lightsail

以下是一个授予管理 Lightsail 实例所需权限的策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "lightsail:CreateRelationalDatabaseSnapshot",
                "lightsail:GetRelationalDatabaseEvents",
                "lightsail:CreateContainerService",
                "lightsail:GetKeyPair",
                "lightsail:GetContactMethods",
                "lightsail:GetCloudFormationStackRecords",
                "lightsail:GetContainerServiceDeployments",
                "lightsail:GetBucketAccessKeys",
                "lightsail:CreateContainerServiceRegistryLogin",
                "lightsail:GetContainerImages",
                "lightsail:CreateRelationalDatabase",
                "lightsail:CreateContactMethod",
                "lightsail:CreateDistribution",
                "lightsail:GetDomain",
                "lightsail:GetBuckets",
                "lightsail:GetRelationalDatabaseParameters",
                "lightsail:GetInstanceState",
                "lightsail:GetOperationsForResource",
                "lightsail:AllocateStaticIp",
                "lightsail:GetInstances",
                "lightsail:GetRelationalDatabase",
                "lightsail:CreateLoadBalancer",
                "lightsail:GetDistributionLatestCacheReset",
                "lightsail:GetLoadBalancerTlsPolicies",
                "lightsail:GetLoadBalancers",
                "lightsail:GetExportSnapshotRecords",
                "lightsail:GetAutoSnapshots",
                "lightsail:GetStaticIp",
                "lightsail:GetRelationalDatabaseBundles",
```

```
"lightsail:GetRelationalDatabaseBlueprints",
"lightsail:CreateInstances",
"lightsail:GetRelationalDatabaseLogEvents",
"lightsail:GetContainerServices",
"lightsail:GetRelationalDatabaseSnapshot",
"lightsail:GetInstancePortStates",
"lightsail:DeleteContactMethod",
"lightsail:GetContainerServicePowers",
"lightsail:GetKeyPairs",
"lightsail:GetLoadBalancer",
"lightsail:DisableAddOn",
"lightsail:CreateCloudFormationStack",
"lightsail:GetRelationalDatabaseSnapshots",
"lightsail:UnpeerVpc",
"lightsail:GetLoadBalancerTlsCertificates",
"lightsail:GetAlarms",
"lightsail:GetInstance",
"lightsail:CreateDomain",
"lightsail:GetDiskSnapshots",
"lightsail:GetRelationalDatabaseMetricData",
"lightsail:PeerVpc",
"lightsail:CreateCertificate",
"lightsail:CreateKeyPair",
"lightsail:SendContactMethodVerification",
"lightsail:GetStaticIps",
"lightsail:GetRegions",
"lightsail:GetOperation",
"lightsail:GetDistributions",
"lightsail:GetDomains",
"lightsail:GetDisks",
"lightsail:CreateDisk",
"lightsail:GetBundles",
"lightsail:GetInstanceMetricData",
"lightsail:GetBucketBundles",
"lightsail:GetContainerServiceMetricData",
"lightsail:GetActiveNames",
"lightsail:GetInstanceSnapshot",
"lightsail:GetOperations",
"lightsail:EnableAddOn",
"lightsail:GetDistributionBundles",
```

```
                "lightsail:GetBlueprints",
                "lightsail:GetContainerAPIMetadata",
                "lightsail:GetCertificates",
                "lightsail:GetLoadBalancerMetricData",
                "lightsail:GetDiskSnapshot",
                "lightsail:DeleteAutoSnapshot",
                "lightsail:CopySnapshot",
                "lightsail:GetDisk",
                "lightsail:GetDistributionMetricData",
                "lightsail:GetRelationalDatabases",
                "lightsail:GetContainerLog",
                "lightsail:GetBucketMetricData",
                "lightsail:ImportKeyPair",
                "lightsail:DownloadDefaultKeyPair",
                "lightsail:IsVpcPeered",
                "lightsail:GetInstanceSnapshots",
                "lightsail:CreateBucket",
                "lightsail:GetRelationalDatabaseLogStreams",
                "lightsail:DeleteInstance",
                "lightsail:DeleteInstanceSnapshot",
                "lightsail:OpenInstancePublicPorts"
            ],
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "lightsail:*",
                "network-firewall:*"
            ],
            "Resource": "arn:aws:lightsail:*:464063468077:Bucket/*"
        }
    ]
}
```

此策略中包含的关键操作有：

```
"lightsail:DeleteInstance",
"lightsail:DeleteInstanceSnapshot",
"lightsail:OpenInstancePublicPorts"
```

此策略可以附加到用户或角色以授予必要的权限。

“‘python import subprocess import random import string import argparse import yaml import os

KEY_PATH = os.path.expanduser(“~/Downloads/LightsailDefaultKey-ap-northeast-1.pem”)

def _get_lightsail_instances(): print(“获取 Lightsail 实例…”) try: result = subprocess.run([“aws”, “lightsail”, “get-instances”], capture_output=True, text=True, check=True) print(“Lightsail 实例获取成功。”) return yaml.safe_load(result.stdout) except subprocess.CalledProcessError as e: print(f”获取 Lightsail 实例时出错: {e}“) return None except yaml.YAMLError as e: print(f”解码 YAML 响应时出错: {e}“) return None except Exception as e: print(f”发生意外错误: {e}“) return None

def _get_lightsail_instance(instance_name): print(f”获取实例详细信息: {instance_name}“) try: result = subprocess.run([” aws” , ” lightsail” , ” get-instance” , ”–instance-name” , instance_name], capture_output=True, text=True, check=True) instance_data = yaml.safe_load(result.stdout) if not instance_data or ‘instance’ not in instance_data: print(f”找不到名称为 {instance_name} 的实例”) return None return instance_data[‘instance’] except subprocess.CalledProcessError as e: print(f”获取实例详细信息时出错: {e}“) return None except yaml.YAMLError as e: print(f”解码 YAML 响应时出错: {e}“) return None except Exception as e: print(f”发生意外错误: {e}“) return None

def create_lightsail_instance(instance_name=None, availability_zone=“ap-northeast-1a”, bundle_id=“nano_2_0”, user_data=None): if not instance_name: random_chars = ’ ’ .join(random.choice(string.ascii_lowercase) for _ in range(4)) instance_name = f”{random_chars}”

```
if not user_data:
    user_data = """#!/bin/bash
    sudo apt update
    """
print(f"创建Lightsail实例，名称: {instance_name}, 区域: {availability_zone}, 套餐: {bundle_id}...")

command = [
    "aws", "lightsail", "create-instances",
    "--instance-names", instance_name,
    "--availability-zone", availability_zone,
    "--bundle-id", bundle_id,
    "--blueprint-id", "ubuntu_24_04"
]

if user_data:
    command.extend(["--user-data", user_data])

try:
    subprocess.run(command, check=True)
    print(f"Lightsail实例 '{instance_name}' 创建成功。")
    return instance_name
```

```
except subprocess.CalledProcessError as e:
    print(f"创建Lightsail实例时出错: {e}")
    return None
```

def delete_all_lightsail_instances(instance_name=None): if instance_name: print(f”删除实例: {instance_name}“) print(f”执行命令: aws lightsail delete-instance –instance-name {instance_name}“) try: subprocess.run([”aws”, ”lightsail”, ”delete-instance”, ”–instance-name”, instance_name], check=True) print(f”Lightsail 实例 ‘{instance_name}’ 删除成功。“) except subprocess.CalledProcessError as e: print(f”删除 Lightsail 实例时出错: {e}“) return

```
instances_yaml = _get_lightsail_instances()
if not instances_yaml or 'instances' not in instances_yaml:
    print("没有找到要删除的Lightsail实例。")
    return

instance_list = instances_yaml['instances']
if not instance_list:
    print("没有找到要删除的Lightsail实例。")
    return

for instance in instance_list:
    instance_name = instance['name']
    print(f"删除实例: {instance_name}")
    print(f"执行命令: aws lightsail delete-instance --instance-name {instance_name}")
    subprocess.run(["aws", "lightsail", "delete-instance", "--instance-name", instance_name], check=True)
print("所有Lightsail实例删除成功。")
```

def install_outline_server(instance_name): instance = _get_lightsail_instance(instance_name) if not instance: return public_ip = instance[‘publicIpAddress’] print(f”在实例 {instance_name} 上安装 Outline 服务器，IP: {public_ip}“) user_data =”“”#!/bin/bash sudo apt update sudo bash -c”$(wget -qO- https://raw.githubusercontent.com/Jigsaw-Code/outline-server/master/src/server_manager/install_scripts/install_server.sh)” “”

```
os.chmod(KEY_PATH, 0o600)
print(f"执行命令: chmod 600 {KEY_PATH}")

ssh_command = [
    "ssh",
    "-i",
    KEY_PATH,
    f"ubuntu@{public_ip}",
    user_data
```

```
]
print(f"执行命令: {' '.join(ssh_command)}")
try:
    subprocess.run(ssh_command, check=True)
    print(f"Outline服务器在 {instance_name} 上安装成功。")
except subprocess.CalledProcessError as e:
    print(f"安装Outline服务器时出错: {e}")
```