mitmproxy Intercepting HTTP(S) proxy

START HERE

`mitmproxy`
Start proxy on :8080

`i`
Set intercept filter

`f`
Filter the flow list

`Enter`
Open flow

`e`
Edit a flow part

`a`
Resume intercepted flow

`r`
Replay flow

`?`
Context help

Front ends

Tool	Use
`mitmproxy`	Interactive terminal UI
`mitmweb`	Web UI on :8081
`mitmdump`	Headless, like tcpdump for HTTP

CONCEPTS

Only intercept traffic from devices and apps you own or are explicitly authorised to inspect.

Three front ends share one engine: mitmproxy (TUI), mitmweb (browser UI) and mitmdump (no UI, scripts and files).

Client proxy set to :8080
↓ HTTP(S)
mitmproxy decrypts with its CA
↓ re-encrypted
Server real destination
Where mitmproxy sits

HTTPS only works once the client trusts the mitmproxy CA; browse to mitm.it through the proxy to install it.

FLOW LIST

`j` / `k` Move down / up

`g` / `G` Go to first / last flow

`Enter` Open the focused flow

`q` Back / leave current view

`Space` Page down

`f` Set the view filter

`m` Mark / unmark flow

`M` Show only marked flows

`U` Clear all marks

`o` Choose flow list order

`v` Reverse list order

`d` Delete flow from view

`D` Duplicate flow

`z` Clear the whole flow list

`Z`
Purge flows hidden by filter

`F` Toggle focus follow (tail)

`n` Create a new request flow

`X` Kill a flow in progress

`;` Add a comment to flow

FLOW VIEW

`Tab`
Switch request / response / detail

`e`
Edit URL, header, body or method

FLOW VIEW (cont.)

`m`
Choose content view (JSON, hex...)

`z`
Toggle body encoding (gzip etc.)

`v`
Open body in external viewer

`p` / `Space`
Previous / next flow

`V` Revert edits to flow

`Ctrl` + `Right` / `Ctrl` + `Left`
Next / previous pane

`-` Cycle layout

INTERCEPT AND REPLAY

`i`
Set intercept filter, e.g. ~u api & ~q

`I` Toggle intercept on / off

`a`
Resume the intercepted flow

`A`
Resume all intercepted flows

`r`
Replay request to the server

`S`
Start server replay of a flow set
Serves saved responses instead of contacting servers.

`|` Run a script on this flow

`x`
Export flow (curl, httpie, raw...)

`b` Save response body to file

SAVE, LOAD, GLOBAL

`w` Save listed flows to file

`L` Load flows from file

`W` Stream flows to file live

`E` Open event log

`O` Open options

`:` Command prompt

`B`
Launch an attached browser

`C` List commands

`K` Show key bindings

`?` Context-sensitive help

`Q` Quit immediately

FILTER EXPRESSIONS

`~u` *`<regex>`*
Match URL

`~d` *`<regex>`*
Match domain

`~m` *`<regex>`*
Match method

`~c` *`<code>`*
Match response status code

`~h` *`<regex>`*
Match any header

`~b` *`<regex>`*
Match any body

`~bq / ~bs` *`<regex>`*
Match request / response body

`~hq / ~hs` *`<regex>`*
Match request / response header

`~t` *`<regex>`*
Match Content-Type

`~q / ~s`
Request without / with response

`~e`
Match flows with errors

`! / & / | / ( )`
Not, and, or, grouping

Regexes are case-insensitive. A bare word with no ~ is matched against the URL.

COMMAND LINE

`mitmproxy -p` *`<port>`*
Listen on another port

`mitmweb`
Web interface

`mitmdump -w` *`<file>`*
Record flows to file

`mitmproxy -r` *`<file>`*
Open a saved flow file

`mitmdump -nr` *`<file>`*
Read file without proxying

`mitmdump "`*`<filter>`***`"`**
Show only matching flows

`--mode regular`
Explicit proxy (default)

`--mode local[:`*`<app>`***`]`**
Capture local apps by name or PID

`--mode reverse:`*`<url>`***
Front a single server

`--mode transparent`
Network-level capture (Linux, macOS)

`--mode socks5`
Run as SOCKS5 proxy

COMMAND LINE (cont.)

`--mode wireguard`
WireGuard server for devices

`--mode upstream:`*`<url>`*
Chain to another proxy

`-s` *`<script.py>`*
Load an addon script

`--set` *`<option>`*`=`*`<value>`*
Set any option

`--ignore-hosts` *`<regex>`*
Do not decrypt matching hosts
Use for apps that pin certificates.

`--allow-hosts` *`<regex>`*
Only decrypt matching hosts

`--set block_global=false`
Accept non-local clients

`--set client_replay=`*`<file>`*
Replay flows at startup

`--set server_replay=`*`<file>`*
Serve responses from a flow file

CERTIFICATES

`http://mitm.it`
Download CA via the proxy

`~/.mitmproxy/mitmproxy-ca-cert.pem`
CA certificate (non-Windows)

`~/.mitmproxy/mitmproxy-ca.pem`
CA with private key: keep secret

`curl --cacert` *`<ca.pem>`* **`-x`** *`<proxy>`* *`<url>`*
Trust CA for one curl call

The CA is generated per installation on first start. Remove it from devices when you finish testing.

RECIPES

Capture then inspect later

```
mitmdump -w session.flows
# use the app through the
proxy, Ctrl+C
mitmproxy -r session.flows
```

Watch one app's API calls

```
mitmproxy --mode local:curl
# in the TUI: f  ~d
api.example.com & ~m POST
curl https://api.example.com/x
```

Reverse proxy a local service

```
mitmweb --mode reverse:http://
localhost:3000 \
  --listen-port 8080
```

Modify with an addon

```
# addon.py
def response(flow):
    flow.response.headers["x-
seen"] = "1"
# mitmdump -s addon.py
```