Bash script that pings every IP in the 23.227.36.x subnet (where x is from 0 to 255) and displays whether each server is up or unreachable.

```
#!/bin/bash

# Loop through all possible values of x (0-255)

for x in {0..255}; do

  # Ping the IP address with a timeout of 1 second and send only 1 packet

  if ping -c 1 -W 1 23.227.36.$x > /dev/null 2>&1

  then

    echo "Server 23.227.36.$x is up and running."

  else

    echo "Server 23.227.36.$x is unreachable."

  fi

done
```

Explanation:

1. **for x in {0..255}**: Iterates over the range 0-255 to cover all possible IP addresses in the 23.227.36.x subnet.
2. **ping -c 1 -W 1 23.227.36.$x > /dev/null 2>&1**:
    - -c 1: Sends only one ping request.
    - -W 1: Sets a timeout of 1 second to avoid long delays.
    - > /dev/null 2>&1: Suppresses the output to keep the script clean.
3. **Conditional Check (if-else)**:
    - If the ping succeeds, it prints "Server 23.227.36.x is up and running."
    - If the ping fails, it prints "Server 23.227.36.x is unreachable."

Multiple IPs

```
#!/bin/bash

IPLIST="path_to_the_Ip_list_file"

for ip in $(cat $IPLIST); do

  ping -c1 $ip &> /dev/null
```

```
    if [ $? -eq 0 ]; then
        echo $ip ping passed
    else
        echo $ip ping failed
    fi
done
```

Тук скриптът използва командата `ping` за проверка дали определен уебсайт е достъпен. Специалната променлива `$?` съдържа статус кода на последната изпълнена команда, който се използва за проверка на резултата от `ping`.

---

port 22 на 87.246.47.66 не е open, тогава спри. Check for open port

```
#!/bin/bash
IP=87.246.47.66
nmap -sT -Pn -p 22 $IP | egrep -q 'open'
if [[ $? -ne 0 ]]; then
  echo "IP $IP Not connection"
   exit 1
fi
```

- **`nmap`**: The command-line utility used for network discovery and security auditing.
- **`-sT`**: This option tells **`nmap`** to perform a TCP connect scan. In a TCP connect scan, **`nmap`** tries to connect to the target ports to determine whether they are open, closed, or filtered. This method is less stealthy than other scan types like SYN scan (**`-sS`**), but it's the most accurate in determining the state of the ports.
- **`-Pn`**: This option tells **`nmap`** not to ping the target host. By default, **`nmap`** sends an ICMP echo request to the target to check if it's up before scanning. The **`-Pn`** option skips this step and assumes the target is up.
- **`-p 22`**: This option specifies that **`nmap`** should scan port 22 on the target. Port 22 is the default port for SSH (Secure Shell) service.
- **`$IP`**: This is a variable representing the target IP address.
- **`|`**: This is a pipe, which takes the output of the **`nmap`** command and passes it as input to the next command.

- `egrep -q 'open'`: This command uses `egrep` to search the output of `nmap` for the word "open". The `-q` option makes `egrep` quiet; it doesn't produce any output. Instead, it sets an exit status:
    - If "open" is found in the `nmap` output, `egrep` will exit with a status of 0 (success).
    - If "open" is not found, `egrep` will exit with a status of 1 (failure).

, `exit 1` is a command used to terminate the current script or shell session with an exit status of 1. The exit status, or exit code, is a numerical value returned by a process to its parent process upon completion. By convention, an exit status of 0 usually signifies success, while any non-zero value (like 1) signifies failure or some sort of error.

Here's a brief explanation:

- `exit`: This is the command used to exit the shell or script.
- `1`: This is the exit status that the command passes to the parent process. It indicates an error or an abnormal termination.

Checking if a Server Is Running

Imagine you're managing a group of servers, and you need to check if one of them is running. Doing this manually for 10 or 20 servers would take forever! Instead, you can write a Bash script that checks the server's status for you.

Here's a script for that:

```
#!/bin/bash

# Define the server's IP or hostname

SERVER="192.168.1.100"

# Ping the server to check if it's up

if ping -c 1 $SERVER &> /dev/null; then

        echo "Server $SERVER is running!"

else

        echo "Server $SERVER is not reachable."

fi
```

Explanation:

- The SERVER variable holds the server's IP address.
- The ping command sends a small message to the server to see if it responds.
- The if statement checks if the ping command works. If it does, the script says the server is running. If not, it says the server isn't reachable.
- You can run this script on your terminal to quickly check your server's status.
