

# लैन आईपी स्कैनर

यह पाइथन स्क्रिप्ट सक्रिय आईपी पत्तों के लिए एक स्थानीय नेटवर्क को स्कैन करता है। यह जांचने के लिए कि कोई होस्ट पहुँचा जा सकता है या नहीं, ping कमांड का उपयोग करता है और स्कैनिंग प्रक्रिया को तेज करने के लिए मल्टीथ्रेडिंग का उपयोग करता है। सिस्टम को ओवरहेल्म करने से बचने के लिए एक सेमाफोर एक साथ चलने वाले थ्रेड्स की संख्या को सीमित करता है। स्क्रिप्ट इनपुट के रूप में एक नेटवर्क पता (जैसे, "192.168.1.0/24") लेती है और प्रिंट करती है कि नेटवर्क में प्रत्येक आईपी पता चालू है या बंद।

```
import subprocess
import ipaddress
import threading
import os

MAX_THREADS = 255 #

def is_host_up(host):
    """
    True , False
    """
    try:
        # -c 1: 1
        # -W 1: 1
        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):
    """
    """
    if is_host_up(ip_str):
        print(f"{ip_str} is up")
    else:
        print(f"{ip_str} is down")

def scan_network(network):
    """
```

```

'''
print(f"Scanning network: {network}")
threads = []
semaphore = threading.Semaphore(MAX_THREADS) #

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" #
    scan_network(network_to_scan)

```