

Mad-proxy

Project

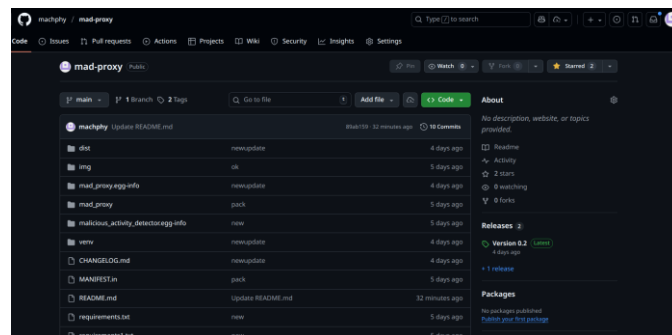
Project Name: mad-proxy

Version: v0.2 (Update as per version)

Maintainer: machphy (Rajeev Sharma)

Repository:

<https://github.com/machphy/mad-proxy>



Aim

mad-proxy is a Python-based local HTTP/HTTPS proxy server designed to intercept, inspect, and block malicious web traffic on the fly. It helps cybersecurity professionals monitor browser traffic and enforce security policies such as blocking unauthorized or suspicious websites using custom rules applied in real-time.

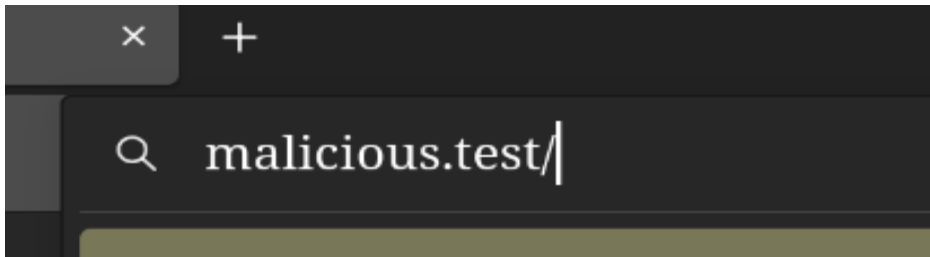
Architecture

The project contains the following components:

- **Browser:** Routes web traffic through the local proxy.
- **mad-proxy (proxy_server.py):** Intercepts browser HTTP/HTTPS traffic.
- **Policy Engine (policy_engine.py and config.yaml):** Applies user-configurable security policies to allow or block traffic.
- **Logging (utils.py):** Handles activity logs and alerts.
- **Analyzer (analyzer.py - planned):** For advanced traffic and content analysis.

The data flow:

Browser → mad-proxy proxy_server → Policy Engine → Internet



Search Engine (malicious.Test) which already make policy in `config.yaml`

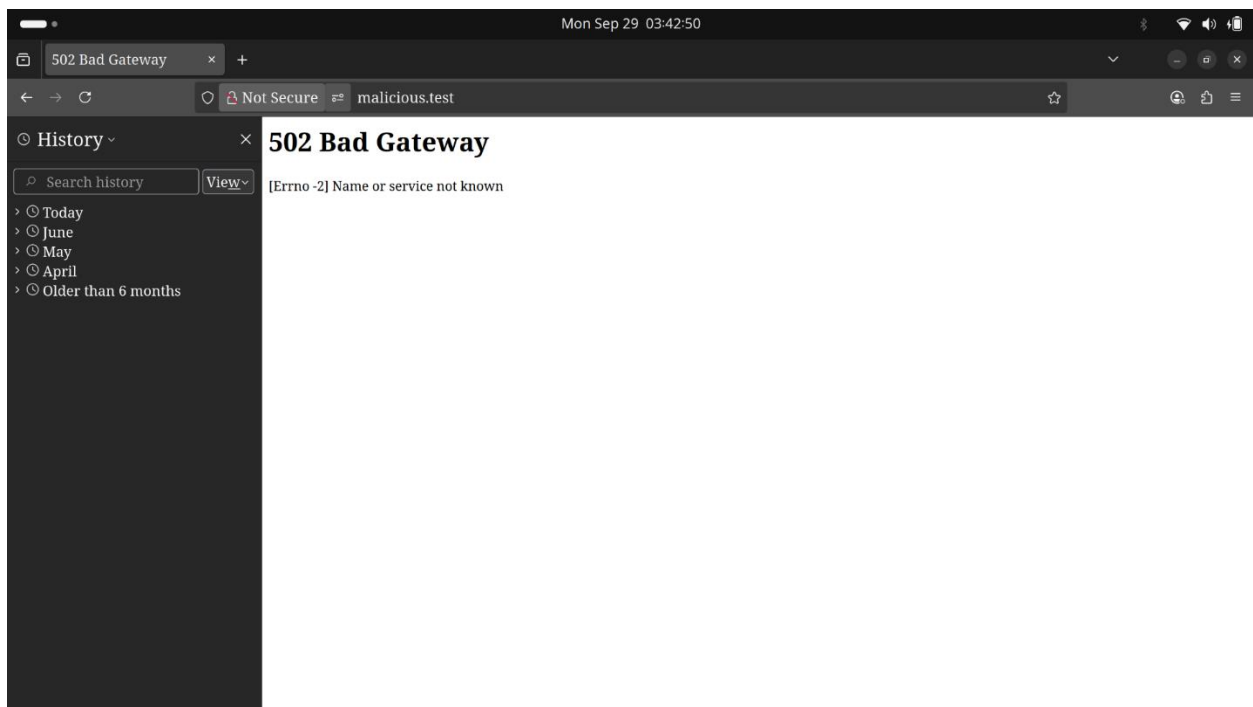
Blocked Domain

```
1  ∨ block_domains:
2      - "example.com"
3      - "malicious.test"
4      - "spammy.site"
5      - "phishing.co"
6      - "unwanted.org"
7      - "ads.example"
8      - "tracker.net"
9      - "fake-news.cm"
10     - "scam-site.io"
11     - "clickbait.info"
12     - "harmful.biz"
13     - "dangerous.co.uk"
14     - "untrustworthy.us"
15     - "fraudulent.ca"
16     - "suspicious.eu"
17     - "unsecure.xyz"
18     - "risky.online"
19     - "threatening.site"
20     - "compromised.org"
21     - "insecure.net"
22     - "deceptive.com"
23     - "illegitimate.info"
24     - "unethical.biz"
25     - "dodgy.co"
```

Working -

When a domain is listed under ***block_domains*** in the ***config.yaml*** policy file, all HTTP(s) requests to this domain will be blocked by the proxy. Specifically:

- The proxy server intercepts each request and matches the URL against the block list.
- If the domain matches a blocked pattern or domain, the proxy returns an HTTP 403 Forbidden response.
- The browser will receive this error, and the request will fail to load.
- No automatic redirection or fallback occurs unless explicitly configured by the user in advanced policy settings.
- This ensures strict enforcement of security policies by preventing access to unauthorized or suspicious domains.



Here is terminal view

```

[03:32:39.262][127.0.0.1:55702] server disconnect firefox-settings-attachments.cdn.mozilla.net:443 (151.101.1.91:443)
[03:32:40.261][127.0.0.1:55804] client disconnect
[03:32:40.262][127.0.0.1:55804] server disconnect merino-images.services.mozilla.com:443 (34.110.253.53:443)
[03:32:42.261][127.0.0.1:55722] client disconnect
[03:32:42.262][127.0.0.1:55722] server disconnect detectportal.firefox.com:80 (34.107.221.82:80)
[03:32:43.262][127.0.0.1:55878] client disconnect
[03:32:43.263][127.0.0.1:55878] server disconnect detectportal.firefox.com:80 (34.107.221.82:80)
[03:32:43.263][127.0.0.1:55876] client disconnect
[03:32:43.264][127.0.0.1:55876] server disconnect detectportal.firefox.com:80 (34.107.221.82:80)
[03:32:44.262][127.0.0.1:55684] client disconnect
[03:32:44.263][127.0.0.1:55684] server disconnect firefox.settings.services.mozilla.com:443 (151.101.65.91:443)
127.0.0.1:57154: POST https://www.google.com/gen_204?atyp=i&ei=HLDZaMKaHeSgseMPnfD8qAs&ct=slh&v=t1&im=M&pv=0.9344645404683088&me=9:1759096864065,V... HTTP/2.0
<< HTTP/2.0 204 No Content 0b
127.0.0.1:57154: POST https://www.google.com/gen_204?atyp=i&ei=HLDZaMKaHeSgseMPnfD8qAs&ct=slh&v=t1&im=M&pv=0.9344645404683088&me=14:1759096864065,V... HTTP/2.0
<< HTTP/2.0 204 No Content 0b
127.0.0.1:57204: POST https://play.google.com/log?hasfast=true&auth=SAPISIDHASH+f0aaf4fee9514455c3faff29b424b9f05719f8ff+SAPISID1PHASH+f0aaf4fee9514455c3faff29b424b9f05719f8ff HTTP/2.0
<< HTTP/2.0 401 Unauthorized 1.8k
[03:32:51.431][127.0.0.1:55378] client disconnect
[03:32:51.432][127.0.0.1:55378] server disconnect img.getpocket.cdn.mozilla.net:443 (34.120.237.76:443)
[03:33:07.554][127.0.0.1:55864] client disconnect
[03:33:07.555][127.0.0.1:55864] server disconnect incoming.telemetry.mozilla.org:443 (34.120.208.123:443)

```

When a domain or URL does not match any entry in the `block_domains` list or is explicitly listed in an allow list, the proxy:

- Permits the request to pass through to the destination server without interruption.
- Receives the response from the server and forwards it back to the browser.
- The browser receives the normal HTTP response, typically HTTP 200 OK for successful requests.
- This allows normal browsing experience for safe and allowed websites.
- All allowed requests are logged in the terminal or log files for auditing and monitoring purposes.

Summary of Blocking vs Allowing

Situation	Proxy Action	Browser Response
URL in block list	Block request, return HTTP 403	Shows “Blocked” error
URL allowed or none	Forward request, return HTTP 200	Loads site normally

1. Install the mad-proxy Package


Use the following command to install the package from PyPI:

```

bash
pip install mad-proxy

```

mad-proxy 0.2

`pip install mad-proxy`

✓ Latest version

Released: Sep 29, 2025

A local HTTP/HTTPS proxy with custom detection and blocking policies.

Manage project

This installs the latest version of mad-proxy on your system.

2. Verify the Installation

Confirm that the package is installed correctly by running:

```
bash
pip show mad-proxy
```

Alternatively, open a Python terminal and check the version:

```
python
import mad_proxy
print(mad_proxy.__version__)
```

3. Configure the Proxy Server Settings

To start the proxy server (default port 8080), execute:

```
bash
mad-proxy
```

Or, if the CLI command is not available, start it via Python module:

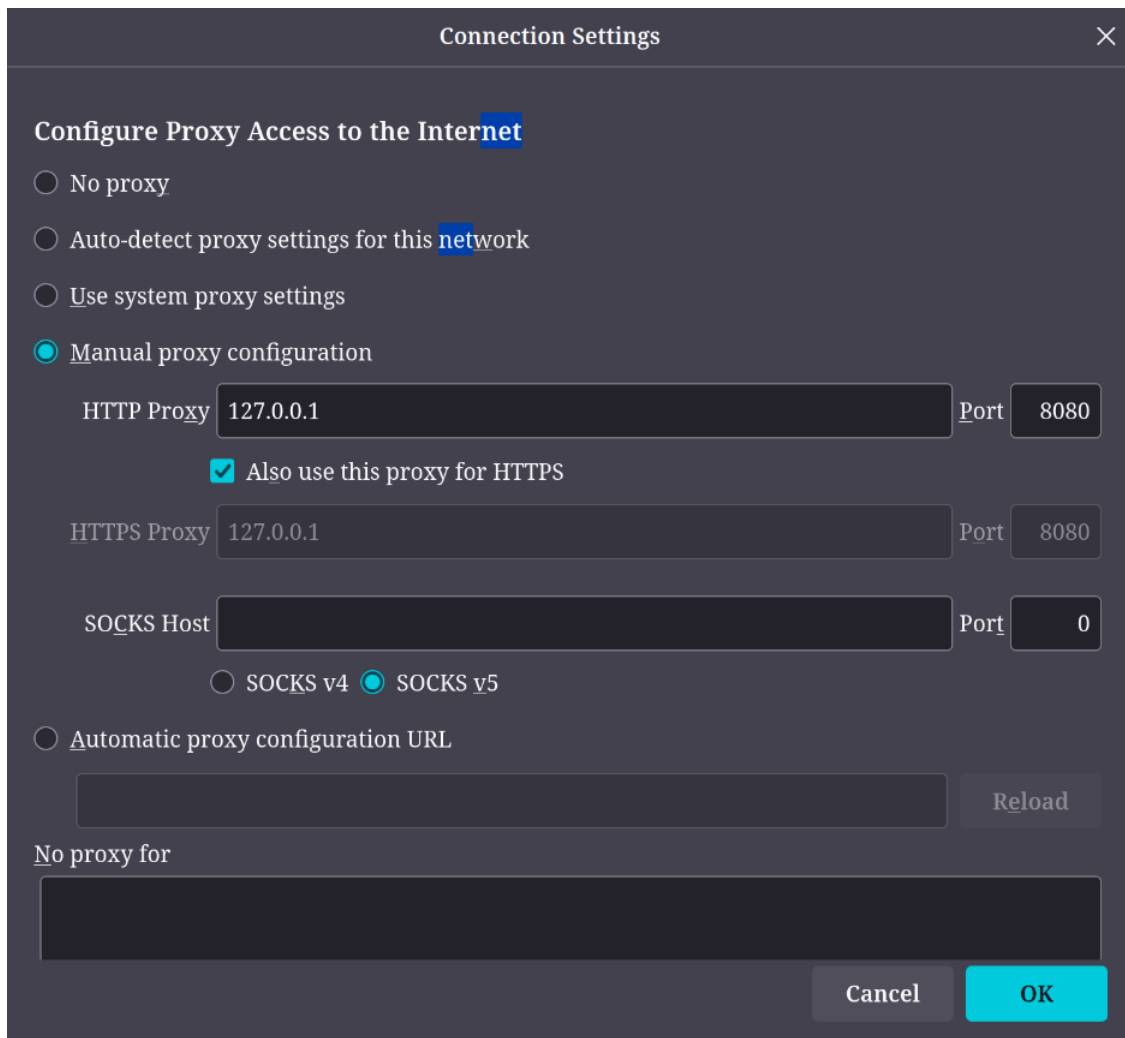
```
bash
python -m mad_proxy.proxy_server
```

4. Configure Your Browser to Use the Local Proxy

To route browser traffic through the mad-proxy server, follow these steps:

For all browsers (Firefox, Chrome, Edge, etc.):

1. Open your browser's network or proxy settings:
 - a. In most browsers, find this under **Settings > Network or System > Proxy Settings**.
2. Set Manual Proxy Configuration:
 - a. HTTP Proxy: localhost
 - b. Port: 8080
 - c. HTTPS Proxy (if separate option): localhost
 - d. Port: 8080
3. Save the proxy settings.



The screenshot shows a 'Connection Settings' dialog box with a close button (X) in the top right corner. The title is 'Connection Settings'. Below the title, there is a section 'Configure Proxy Access to the Internet' with four radio button options: 'No proxy', 'Auto-detect proxy settings for this network', 'Use system proxy settings', and 'Manual proxy configuration'. The 'Manual proxy configuration' option is selected. Below this, there are three rows of proxy settings. The first row is for 'HTTP Proxy' with a text field containing '127.0.0.1' and a 'Port' field containing '8080'. Below this row is a checked checkbox labeled 'Also use this proxy for HTTPS'. The second row is for 'HTTPS Proxy' with a text field containing '127.0.0.1' and a 'Port' field containing '8080'. The third row is for 'SOCKS Host' with an empty text field and a 'Port' field containing '0'. Below this row are two radio button options: 'SOCKS v4' and 'SOCKS v5', with 'SOCKS v5' selected. Below these options is a radio button labeled 'Automatic proxy configuration URL' which is not selected. Below this is an empty text field and a 'Reload' button. At the bottom, there is a section 'No proxy for' with an empty text field. At the very bottom right are 'Cancel' and 'OK' buttons.

Connection Settings

Configure Proxy Access to the Internet

☐ No proxy

☐ Auto-detect proxy settings for this network

☐ Use system proxy settings

☒ Manual proxy configuration

HTTP Proxy 127.0.0.1 Port 8080

☒ Also use this proxy for HTTPS

HTTPS Proxy 127.0.0.1 Port 8080

SOCKS Host Port 0

☐ SOCKS v4 ☒ SOCKS v5

☐ Automatic proxy configuration URL

Reload

No proxy for

Cancel OK

Note:

- This configuration sends all HTTP and HTTPS traffic through the mad-proxy server running on your local machine.
- Ensure the mad-proxy service is running before browsing.

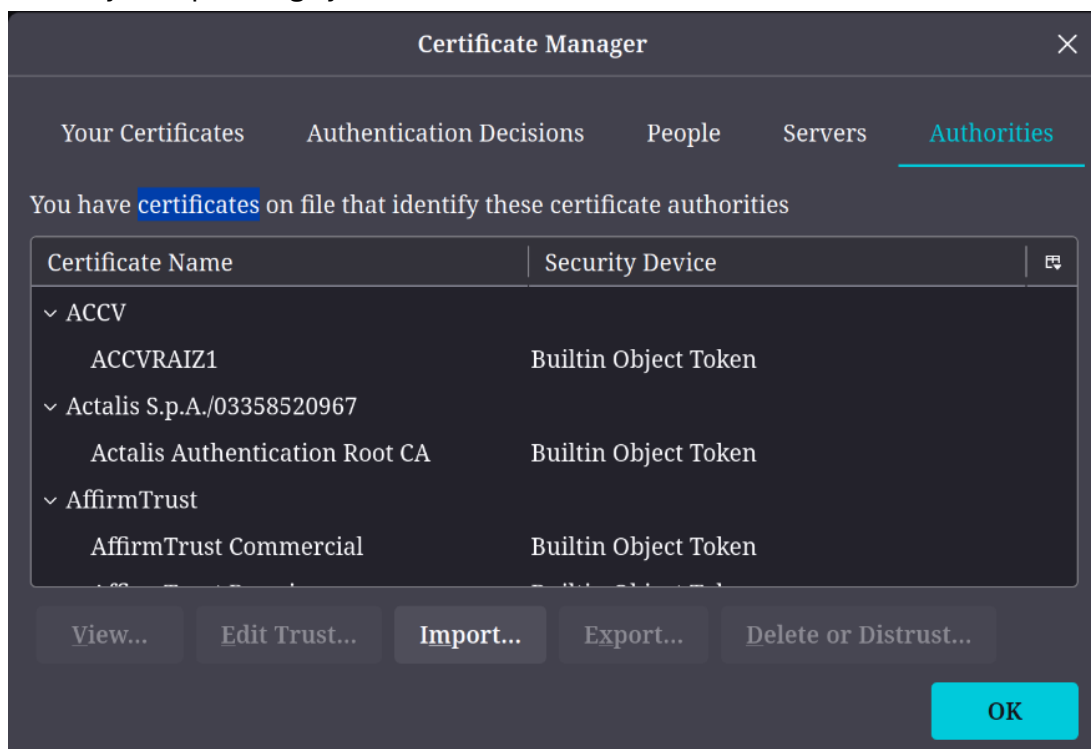
5. Install and Trust the mitmproxy Root Certificate

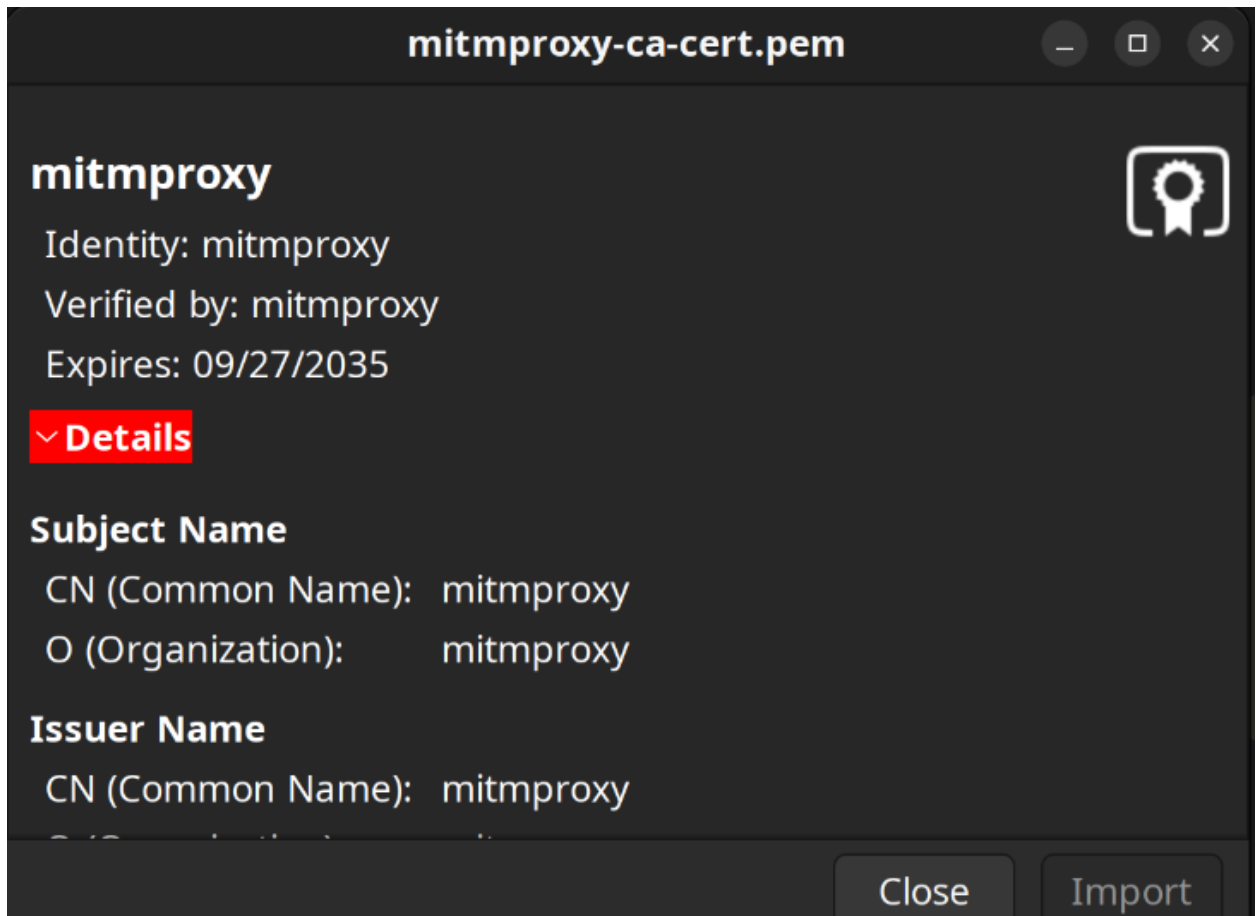
HTTPS traffic requires the installation of the mitmproxy root certificate to enable trusted interception. Follow these steps:

1. While the mad-proxy server is running, open your browser and visit:

<http://mitm.it>

2. Download the appropriate certificate for your operating system and browser.
3. Follow your operating system's instructions to trust and install the certificate.





6. Test Your Setup

- Visit allowed websites (e.g., <https://www.google.com>); the page should load normally.
- Access blocked domains configured in your policy; the browser should display a "Blocked by security policy" message.

Logs showing allowed and blocked requests will appear in the terminal where mad-proxy runs.

Summary

Step	Command / Action
Install package	pip install mad-proxy
Start proxy server	mad-proxy or python -m mad_proxy.proxy_server
Configure browser proxy	HTTP & HTTPS Proxy to localhost:8080
Install mitmproxy certificate	Visit http://mitm.it and install certificate

Following these steps enables your browser to send all traffic through mad-proxy, allowing it to inspect and enforce your custom security policies effectively.

File Structure and Purpose

File / Directory	Purpose
proxy_server.py	Core proxy server; intercepts and routes traffic.
policy_engine.py	Applies filtering/blocking rules per config.
config.yaml	User definable security policies (block/allow).
analyzer.py	(Planned) Content & traffic analysis tools.
utils.py	Helper functions for logging and alerts.
requirements.txt	Project dependencies
README.md	Project documentation
CHANGELOG.md	Version log and update history
setup.py	Packaging and build instructions
MANIFEST.in	Include files for packaging

How It Works

1. **Start Proxy Server:** Run `python3 proxy_server.py` which initiates a local HTTP/HTTPS proxy (default port 8080).
2. **Configure Browser:** Set HTTP and HTTPS proxy to localhost:8080.

3. **HTTPS Setup:** Install mitmproxy root certificate by visiting <http://mitm.it> with proxy running.
4. **Traffic Interception:** All browser requests go through mad-proxy.
5. **Policy Application:** The policy engine checks each request URL against configured block/allow rules in config.yaml.
 - a. If blocked, a 403 Forbidden response is sent.
 - b. If allowed, the request is forwarded normally.
6. **Logging:** All decisions logged to terminal or file as configured.
7. **User Configuration:** Edit config.yaml to update block/allow domains or policies.

Setup and Installation

Prerequisites

- Python 3.7 or higher (3.12+ recommended)
- pip package manager
- mitmproxy installed (pip install mitmproxy)
- Linux environment (Ubuntu/Debian recommended)

Steps

1. Clone repo:

```
bash
git clone https://github.com/machphy/mad-proxy.git
cd mad-proxy
```

2. Create and activate virtual environment:

```
bash
python3 -m venv venv
source venv/bin/activate
```

3. Install dependencies:

```
bash
pip install -r requirements.txt
```

4. Build package: (Optional)

```
bash
```

```
python3 -m build
```

5. Install package locally:

```
bash
```

```
pip install dist/mad_proxy-<version>-py3-none-any.whl
```

6. Run the proxy server:

```
bash
```

```
python3 proxy_server.py
```

Usage and Testing

- **Allowed Request:** Access allowed domains (e.g., <https://www.google.com>).
Console logs:
Allowed request: <https://www.google.com>
- **Blocked Request:** Access blocked domains (e.g., <http://example.com>). Console logs:
Blocked request to <http://example.com>
Browser shows HTTP 403 and blocked message.

Extending the Project

- Expand policy_engine.py rules to support regex and heuristics.
- Develop analyzer.py for content-based traffic analysis.
- Add alerting systems (desktop, email, webhook).
- Build UI for easier configuration management.
- Integrate threat intelligence feeds for dynamic policy updates.

Troubleshooting

- **Proxy certificate errors:** Reinstall mitmproxy root cert from <http://mitm.it>.
- **Port conflicts:** Change proxy port if 8080 is busy.
- **Configuration errors:** Check YAML syntax in config.yaml.

Maintainer

Maintained by [machphy](#) & rajeevsharmamachphy@gmail.com
Rajeev Sharma