Faucet on ZOF

OpenFlow as a Micro-Service

Bill Fisher
william.w.fisher@gmail.com

What is ZOF?

- Python OpenFlow framework
- MIT License
- Alternative to RYU
- Supports OpenFlow only

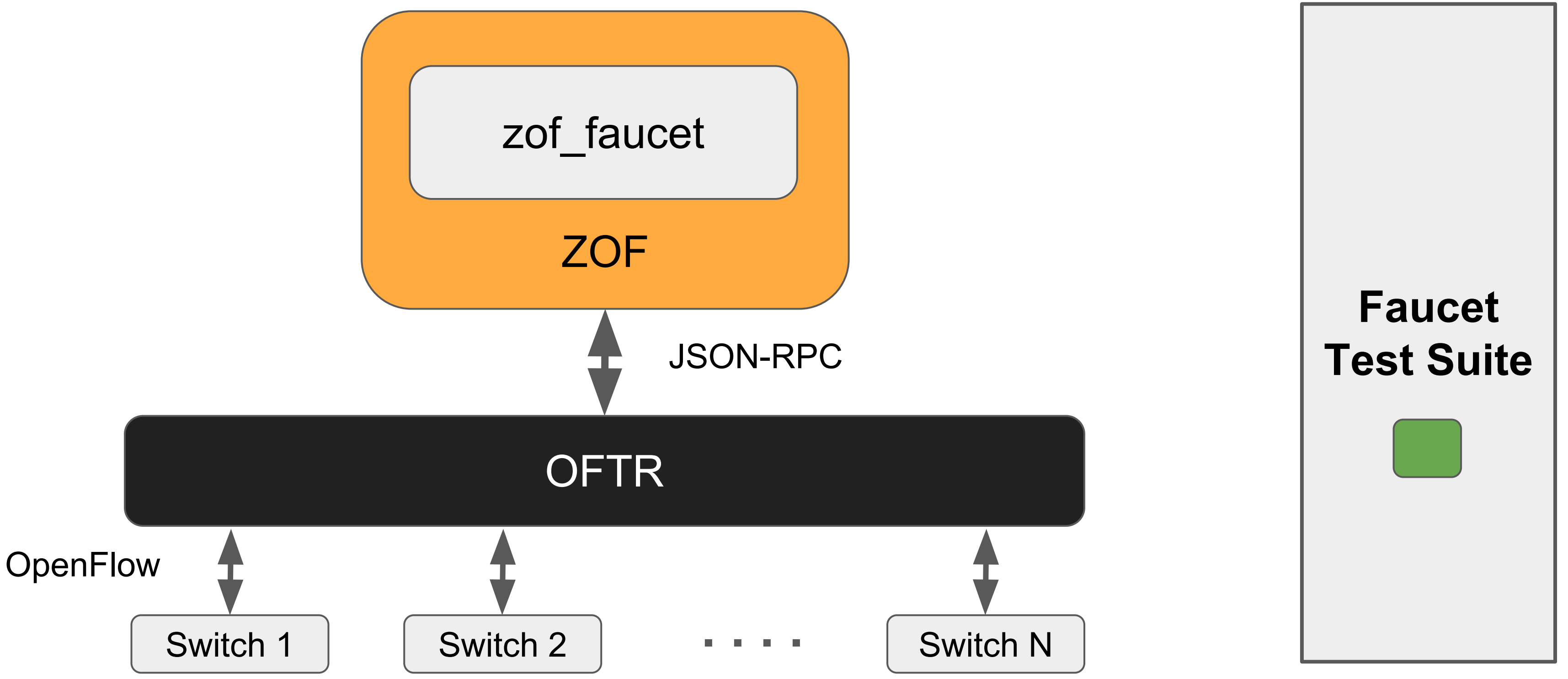
https://github.com/byllyfish/zof **Python**

https://github.com/byllyfish/oftr **C++**

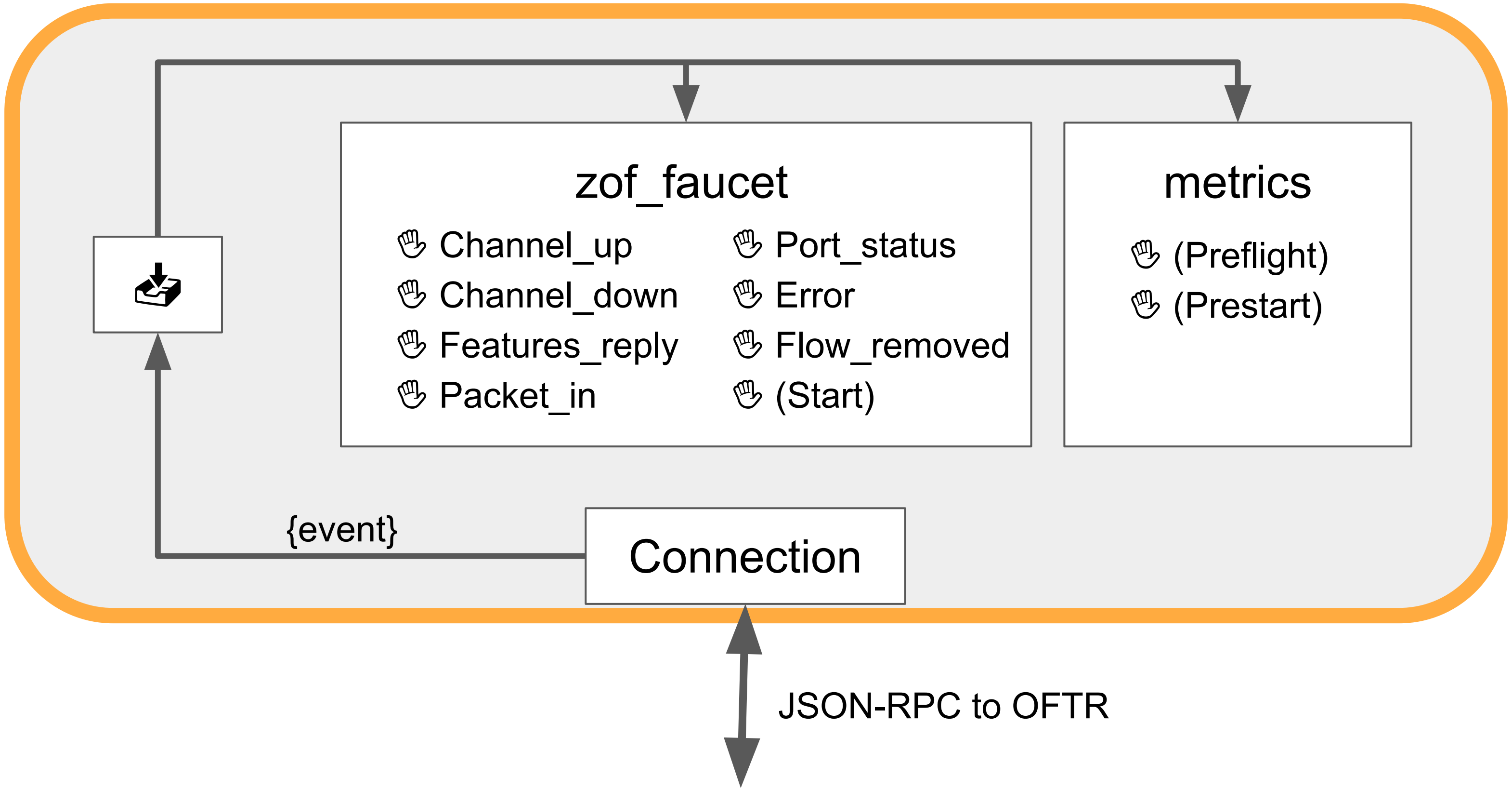
ZOF Data Sheet

- Event-driven “Apps”
- OpenFlow 1.0 - 1.4 (Some of 1.5)
- TLS 1.0 - 1.3
- Limited packet parsing and generation. BYOPG
- Supports both sides of OpenFlow protocol
- **Asyncio (async/await)**
- **JSON-RPC Micro-service (C++)**
- **Declarative syntax for OpenFlow messages (YAML)**

Faucet on ZOF

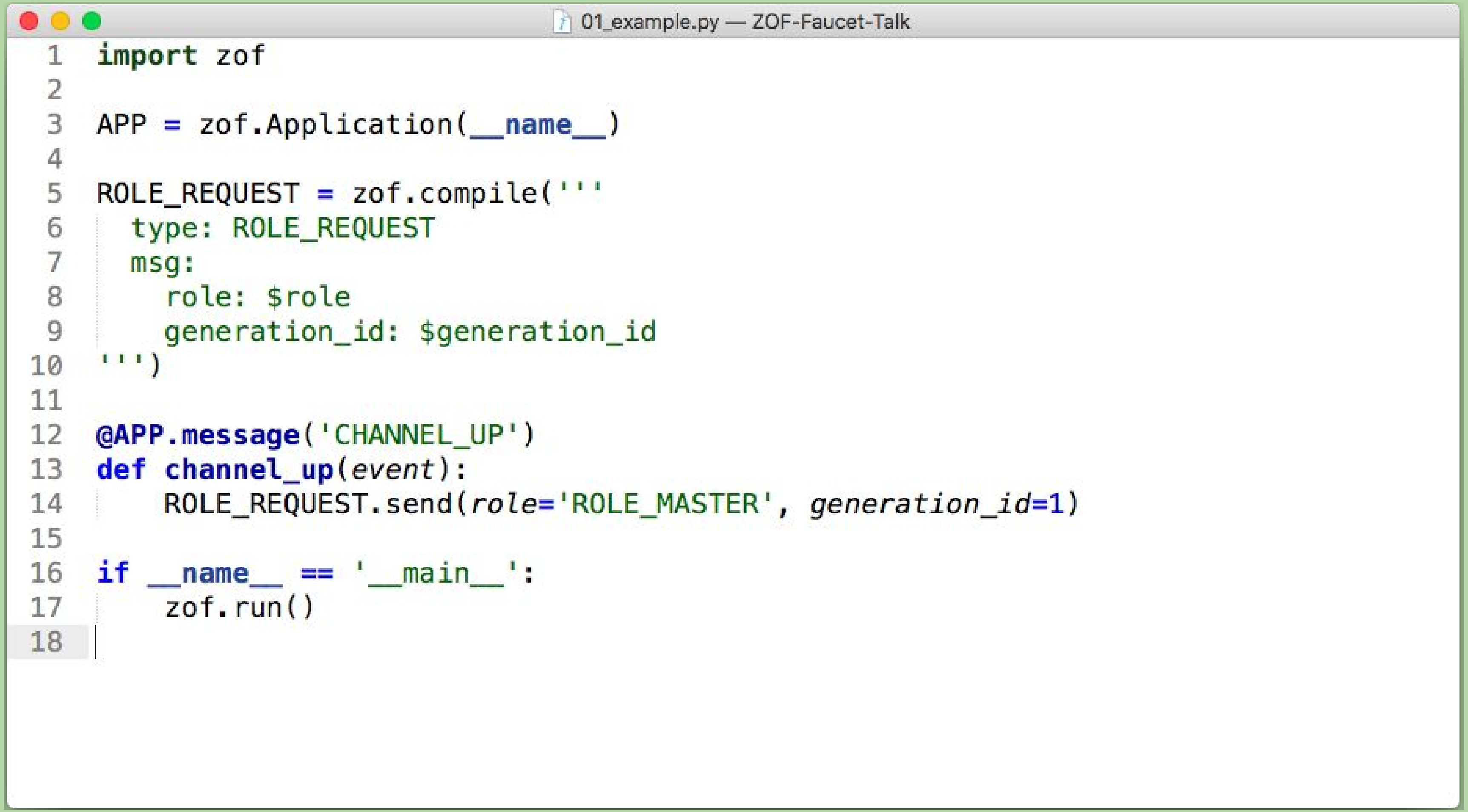


ZOF (Inside the Python process)

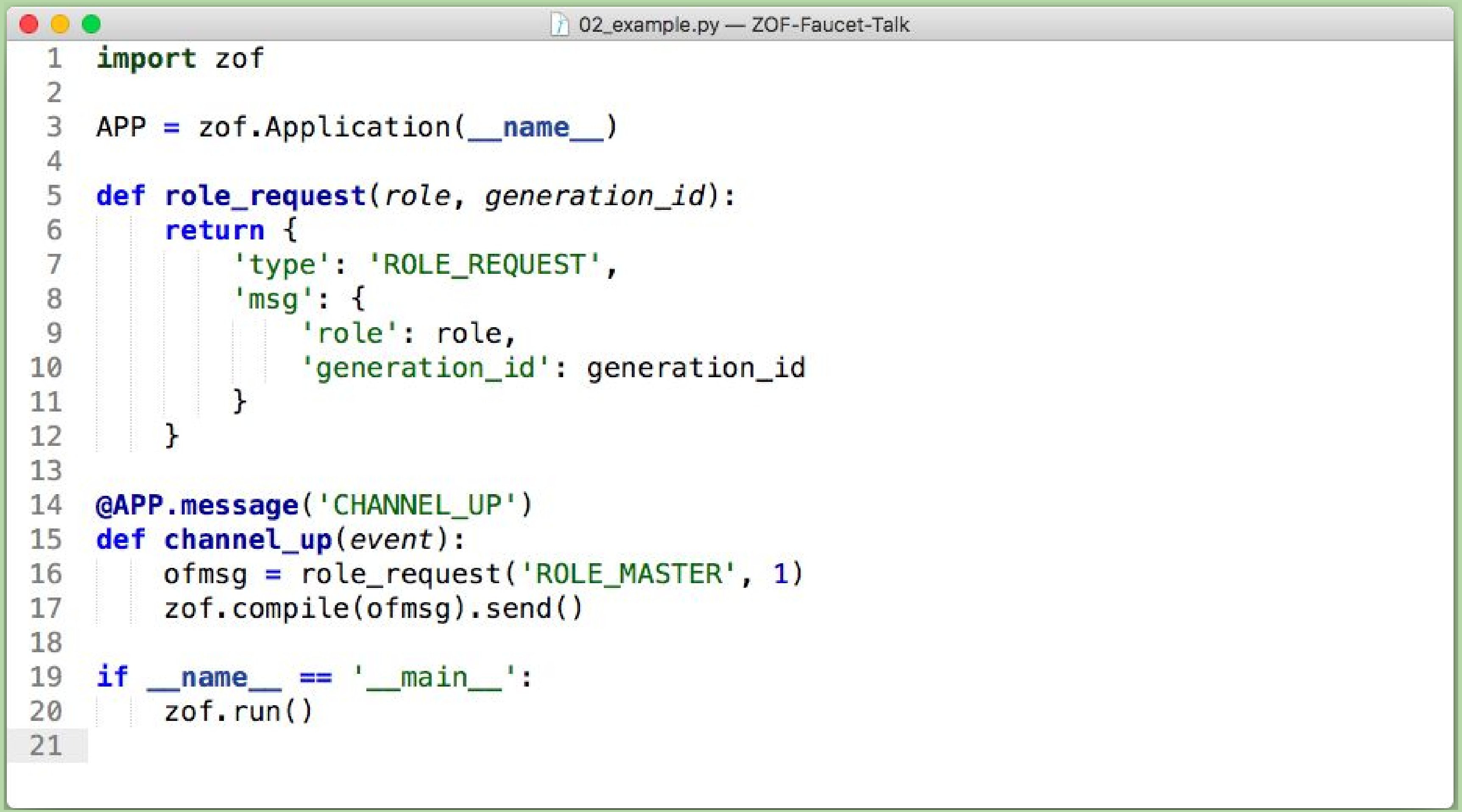


App Lifecycle Phases and Events

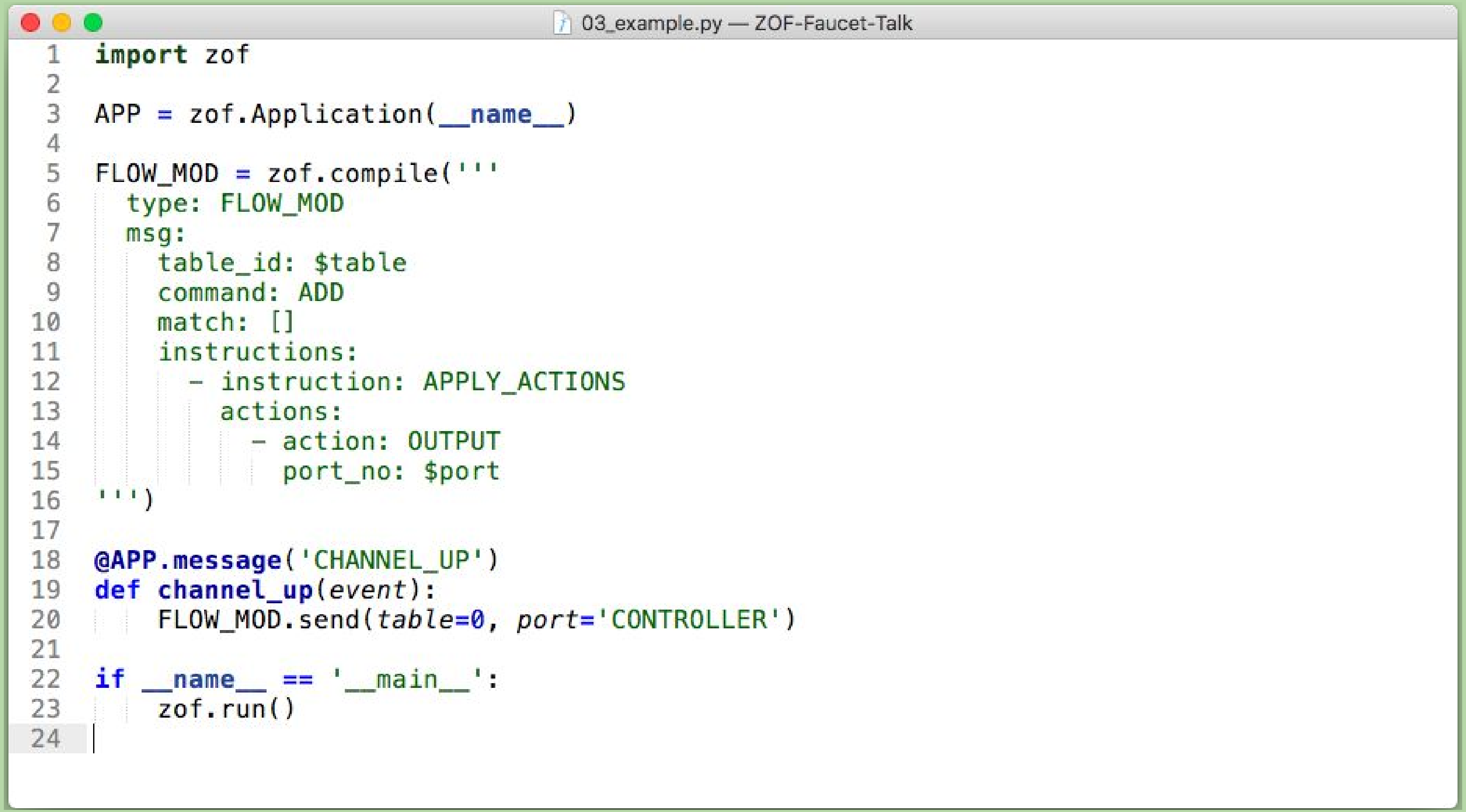
- INIT
- PREFLIGHT
- PRESTART (async)
- START (async)
- STOP (async)
- POSTFLIGHT


01_example.py — ZOF-Faucet-Talk

```python
import zof

APP = zof.Application(__name__)

ROLE_REQUEST = zof.compile('''
  type: ROLE_REQUEST
  msg:
    role: $role
    generation_id: $generation_id
''')

@APP.message('CHANNEL_UP')
def channel_up(event):
    ROLE_REQUEST.send(role='ROLE_MASTER', generation_id=1)

if __name__ == '__main__':
    zof.run()

```

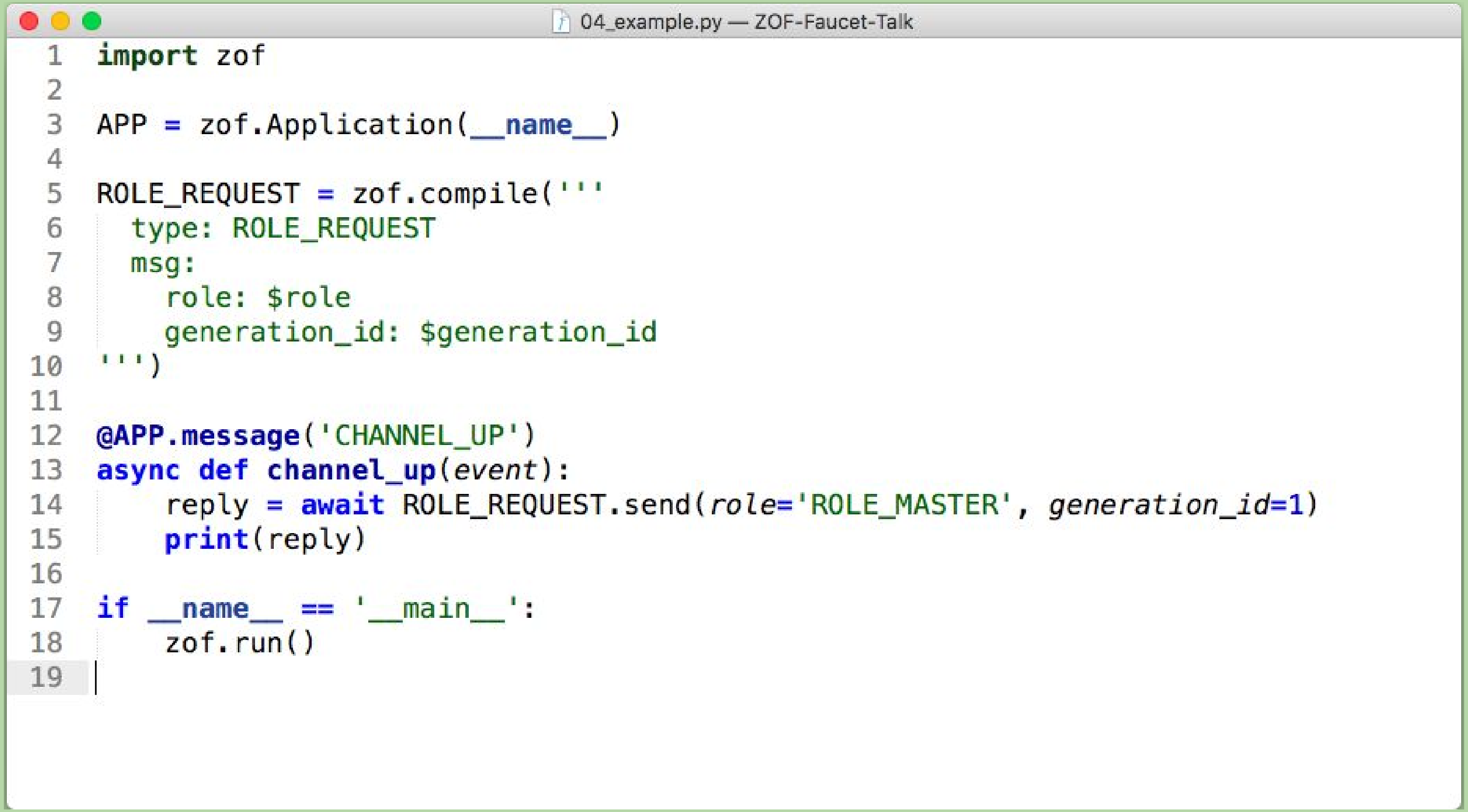


```python
import zof

APP = zof.Application(__name__)

def role_request(role, generation_id):
    return {
        'type': 'ROLE_REQUEST',
        'msg': {
            'role': role,
            'generation_id': generation_id
        }
    }

@APP.message('CHANNEL_UP')
def channel_up(event):
    ofmsg = role_request('ROLE_MASTER', 1)
    zof.compile(ofmsg).send()

if __name__ == '__main__':
    zof.run()
```

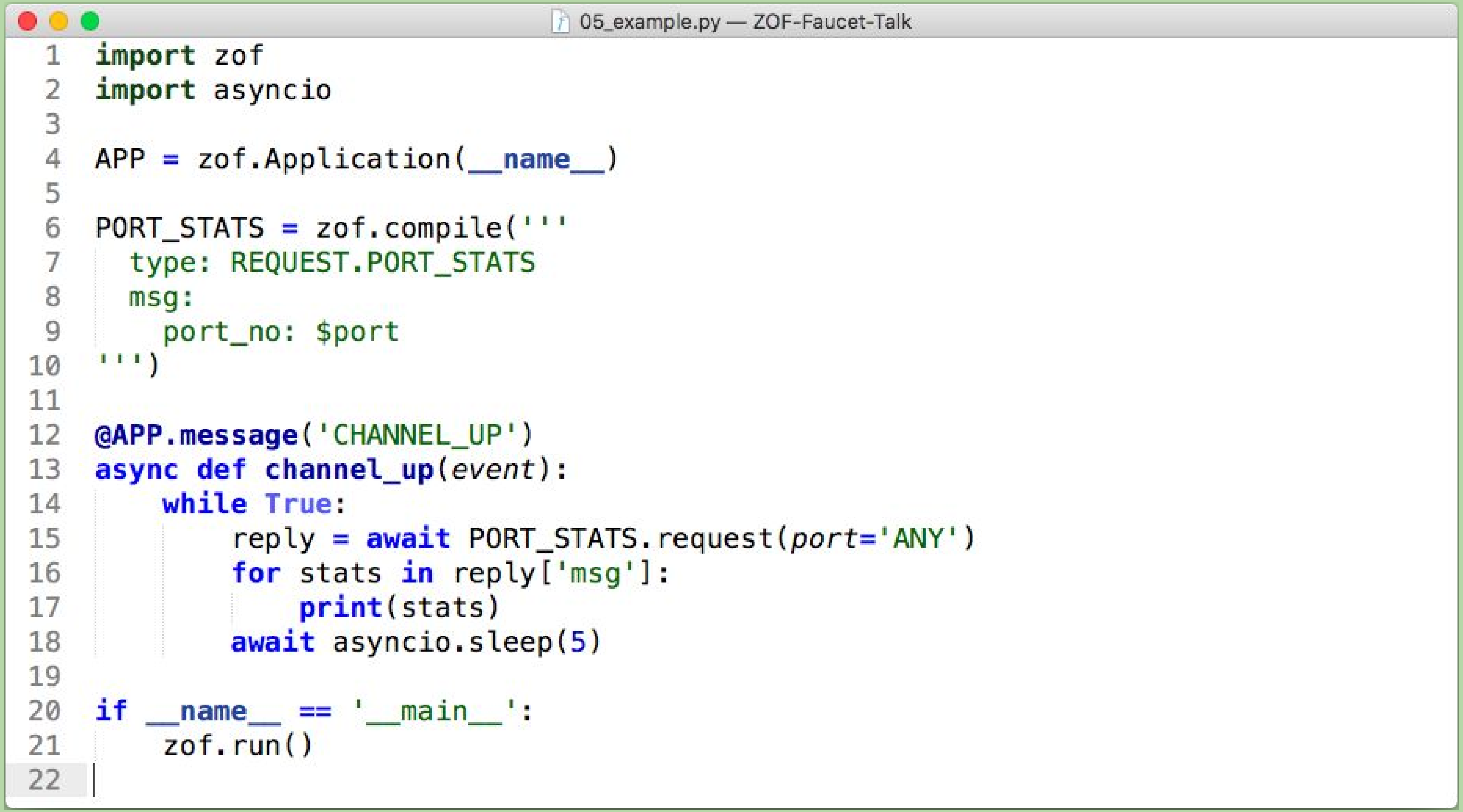


03_example.py — ZOF-Faucet-Talk

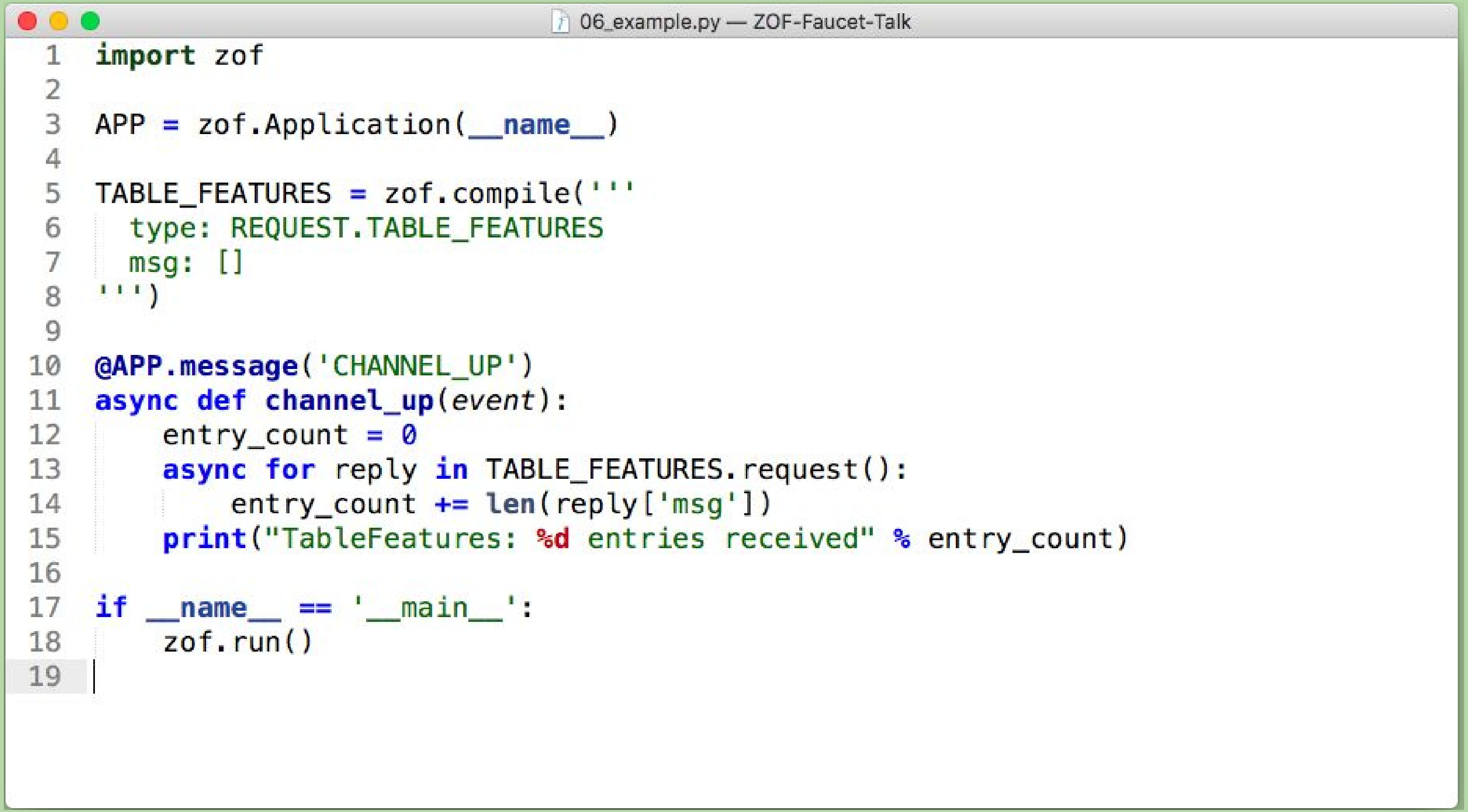
```python
import zof

APP = zof.Application(__name__)

FLOW_MOD = zof.compile('''
  type: FLOW_MOD
  msg:
    table_id: $table
    command: ADD
    match: []
    instructions:
      - instruction: APPLY_ACTIONS
        actions:
          - action: OUTPUT
            port_no: $port
''')

@APP.message('CHANNEL_UP')
def channel_up(event):
    FLOW_MOD.send(table=0, port='CONTROLLER')

if __name__ == '__main__':
    zof.run()

```


04_example.py — ZOF-Faucet-Talk

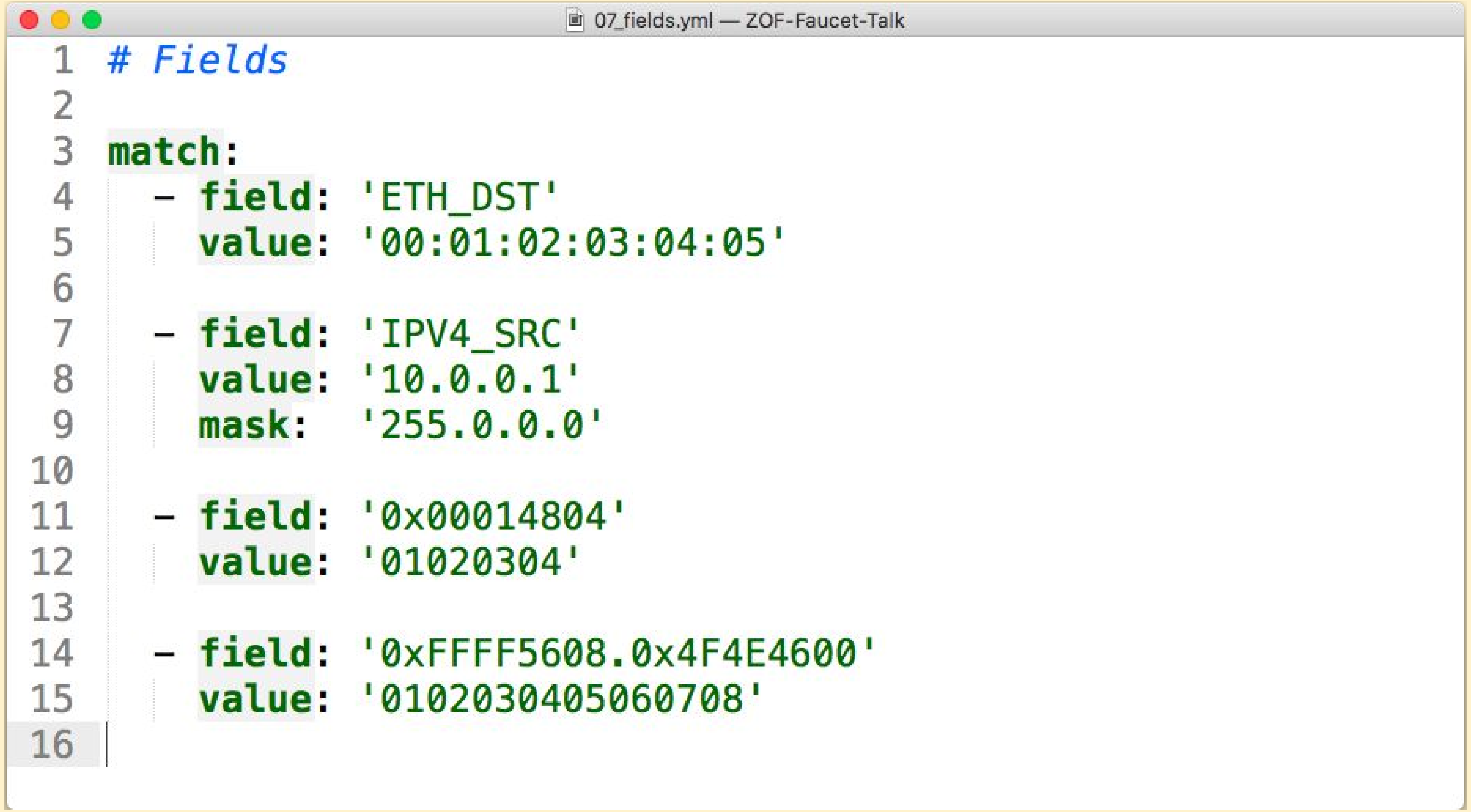
```python
import zof

APP = zof.Application(__name__)

ROLE_REQUEST = zof.compile('''
  type: ROLE_REQUEST
  msg:
    role: $role
    generation_id: $generation_id
''')

@APP.message('CHANNEL_UP')
async def channel_up(event):
    reply = await ROLE_REQUEST.send(role='ROLE_MASTER', generation_id=1)
    print(reply)

if __name__ == '__main__':
    zof.run()
```


05_example.py — ZOF-Faucet-Talk

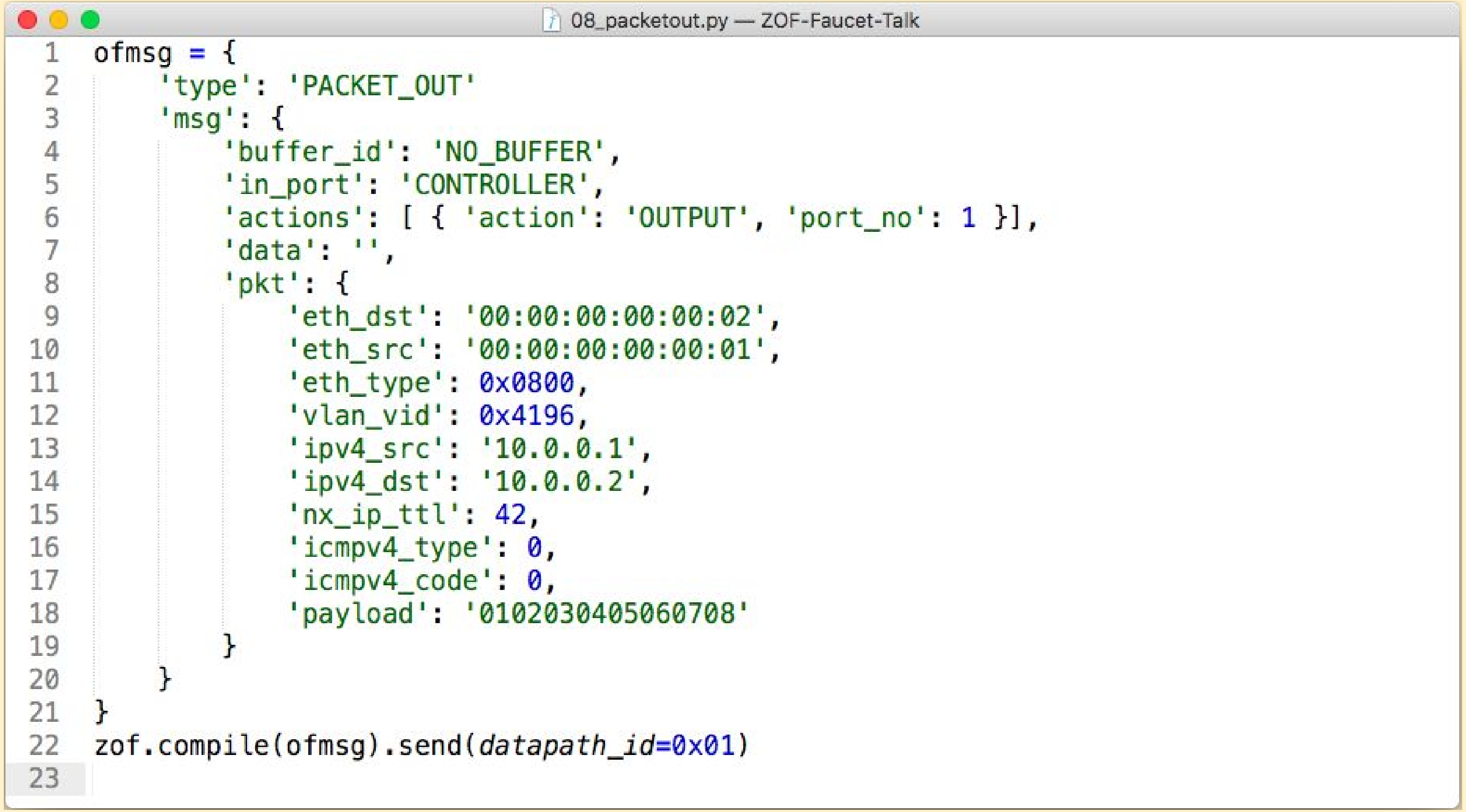
```python
import zof
import asyncio

APP = zof.Application(__name__)

PORT_STATS = zof.compile('''
  type: REQUEST.PORT_STATS
  msg:
    port_no: $port
''')

@APP.message('CHANNEL_UP')
async def channel_up(event):
    while True:
        reply = await PORT_STATS.request(port='ANY')
        for stats in reply['msg']:
            print(stats)
        await asyncio.sleep(5)

if __name__ == '__main__':
    zof.run()

```



```python
import zof

APP = zof.Application(__name__)

TABLE_FEATURES = zof.compile('''
  type: REQUEST.TABLE_FEATURES
  msg: []
''')

@APP.message('CHANNEL_UP')
async def channel_up(event):
    entry_count = 0
    async for reply in TABLE_FEATURES.request():
        entry_count += len(reply['msg'])
    print("TableFeatures: %d entries received" % entry_count)

if __name__ == '__main__':
    zof.run()

```

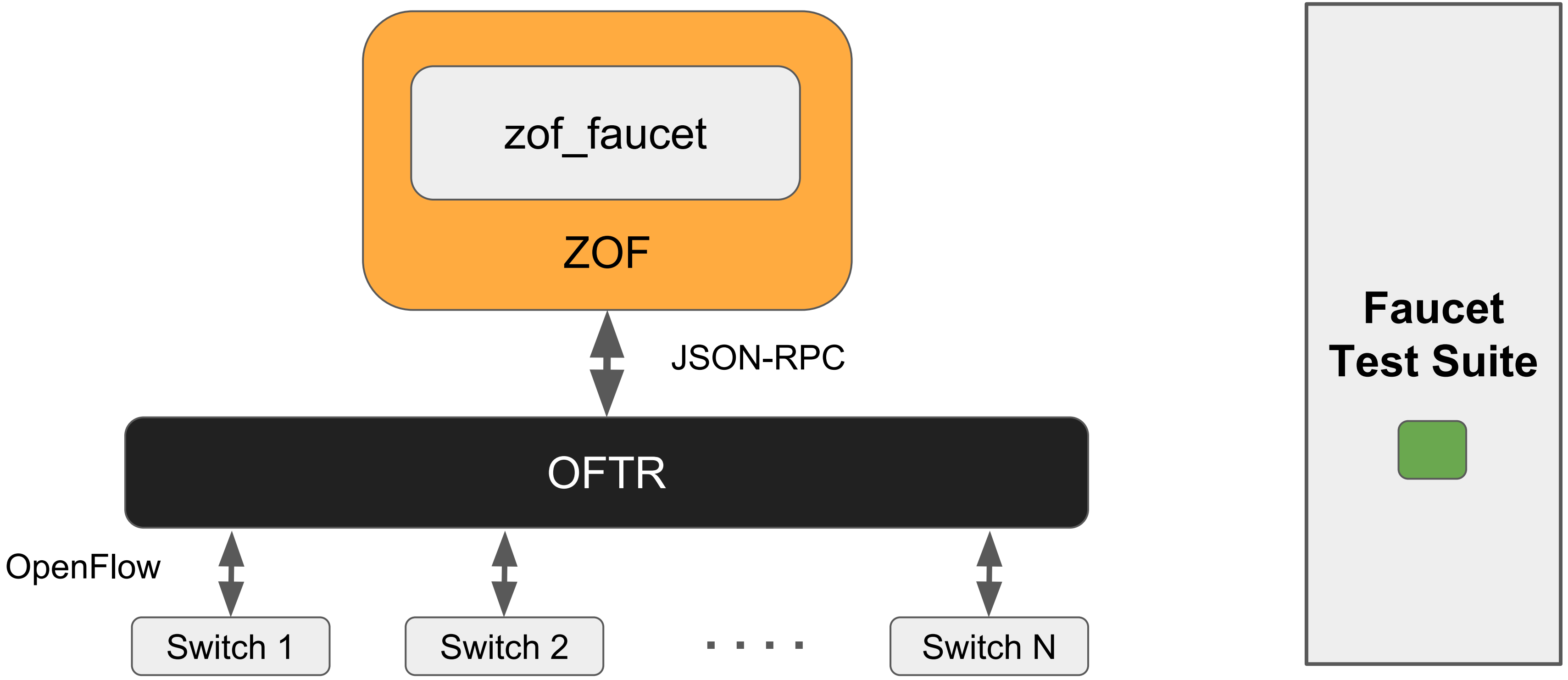


```yaml
# Fields

match:
  - field: 'ETH_DST'
    value: '00:01:02:03:04:05'

  - field: 'IPV4_SRC'
    value: '10.0.0.1'
    mask:  '255.0.0.0'

  - field: '0x00014804'
    value: '01020304'

  - field: '0xFFFF5608.0x4F4E4600'
    value: '0102030405060708'
```



```python
ofmsg = {
    'type': 'PACKET_OUT'
    'msg': {
        'buffer_id': 'NO_BUFFER',
        'in_port': 'CONTROLLER',
        'actions': [ { 'action': 'OUTPUT', 'port_no': 1 }],
        'data': '',
        'pkt': {
            'eth_dst': '00:00:00:00:00:02',
            'eth_src': '00:00:00:00:00:01',
            'eth_type': 0x0800,
            'vlan_vid': 0x4196,
            'ipv4_src': '10.0.0.1',
            'ipv4_dst': '10.0.0.2',
            'nx_ip_ttl': 42,
            'icmpv4_type': 0,
            'icmpv4_code': 0,
            'payload': '0102030405060708'
        }
    }
}
zof.compile(ofmsg).send(datapath_id=0x01)

```

OFTR Demo


zof_faucet
ZOF
JSON-RPC
OFTR
OpenFlow
Switch 1
Switch 2
. . . .
Switch N
Faucet Test Suite