LAN IP 掃描器

此 Python 腳本掃描本機網絡中的活躍 IP 地址。它使用 `ping` 命令檢查主機是否可達，並使用多線程來加快掃描過程。信號量限制同時運行的線程數量，以避免壓垮系統。腳本以網絡地址（例如，「192.168.1.0/24」）作為輸入，並打印網絡中每個 IP 地址的狀態（正常或關閉）。

```
import subprocess
import ipaddress
import threading
import os

MAX_THREADS = 255  # 最大同時執行線程數

def is_host_up(host):
    """
    使用 ping 檢查主機是否正常。
    如果主機正常則返回 True，否則返回 False。
    """
    try:
        # -c 1: 只發送 1 個數據包
        # -W 1: 等待 1 秒鐘以接收響應
        subprocess.check_output(["ping", "-c", "1", "-W", "1", host], timeout=1)
        return True
    except subprocess.CalledProcessError:
        return False
    except subprocess.TimeoutExpired:
        return False

def scan_ip(ip_str):
    """
    掃描單個 IP 地址並打印其狀態。
    """
    if is_host_up(ip_str):
        print(f"{ip_str} is up")
    else:
        print(f"{ip_str} is down")

def scan_network(network):
    """
    使用線程掃描網絡中的活躍主機，限制同時運行的線程數。
```

```
    """
    print(f"Scanning network: {network}")
    threads = []
    semaphore = threading.Semaphore(MAX_THREADS)  # 限制同時執行線程數

    def scan_ip_with_semaphore(ip_str):
        semaphore.acquire()
        try:
            scan_ip(ip_str)
        finally:
            semaphore.release()

    for ip in ipaddress.IPv4Network(network):
        ip_str = str(ip)
        thread = threading.Thread(target=scan_ip_with_semaphore, args=(ip_str,))
        threads.append(thread)
        thread.start()

    for thread in threads:
        thread.join()

if __name__ == "__main__":
    network_to_scan = "192.168.1.0/24"  # 將此更改為您的網絡
    scan_network(network_to_scan)
```