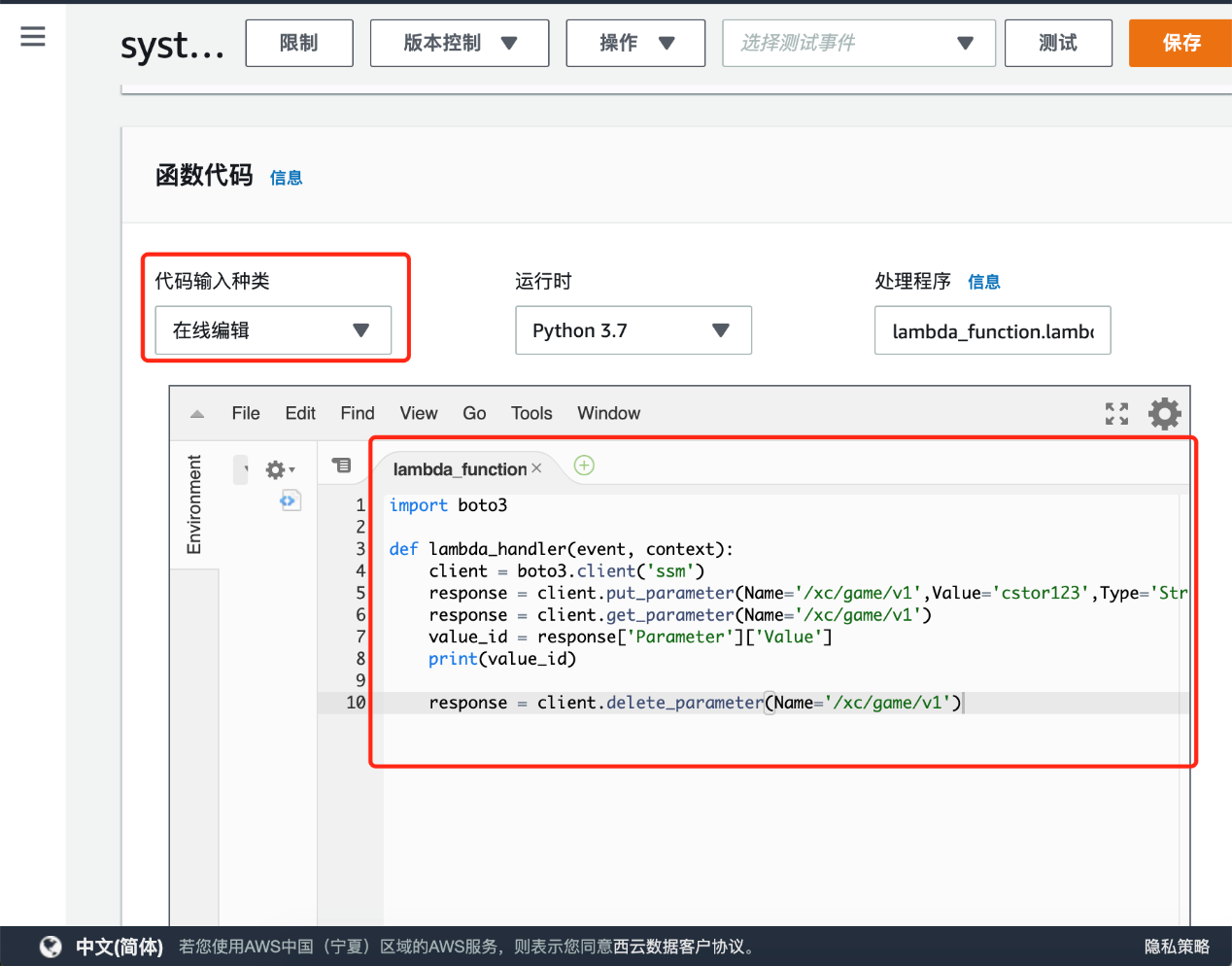
**实验步骤**

1、点击【服务】，选择【Lambda】，如下：  


2、在Lambda左侧导航窗格中点击【函数】后再点击【创建函数】，如下：  


3、在“创建函数”页，选择【从头开始制作】，如下：  


4、在下方“基本信息”中配置以下信息：  
        函数名称：【systems\_lambda】  
        运行语言：【Python3.7】  
        权限：点击【选择或创建执行角色】  
        执行角色：【使用现有角色】  
        角色名称：”ROLE\_NAME” \*\*\*注意：此角色拥有管理员的权限。  
完成以上操作后，点击【创建函数】，如下：  
  


5、在“函数代码”模块，编写代码。图中代码是上传一个参数，然后参数值打印出来，如下：  


6、代码如下：

import boto3

def lambda\_handler(event, context):

client = boto3.client('ssm')

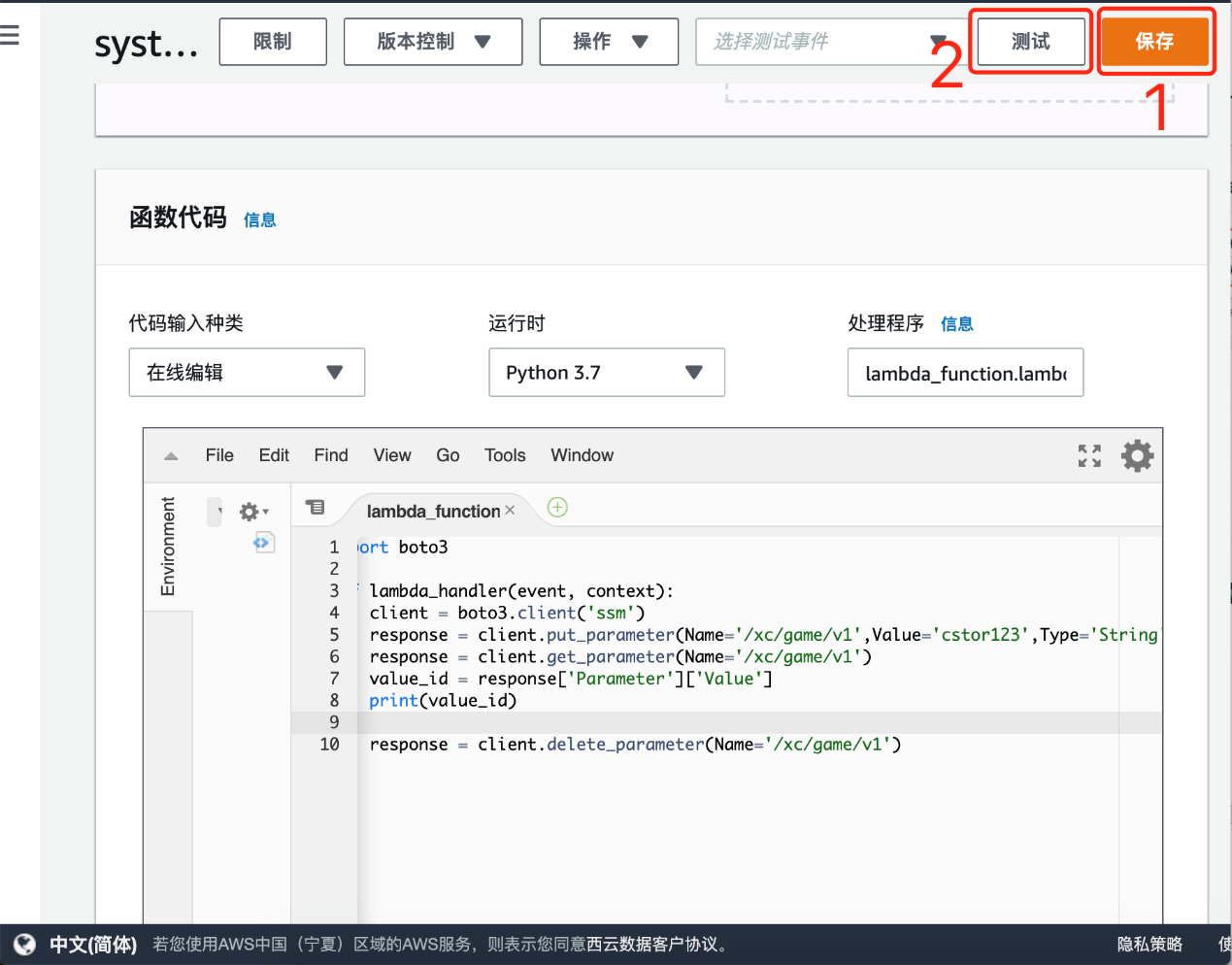
response = client.put\_parameter(Name='/xc/game/v1',Value='cstor123',Type='String')

response = client.get\_parameter(Name='/xc/game/v1')

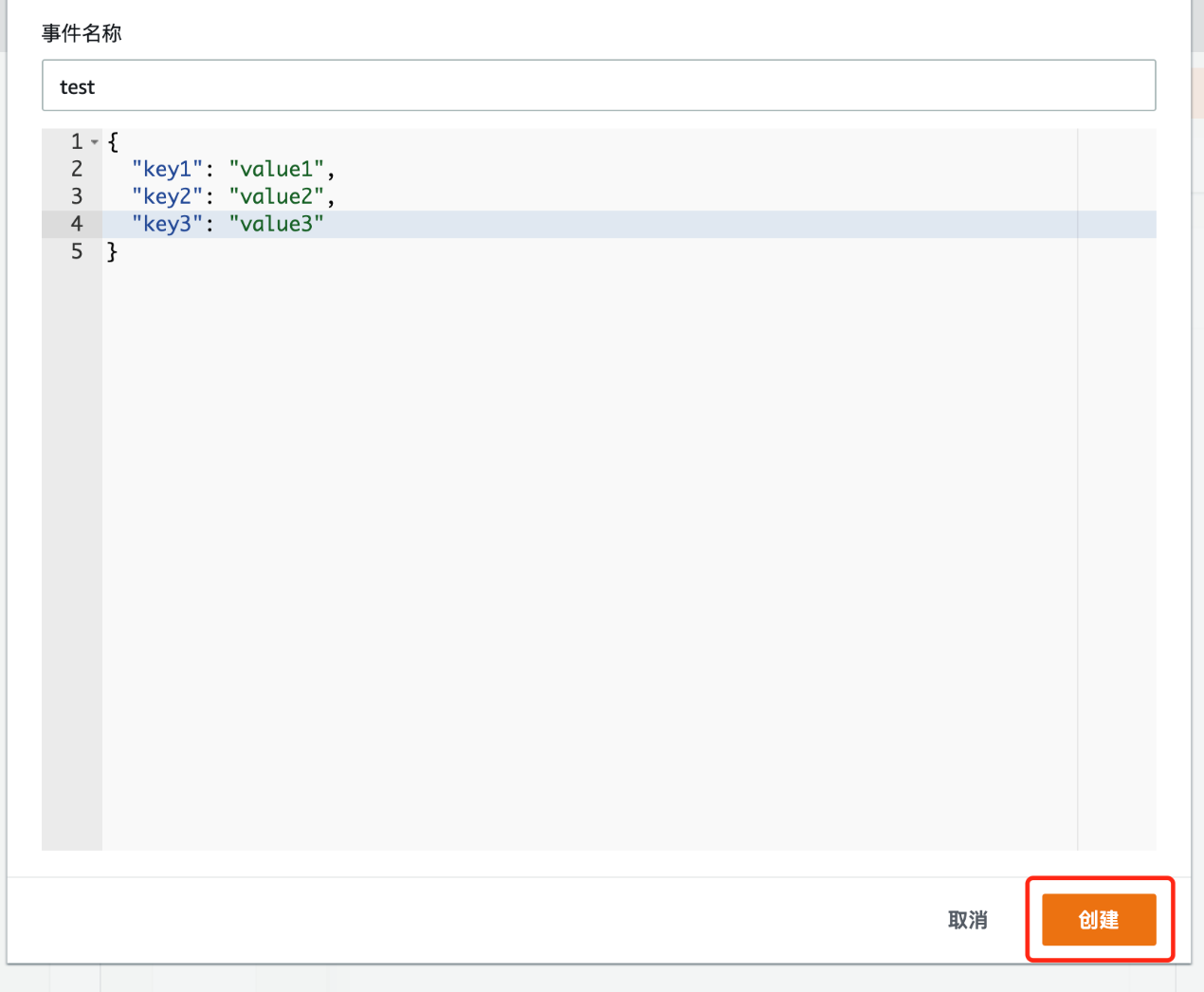
value\_id = response['Parameter']['Value']

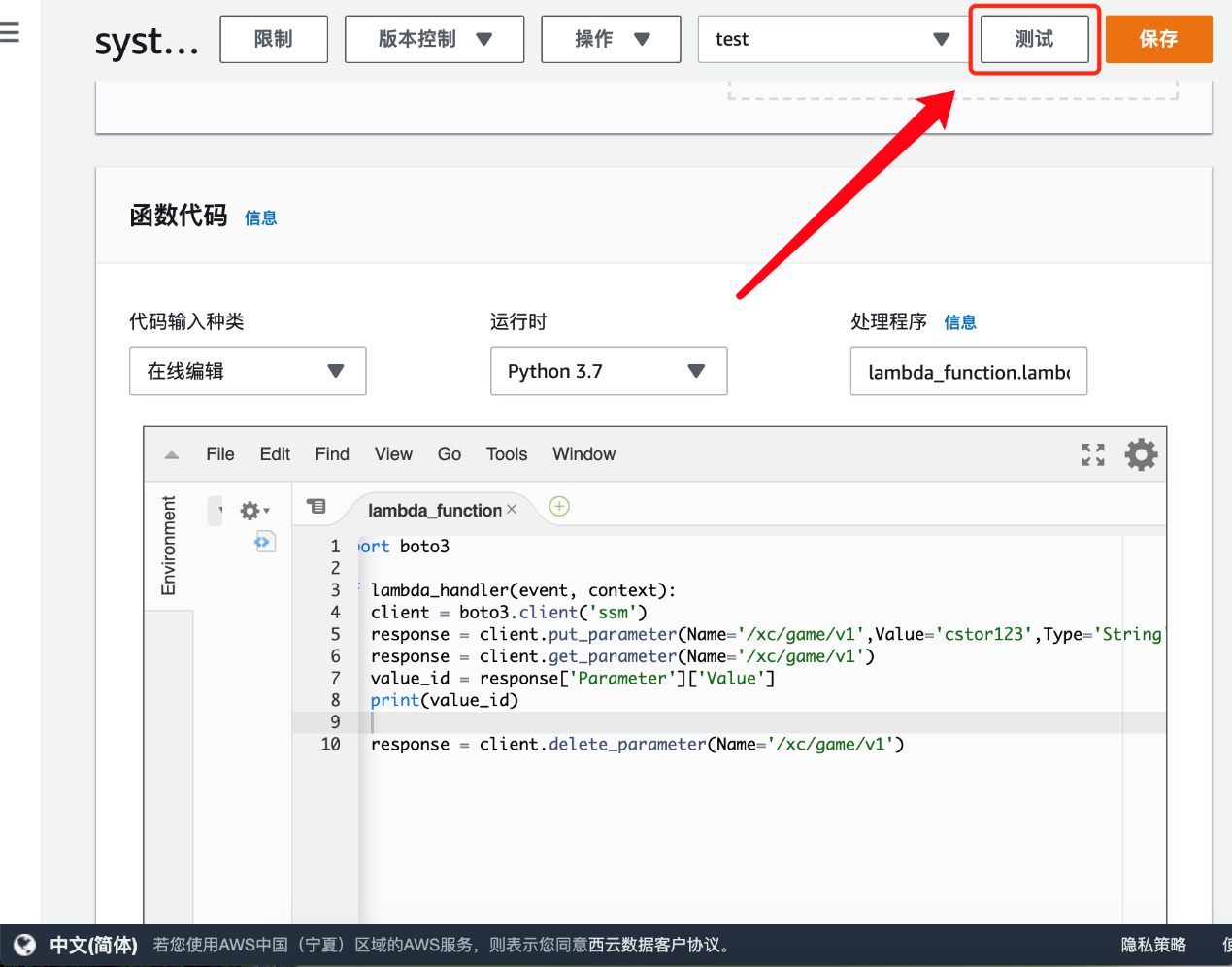
print(value\_id)

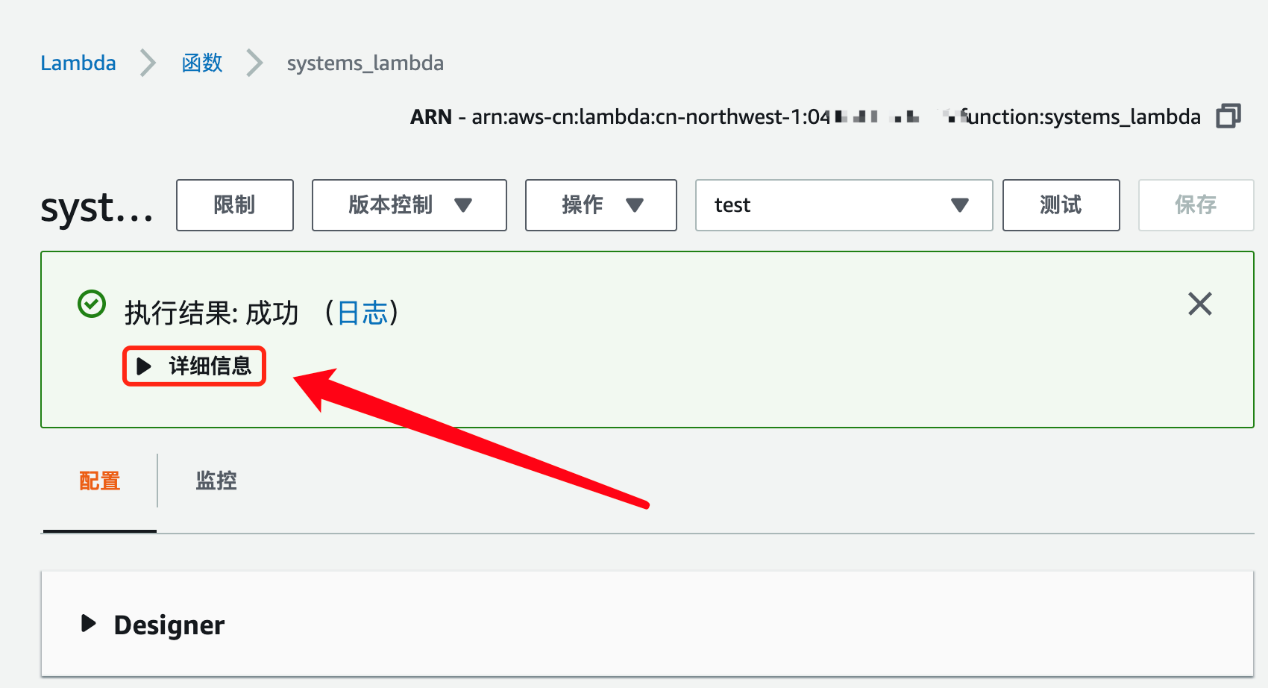
response = client.delete\_parameter(Name='/xc/game/v1')

7、编写完成代码后，点击上方菜单栏中的【保存】，然后点击【测试】，如下：  


8、在“配置测试事件”弹窗页中，配置事件的名称【test】，如下：  


9、下拉菜单栏，点击【创建】，如下：  


10、创建完成后，点击【测试】，如下：  


11、显示执行成功，点击【详细信息】以查看详细信息，如下：  


12、在日志输出中，可以看到我们上传到参数的值，如下：  


**实验结果**

Lambda执行成功  
