

---

# **pytest-asyncio-network-simulator**

## **Documentation**

*Release 0.1.0-alpha.2*

**Ethereum Foundation**

**Apr 26, 2019**



---

## Contents

---

<b>1</b>	<b>Contents</b>	<b>3</b>
1.1	Quickstart . . . . .	3
1.1.1	Installation . . . . .	3
1.1.2	Patching <code>asyncio</code> . . . . .	3
1.2	Release Notes . . . . .	4
1.2.1	v0.1.0-alpha.1 . . . . .	4
<b>2</b>	<b>Indices and tables</b>	<b>5</b>



**Warning:** This project should be considered alpha quality software.

This library can be used to transparently bypass the networking component when testing `asyncio` applications. This is accomplished by monkeypatching various `asyncio` APIs to use locally connected stream readers and writers instead of ones connected via a network. The goal is for this to be seamless, requiring no code changes in your application and a minimal boilerplate in your test suite.



## 1.1 Quickstart

### 1.1.1 Installation

Install with pip

```
$ pip install pytest-asyncio-network-simulator
```

### 1.1.2 Patching `asyncio`

A pytest fixture is the easiest and quickest way to leverage this library. Place the following either in a specific test module, or in a `conftest.py` file.

```
import pytest

@pytest.fixture(autouse=True)
def network_sim(router):
    network = router.get_network(name='testing')
    with network.patch_asyncio():
        yield network
```

This will replace the following `asyncio` APIs with the patched versions.

- `asyncio.open_connection`
- `asyncio.start_server`

---

**Note:** The `router` fixture used in the example above is provided by this library by default.

---

---

**Note:** You can drop the `autouse=True` part from the fixture definition if you want to selectively include the fixture in your tests.

---

---

**Note:** The `name='testing'` is arbitrary. Any name will do.

---

## 1.2 Release Notes

### 1.2.1 v0.1.0-alpha.1

- Launched repository, claimed names for pip, RTD, github, etc



## CHAPTER 2

---

### Indices and tables

---

- `genindex`
- `modindex`