Objective: Implement a network interceptor on a Linux system to manage API requests.

Scenario:

- The system makes a request to an external API, such as `https://workingapi.com`.
- When a request is made to a specific URI, like `https://workingapi.com/books`:
  - If the API returns a `200 OK` response, the system should return the API's original response.
  - If the API returns any other status code or fails to respond, the request should be redirected to a local mock server on the machine at `https://localhost:3000/books`, which will provide a fallback response.

Requirements:

- The solution should run on a Linux environment.
- The interceptor should seamlessly handle the request and redirection based on the API's response.

1 - Network interceptor installing mode:

sudo pacman -S iptables
sudo pacman -S mitmproxy

how to run the script:

mitmproxy -s proxy_script.py --listen-port 8888

Handle HTTPS Traffic

Generate the certificate:

mitmproxy --mode regular

This will generate the certificate files in ~/.mitmproxy.

Install the certificate:

For system-wide use, you can copy the mitmproxy-ca-cert.pem file to /etc/ca-certificates/trust-source/anchors/ and update the certificates:

sudo cp ~/.mitmproxy/mitmproxy-ca-cert.pem /etc/ca-certificates/trust-source/anchors/mitmproxy-ca-cert.pem
sudo update-ca-trust

Add a Fake DNS Entry for workingapi.com

To simulate the domain locally, you can map workingapi.com to 127.0.0.1 (or another IP) using the /etc/hosts file.

Edit the /etc/hosts file:

sudo nano /etc/hosts

127.0.0.2 workingapi.com

127.0.0.2 workingapi.com

Update iptables or Use curl with the Local Address

sudo iptables -t nat -A OUTPUT -p tcp --dport 80 -d 127.0.0.2 -j REDIRECT --to-ports 3000