



# Tezos development with chinstrap

building, testing and deploying Tezos smart contracts

Chai, Co-Founder



# Chai

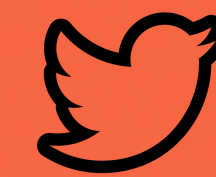
Co-founder and hacker @wefuzz\_io



You can come to me for:

- ✓ Auditing and Bug bounties
- ✓ Vulnerability Research
- ✓ Blockchain and smart contract security
- ✓ Fuzzing

Contact:



[ant4g0nist](#)



[chai@wefuzz.io](mailto:chai@wefuzz.io)

# What we will be doing today

- Chinstrap introduction
- Chinstrap CLI Crash course
- Building and testing SmartPy contracts with Chinstrap
- Building and testing \*Ligo contracts with Chinstrap
- Sandbox for local development
- Originating smart contracts to Localnet/Mainnet

# Chinstrap

- Mission 🚬:
  - *makes developers' lives easier by providing support for multiple contract compilations, tests, and origination on public and private Tezos networks.*
- Backed by Tezos Foundation 🎉 ₮
- Fully open-source @ <https://github.com/ant4g0nist/chinstrap>
- Written in Python
- Currently at version 1.1.2
- Clear documentation: [chinstrap.io/docs](https://chinstrap.io/docs)

# Usage

```
➔ ~ chinstrap -h
```

[illegible]

Docs 📖 : <https://chinstrap.io/>  
Tele 🗨 : [https://t.me/chinstrap\\_io](https://t.me/chinstrap_io)

🐼 **Chinstrap** - a cute framework for developing Tezos Smart Contracts!

```
usage: chinstrap [-h] {init,config,networks,compile,install,create,templates,test,sandbox,develop,originate} ...
```

positional arguments:

```
{init,config,networks,compile,install,create,templates,test,sandbox,develop,originate}
```

init	Initialize a new Chinstrap project
config	Verify Chinstrap configuration
networks	List currently available test networks
compile	Compile contract source files
install	Helper to install compilers
create	Helper to create new contracts, originations and tests
templates	Download templates provided by SmartPy and *LIGO
test	Run pytest/smartpy/ligo tests
sandbox	Start a Tezos local sandbox
develop	Open an interactive console for Tezos
originate	Run originations and deploy contracts

optional arguments:

```
-h, --help    show this help message and exit
```


chinstrap

**crash course**

# Prerequisites

- Python  $\geq$  3.7, Docker, node.js
- On macOS
  - `brew tap cuber/homebrew-libsecp256k1`
  - `brew install libsodium libsecp256k1 gmp`
- On Ubuntu
  - `apt install libsodium-dev libsecp256k1-dev libgmp-dev pkg-config`
- Installation:
  - `pip3 install chinstrap`

# Install compilers 🍷

- Make sure docker is running 
- *chinstrap install* installs Ligo and SmartPy compilers

```
➔ ~ chinstrap install  
🎉 Ligo installed  
🎉 SmartPy installed
```



## Init Chinstrap project 🐧

- `chinstrap init` quickly sets up a new Chinstrap project
- Create and enter a chinstrap project directory:

```
→ ~ mkdir chinstrap-hello-world  
→ ~ cd chinstrap-hello-world  
→ chinstrap-hello-world █
```

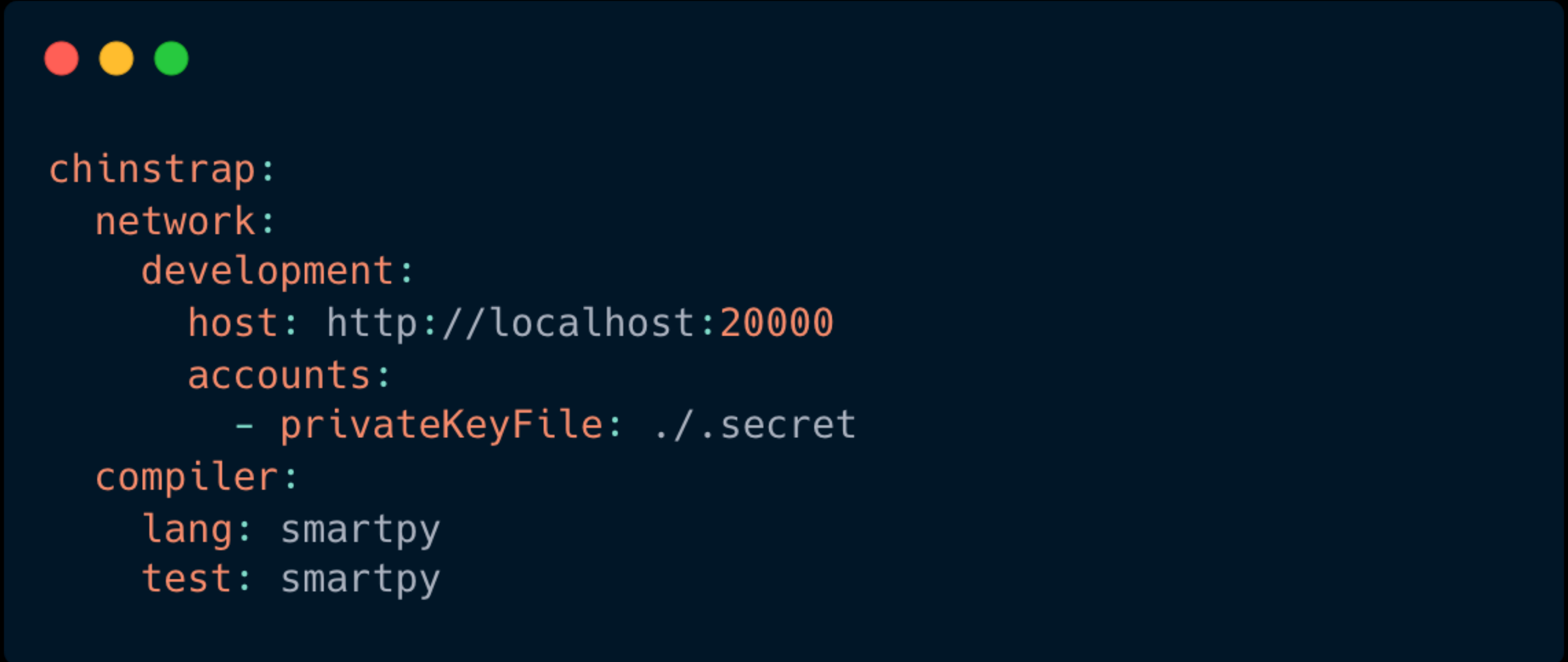
- Initialise your project:
  - > `chinstrap init -s`
- > `-s` flag initialises with samples for contract, test and deployments

# Chinstrap project structure

- **contracts**: the folder containing all the LIGO/SmartPy smart contracts that Chinstrap has to compile.
- **originations**: the folder containing the Chinstrap origination/deployment scripts for the deployment of the contracts.
- **test**: the folder containing Javascript tests
- **chinstrap-config.yaml**: the configuration file which defines networks and accounts to be used for the deployment.

# Chinstrap config

- A minimal configuration file looks like this:



```
chinstrap:
  network:
    development:
      host: http://localhost:20000
      accounts:
        - privateKeyFile: ../.secret
  compiler:
    lang: smartpy
    test: smartpy
```

- Supported options for lang: *smartpy, cameligo, pasceligo, reasonligo, jsligo*
- Supported options for test: *pytest, smartpy, cameligo, pasceligo, reasonligo, jsligo*

# Contract development

- Sample SmartPy contract:
- Compilation: ***chinstrap compile***
- Testing : ***chinstrap test***
- Origination : ***chinstrap originate***

```
import smartpy as sp

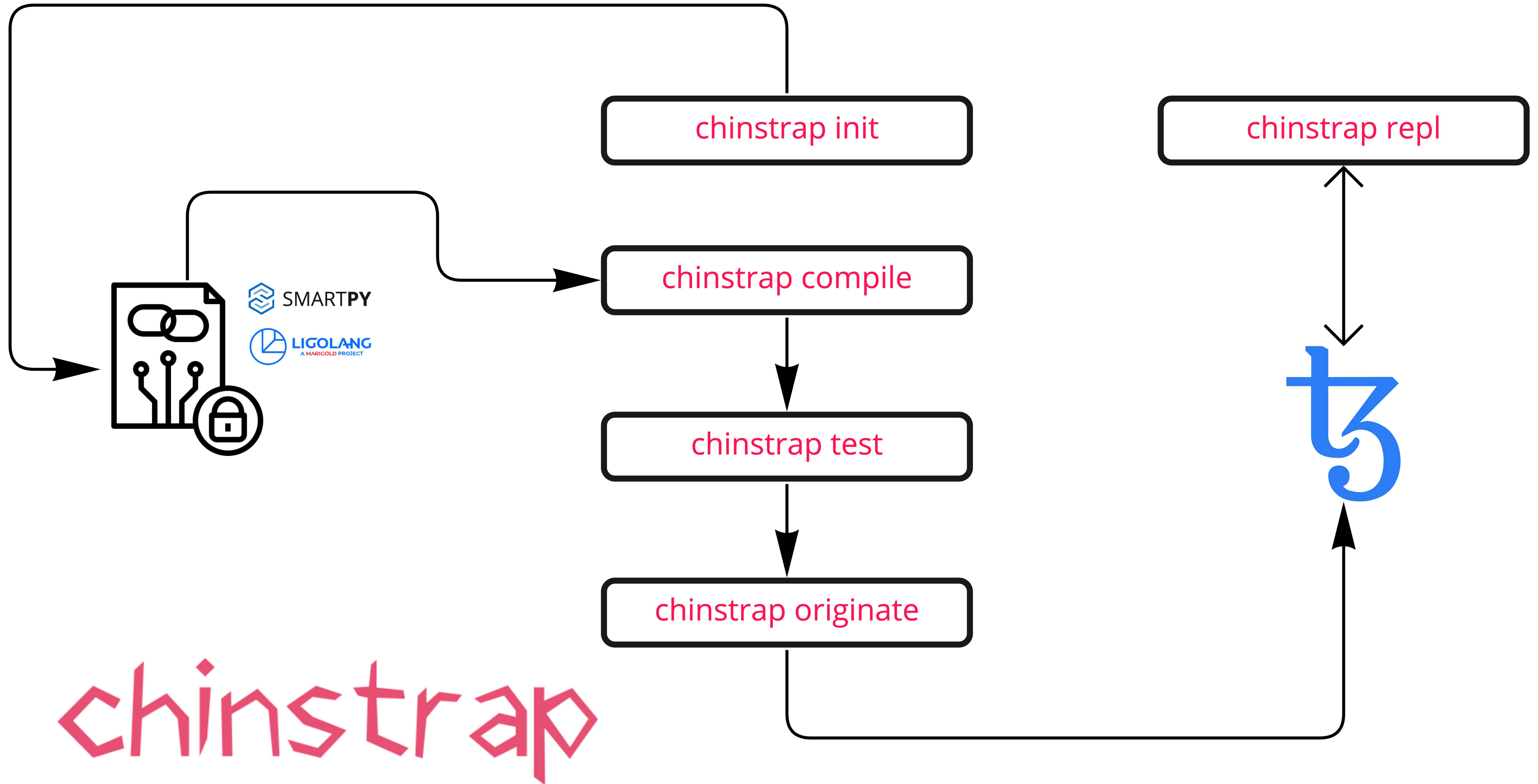
class SampleContract(sp.Contract):
    def __init__(self, value, owner):
        self.init_type(sp.TRecord(counter=sp.TInt,
owner=sp.TAddress))
        self.init(counter=value, owner=owner)

    @sp.entry_point
    def increment(self, value):
        sp.verify(sp.sender == self.data.owner, message="Only
owner can increment")
        self.data.counter += value

    @sp.entry_point
    def decrement(self, value):
        sp.verify(sp.sender == self.data.owner, message="Only
owner can decrement")
        self.data.counter -= value

sp.add_compilation_target(
    "SampleContract",
    SampleContract(0,
sp.address("tz1a9GCc4UU6d5Z9spyozgKTARngb8DZKbNe")),
)
```

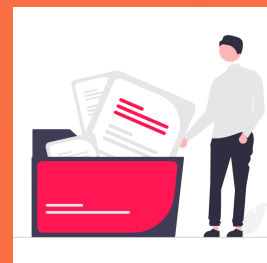
DEMO



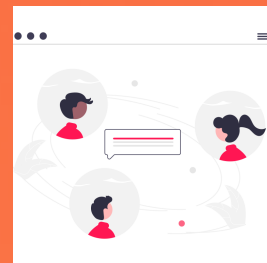
# References



[https://twitter.com/chinstrap\\_io](https://twitter.com/chinstrap_io)



[chinstrap.io](https://chinstrap.io)



[https://t.me/chinstrap\\_io](https://t.me/chinstrap_io)

Happy hacking :)

