إدارة عناوين □□ المحجوزة في □□□□□□□□□□□□

العظيم الناري الجدار خلال من بسهولة حظرها يتم أن يمكن بالخوادم الخاصة □□ عناوين أن الشائعة التحديات من استخدام هي الاستراتيجيات إحدى هذا، من للتخفيف السحابة. خوادم على خاص بشكل ينطبق هذا □□□□□. العنوان حظر يتم عندما بك الخاص □□□□□□□ إلى تعيينها وإعادة □□□□□□□□□□□□ من المحجوزة □□ عناوين كمصدر أيضًا متاح البرمجي النص العملية. هذه لأتمتة □□□□□□ بلغة برمجيًا نصًا المقالة هذه تقدم الحالي. مفتوح على □□□□□□.

النص البرمجي يسمح لك بـ:

- □ التحقق مما إذا كان عنوان □□ محجوزًا معينًا معينًا لـ □□□□□□□ معين.
- □ إعادة تعيين عنوان □□ محجوز جديد إلى □□□□□□□ إذا تم حظر العنوان الحالي.
- □ التحقق مما إذا كان المنفذ 80 مفتوحًا على عنوان □□ المحجوز بطريقة□ بسيطة للتحقق مما إذا كان العنوان يعمل□.

إليك النص البرمجي بلغة □□□□□□:

```
import socket
import os
import argparse
import json
import requests
import time

#              DigitalOcean API
def get_digitalocean_headers():
    api_key = os.environ.get("DO_API_KEY")
    if not api_key:
        print("Error: DO_API_KEY not found in environment variables.")
        return None
    return {
        "Authorization": f"Bearer {api_key}",
        "Content-Type": "application/json"
    }

#                IP          DigitalOcean
def fetch_reserved_ips():
```

```
    headers = get_digitalocean_headers()
    if not headers:
        return None
    try:
        url = "https://api.digitalocean.com/v2/reserved_ips"
        resp = requests.get(url, headers=headers)
        resp.raise_for_status()
        reserved_ips_data = resp.json().get("reserved_ips", [])
        with open('response.json', 'w') as f:
            json.dump(reserved_ips_data, f, indent=4) #
        return reserved_ips_data
    except requests.exceptions.RequestException as e:
        print(f"Error getting reserved IP address: {e}")
        return None

#               IP       Droplet
def unassign_ip_from_droplet(ip_address, droplet_id, droplet_name):
    headers = get_digitalocean_headers()
    if not headers:
        return False

    try:
        url = f"https://api.digitalocean.com/v2/reserved_ips/{ip_address}"
        resp = requests.delete(url, headers=headers)
        resp.raise_for_status()
        print(f"Successfully deleted IP {ip_address} from droplet {droplet_name}")
        return True
    except requests.exceptions.RequestException as e:
        print(f"Error deleting IP {ip_address} from droplet {droplet_name}: {e}")
        return False

#            IP       Droplet
def assign_ip_to_droplet(ip_address, droplet_id, droplet_name):
    headers = get_digitalocean_headers()
    if not headers:
        return False
```

```
    try:
        url = f"https://api.digitalocean.com/v2/reserved_ips/{ip_address}/actions"
        req = {
            "type": "assign",
            "droplet_id": droplet_id
        }
        resp = requests.post(url, headers=headers, json=req)
        resp.raise_for_status()
        print(f"Successfully assigned IP {ip_address} to droplet {droplet_name}")
        return True
    except requests.exceptions.RequestException as e:
        print(f"Error assigning IP {ip_address} to droplet {droplet_name}: {e}")
        return False

#             IP
def process_reserved_ips(reserved_ips, droplet_name, only_check=False):
    if not reserved_ips:
        print("No reserved IPs found in your account.")
        return None

    for reserved_ip in reserved_ips:
        ip_address = reserved_ip.get("ip")
        if not ip_address:
            print("No IP address found for a reserved IP.")
            continue

        droplet = reserved_ip.get("droplet", None)
        if droplet_name:
            if droplet and droplet.get("name") == droplet_name:
                print(f"The reserved IP {ip_address} is assigned to droplet: {droplet_name}")
                if only_check:
                    if check_port_80(ip_address):
                        print(f"Port 80 is open on {ip_address} for droplet {droplet_name}")
                    else:
                        print(f"Port 80 is closed on {ip_address} for droplet {droplet_name}")
```

```
                    return ip_address
                droplet_id = droplet.get("id")
                if droplet_id:
                    if unassign_ip_from_droplet(ip_address, droplet_id, droplet_name):
                        #            IP

                        new_ip = create_new_reserved_ip(droplet_id)
                        if new_ip:
                            print("Sleeping for 10 seconds before assigning new IP...")
                            time.sleep(10)
                            if assign_ip_to_droplet(new_ip, droplet_id, droplet_name):
                                print(f"Successfully assigned new IP {new_ip} to droplet {droplet_name}")
                            else:
                                print(f"Failed to reassign new IP {new_ip} to droplet {droplet_name}")
                        else:
                            print("No available IP to assign")

                else:
                    print(f"Could not unassign IP {ip_address} because droplet ID was not found.")
                return None
            elif droplet:
                print(f"The reserved IP {ip_address} is not assigned to droplet: {droplet_name}")
            else:
                print(f"No droplets are assigned to the reserved IP: {ip_address}")
        else:
            return ip_address
    return None

#           IP
def create_new_reserved_ip(droplet_id):
    headers = get_digitalocean_headers()
    if not headers:
        print("Failed to get DigitalOcean headers.")
        return False
    try:
        url = "https://api.digitalocean.com/v2/reserved_ips"
```

```
        req = {
            "region": "sgp1", #
        }
        print(f"Attempting to create a new reserved IP for droplet ID: {droplet_id}")
        resp = requests.post(url, headers=headers, json=req)
        resp.raise_for_status()
        new_ip = resp.json().get("reserved_ip", {}).get("ip")
        print(f"Successfully created new reserved IP: {new_ip}")
        return new_ip
    except requests.exceptions.RequestException as e:
        print(f"Error creating new reserved IP: {e}")
        return False

#                   80           IP
def check_port_80(ip_address):
    try:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
            s.settimeout(5)
            s.connect((ip_address, 80))
            return True
    except Exception:
        return False

#                    IP
def get_reserved_ip(droplet_name=None, only_check=False):
    reserved_ips = fetch_reserved_ips()
    if reserved_ips is None:
        return None
    return process_reserved_ips(reserved_ips, droplet_name, only_check)

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description="Get DigitalOcean reserved IP address.")
    parser.add_argument("--droplet-name", required=True, help="Name of the droplet to check if the rese
    parser.add_argument("--only-check", action="store_true", help="Only check if the IP is assigned to
    args = parser.parse_args()
```

```
reserved_ip = get_reserved_ip(args.droplet_name, args.only_check)
if reserved_ip:
    print(f"The reserved IP address is: {reserved_ip}")
```

شرح:

1. **المكتبات: استيراد** تم يتم استيراد المكتبات اللازمة للتعامل مع واجهة برمجة تطبيقات DigitalOcean، والمتغيرات الشبكة، والوقت. وتأخير، وطلبات، ومعالجة الوسائط، والتحليل البيئية.
2. **get_digitalocean_headers()**: تسترجع مفتاح API الخاص بـ DigitalOcean من المتغيرات البيئية وتقوم ببناء الرؤوس اللازمة لطلبات API.
3. **fetch_reserved_ips()**: تجلب جميع عناوين IP المحجوزة الخاصة بحساب DigitalOcean المرتبطة بمفتاح API الخاص بك. تقوم بحفظ الاستجابة في ملف `response.json` لأغراض التصحيح.
4. **unassign_ip_from_droplet()**: تلغي تعيين عنوان IP محجوز من Droplet محدد.
5. **assign_ip_to_droplet()**: تعين عنوان IP محجوز إلى Droplet محدد.
6. **process_reserved_ips()**: هذه هي المنطق الأساسي:
   - تقوم بالتكرار عبر جميع عناوين IP المحجوزة.
   - إذا تم توفير `droplet_name`، فإنها تتحقق مما إذا كان عنوان IP هذا معينًا لـ Droplet.
   - إذا كان `only_check` صحيحًا، فإنها تتحقق مما إذا كان المنفذ 80 مفتوحًا على عنوان IP.
   - إذا لم يكن `only_check`، فإنها تلغي تعيين عنوان IP الحالي، وتنشئ عنوان IP جديدًا، وتعيّن العنوان الجديد إلى Droplet.
7. **create_new_reserved_ip()**: تنشئ عنوان IP محجوز جديد في منطقة `sgp1` هذا. يمكنك تغييره.
8. **check_port_80()**: هذه طريقة بسيطة للتحقق مما إذا كان المنفذ 80 مفتوحًا على عنوان IP معين. قد لا يكون العنوان قابلاً للوصول.
9. **get_reserved_ip()**: تقوم بتنظيم وتجلب عناوين IP المحجوزة.
10. **if __name__ == '__main__'**: تحلل الوسائط سطر الأوامر `--droplet-name` و`--only-check` وتستدعي `get_reserved_ip` لتنفيذ النص البرمجي.

الاستخدام: كيفية

1. **إعداد مفتاح API**: قم بتعيين متغير البيئة `DO_API_KEY` بمفتاح API الخاص بـ DigitalOcean.
2. **تشغيل النص البرمجي:**
   - للتحقق مما إذا كان المنفذ 80 مفتوحًا: `python your_script_name.py --droplet-name your_droplet_name --only-check`
   - لإعادة تعيين عنوان IP إلى Droplet: `python your_script_name.py --droplet-name your_droplet_name`

هذا النص البرمجي إطارًا أساسيًا لإدارة عناوين IP المحجوزة. يمكنك توسيعه بناءً على احتياجاتك الخاصة.