Gestión de IPs Reservadas en DigitalOcean

Es un desafío común que las direcciones IP de los servidores puedan ser bloqueadas fácilmente por el Gran Firewall (GFW). Esto es especialmente cierto para los servidores en la nube. Para mitigar esto, una estrategia es utilizar las IPs reservadas de DigitalOcean y reasignarlas a tu droplet cuando la IP actual sea bloqueada. Este post introduce un script en Python para automatizar este proceso. El script también es de código abierto y está disponible en GitHub.

El script te permite:

- Verificar si una IP reservada está asignada a un droplet específico.
- Reasignar una nueva IP reservada a un droplet si la actual está bloqueada.
- Verificar si el puerto 80 está abierto en la IP reservada (una forma simple de verificar si la IP está funcionando).

Aquí está el script en Python:

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

# Función para obtener los encabezados de la API de DigitalOcean
def get_digitalocean_headers():
    api_key = os.environ.get("DO_API_KEY")
    if not api_key:
        print("Error: DO_API_KEY no encontrada en las variables de entorno.")
        return None
    return {
        "Authorization": f"Bearer {api_key}",
        "Content-Type": "application/json"
    }

# Función para obtener todas las IPs reservadas de 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) # Guarda la respuesta en un archivo para depuraci
        return reserved_ips_data
    except requests.exceptions.RequestException as e:
        print(f"Error obteniendo la dirección IP reservada: {e}")
        return None

# Función para desasignar una IP reservada de un 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"IP {ip_address} eliminada exitosamente del droplet {droplet_name}")
        return True
    except requests.exceptions.RequestException as e:
        print(f"Error eliminando la IP {ip_address} del droplet {droplet_name}: {e}")
        return False

# Función para asignar una IP reservada a un 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"IP {ip_address} asignada exitosamente al droplet {droplet_name}")
        return True
    except requests.exceptions.RequestException as e:
        print(f"Error asignando la IP {ip_address} al droplet {droplet_name}: {e}")
        return False

# Función para procesar las IPs reservadas, verificar su asignación y reasignar si es necesario
def process_reserved_ips(reserved_ips, droplet_name, only_check=False):
    if not reserved_ips:
        print("No se encontraron IPs reservadas en tu cuenta.")
        return None

    for reserved_ip in reserved_ips:
        ip_address = reserved_ip.get("ip")
        if not ip_address:
            print("No se encontró una dirección IP para una IP reservada.")
            continue

        droplet = reserved_ip.get("droplet", None)
        if droplet_name:
            if droplet and droplet.get("name") == droplet_name:
                print(f"La IP reservada {ip_address} está asignada al droplet: {droplet_name}")
                if only_check:
                    if check_port_80(ip_address):
                        print(f"El puerto 80 está abierto en {ip_address} para el droplet {droplet_name}
                    else:
                        print(f"El puerto 80 está cerrado en {ip_address} para el 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):
                        # Intentar asignar una nueva IP después de desasignar

                        new_ip = create_new_reserved_ip(droplet_id)
                        if new_ip:
                            print("Durmiendo durante 10 segundos antes de asignar la nueva IP...")
                            time.sleep(10)
                            if assign_ip_to_droplet(new_ip, droplet_id, droplet_name):
                                print(f"Nueva IP {new_ip} asignada exitosamente al droplet {droplet_nam
                            else:
                                print(f"Error al reasignar la nueva IP {new_ip} al droplet {droplet_nam
                        else:
                            print("No hay IPs disponibles para asignar")

                else:
                    print(f"No se pudo desasignar la IP {ip_address} porque no se encontró el ID del dr
                return None
            elif droplet:
                print(f"La IP reservada {ip_address} no está asignada al droplet: {droplet_name}")
            else:
                print(f"No hay droplets asignados a la IP reservada: {ip_address}")
        else:
            return ip_address
    return None

# Función para crear una nueva IP reservada
def create_new_reserved_ip(droplet_id):
    headers = get_digitalocean_headers()
    if not headers:
        print("Error al obtener los encabezados de DigitalOcean.")
        return False
    try:
        url = "https://api.digitalocean.com/v2/reserved_ips"
```

```
            req = {
                "region": "sgp1", # Puedes cambiar la región si es necesario
            }
            print(f"Intentando crear una nueva IP reservada para el 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"Nueva IP reservada creada exitosamente: {new_ip}")
            return new_ip
        except requests.exceptions.RequestException as e:
            print(f"Error creando una nueva IP reservada: {e}")
            return False

# Función para verificar si el puerto 80 está abierto en una dirección 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

# Función principal para obtener la IP reservada
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="Obtener la dirección IP reservada de DigitalOcean.")
    parser.add_argument("--droplet-name", required=True, help="Nombre del droplet para verificar si la I
    parser.add_argument("--only-check", action="store_true", help="Solo verificar si la IP está asignada
    args = parser.parse_args()
```

```
reserved_ip = get_reserved_ip(args.droplet_name, args.only_check)
if reserved_ip:
    print(f"La dirección IP reservada es: {reserved_ip}")
```

Explicación:

1. **Importar Librerías:** Importa las librerías necesarias para operaciones de red, variables de entorno, análisis de argumentos, manejo de JSON, solicitudes HTTP y retrasos de tiempo.
2. `get_digitalocean_headers()`**:** Obtiene la clave API de DigitalOcean de las variables de entorno y construye los encabezados necesarios para las solicitudes API.
3. `fetch_reserved_ips()`**:** Obtiene todas las IPs reservadas asociadas con tu cuenta de DigitalOcean usando la API. También guarda la respuesta cruda en `response.json` para fines de depuración.
4. `unassign_ip_from_droplet()`**:** Desasigna una IP reservada dada de un droplet específico.
5. `assign_ip_to_droplet()`**:** Asigna una IP reservada dada a un droplet específico.
6. `process_reserved_ips()`**:** Esta es la lógica central:
   - Itera a través de todas las IPs reservadas.
   - Si se proporciona un `droplet_name`, verifica si la IP está asignada a ese droplet.
   - Si `only_check` es verdadero, verifica si el puerto 80 está abierto y devuelve la IP.
   - Si no es `only_check`, desasigna la IP actual, crea una nueva y asigna la nueva IP al droplet.
7. `create_new_reserved_ip()`**:** Crea una nueva IP reservada en la región `sgp1` (puedes cambiarla).
8. `check_port_80()`**:** Verifica si el puerto 80 está abierto en una dirección IP dada. Esta es una forma simple de verificar si la IP es accesible.
9. `get_reserved_ip()`**:** Orquesta el proceso de obtener y procesar las IPs reservadas.
10. `if __name__ == '__main__':`**:** Analiza los argumentos de la línea de comandos (`--droplet-name` y `--only-check`) y llama a `get_reserved_ip` para ejecutar el script.

Cómo Usar:

1. **Configurar la Clave API de DigitalOcean:** Establece la variable de entorno `DO_API_KEY` con tu clave API de DigitalOcean.
2. **Ejecutar el script:**
   - Para verificar si una IP está asignada a un droplet y si el puerto 80 está abierto: `bash python nombre_de_tu_script.py --droplet-name nombre_de_tu_droplet --only-check`

- Para reasignar una nueva IP a un droplet: `bash python nombre_de_tu_script.py --droplet-name nombre_de_tu_droplet`

Este script proporciona un marco básico para gestionar IPs reservadas. Puedes extenderlo según tus necesidades específicas.