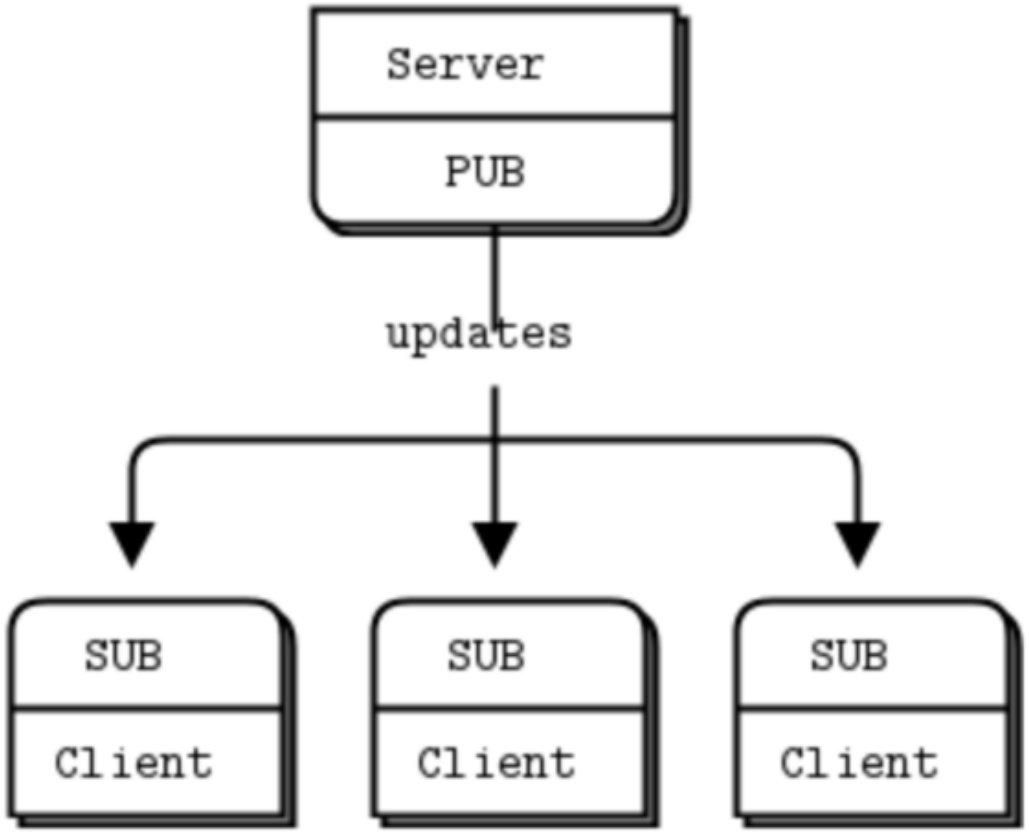
WebSocket 기반의 스케일 아웃 고려한 Real-time Pub/Sub BaaS

김광호 남윤지 이민철

PubSub



Stack

- API Server
  - Python
  - Sanic
  - ZeroMQ
  - Redis
- Admin Server
  - Python
  - Django
  - MySQL

ZeroMQ

분산 메시징

ZeroMQ \zero-em-queue\, \ØMQ\:

- Ø 코드를 모든 언어, 모든 플랫폼 상에서 연결
- Ø inproc, IPC, TCP, TIPC, 멀티캐스트를 통해 메시지들을 전달
- Ø pub-sub, push-pull, 그리고 router-dealer와 같은 영리한 패턴들
- Ø 아주 작은 라이브러리에 고속 비동기 I/O 엔진
- Ø 대규모의 활동적인 오픈 소스 커뮤니티 지원
- Ø 모든 현대 언어와 플랫폼을 지원
- Ø 모든 아키텍처를 구성: 중앙 집중, 분산, 소규모 혹은 대형
- Ø 완전 상업적 지원을 동반하는 무료 소프트웨어

ZeroMQ

Server:

```
#
#   Hello World server in Python
#   Binds REP socket to tcp://*:5555
#   Expects b"Hello" from client, replies with b"World"
#

import time
import zmq

context = zmq.Context()
socket = context.socket(zmq.REP)
socket.bind("tcp://*:5555")

while True:
    #  Wait for next request from client
    message = socket.recv()
    print("Received request: %s" % message)

    #  Do some 'work'
    time.sleep(1)

    #  Send reply back to client
    socket.send(b"World")
```

Client:

```
#
#   Hello World client in Python
#   Connects REQ socket to tcp://localhost:5555
#   Sends "Hello" to server, expects "World" back
#

import zmq

context = zmq.Context()

#  Socket to talk to server
print("Connecting to hello world server…")
socket = context.socket(zmq.REQ)
socket.connect("tcp://localhost:5555")

#  Do 10 requests, waiting each time for a response
for request in range(10):
    print("Sending request %s …" % request)
    socket.send(b"Hello")

    #  Get the reply.
    message = socket.recv()
    print("Received reply %s [ %s ]" % (request, message))
```

ZeroMQ PUBSUB Library

- ZeroMQ 프로토콜을 이용한 Scalable PubSub Server와 Client
- Connection 정보를 Redis에 저장 및 동기화

example/server.py

```python
import asyncio

from zmq_pubsub import PubSubServer

if __name__ == '__main__':
    async def main():
        server = await PubSubServer.create('redis://127.0.0.1')
        asyncio.get_event_loop().create_task(server.run_forever())

        while True:
            await asyncio.sleep(1)
            await server.publish('topic', 'header', {'msg': f'hello'})


    asyncio.get_event_loop().run_until_complete(main())
```

example/client.py

```
import asyncio

from zmq_pubsub import PubSubClient

if __name__ == '__main__':
    async def main():
        client = await PubSubClient.create('redis://127.0.0.1')

        client.subscribe('topic')

        async for msg in client.read_iter():
            print(msg)
            if msg.header == 'quit':
                break


    asyncio.get_event_loop().run_until_complete(main())
```

API Server

- HTTP
  - 특정 채널로 Publish
  - 각 채널의 이름, 연결된 Subscriber 수, Publish Per Minute 를 포함하는 채널 리스트 가져오기
- WebSocket
  - 각 채널의 Websocket endpoint connect하여 subscribe
  - 특정 채널로 Publish

API Server

WS

Channel Event

```
http://{hostname}/channel/{channel_name}/
```

- subscribe
- channel data set redis
- websocket send & receive response:

```
{
    "header":"exchange",
    "body": {"msg":"ok"}
}
```

HTTP

Channel List(GET)

```
http://{hostname}/v1/channel
```

response :

```
{
    "data": [
        {
            "channel_name": "a_channel",
            "cnt": 3,
            "rpm": 0
        }
    ]
}
```

Publish(POST)

```
http://{hostname}/v1/channel/<channel_name>/publish
```

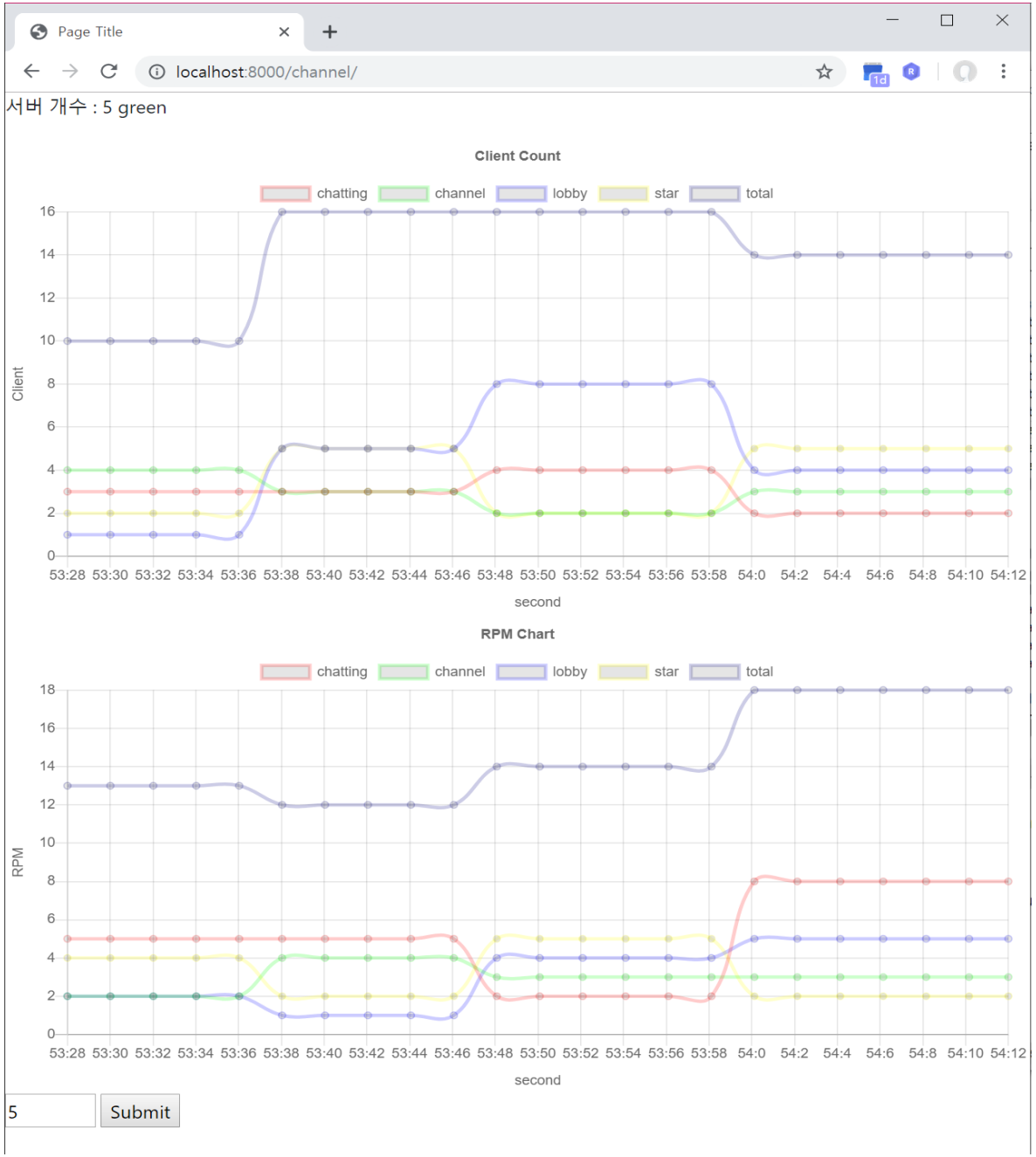
response:

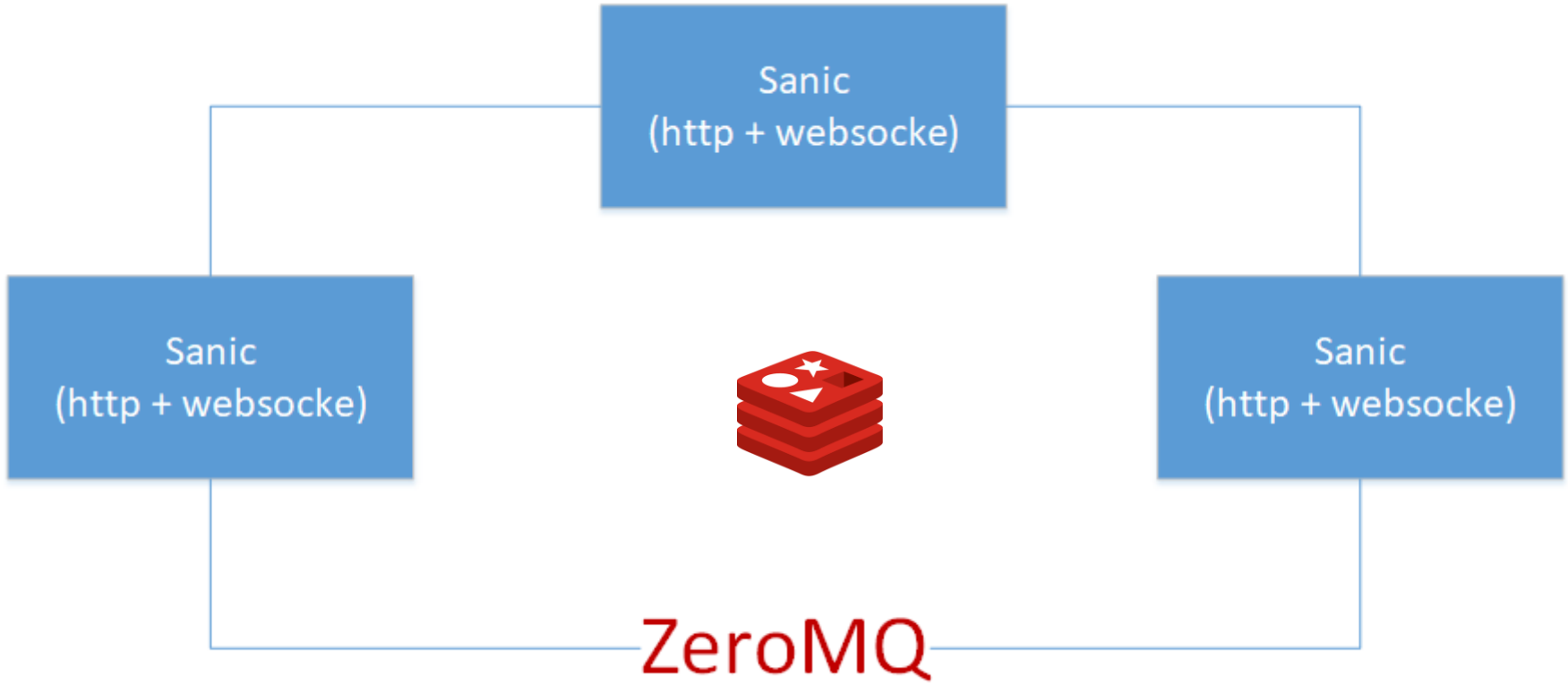
```
{
    "status": "ok"
}
```

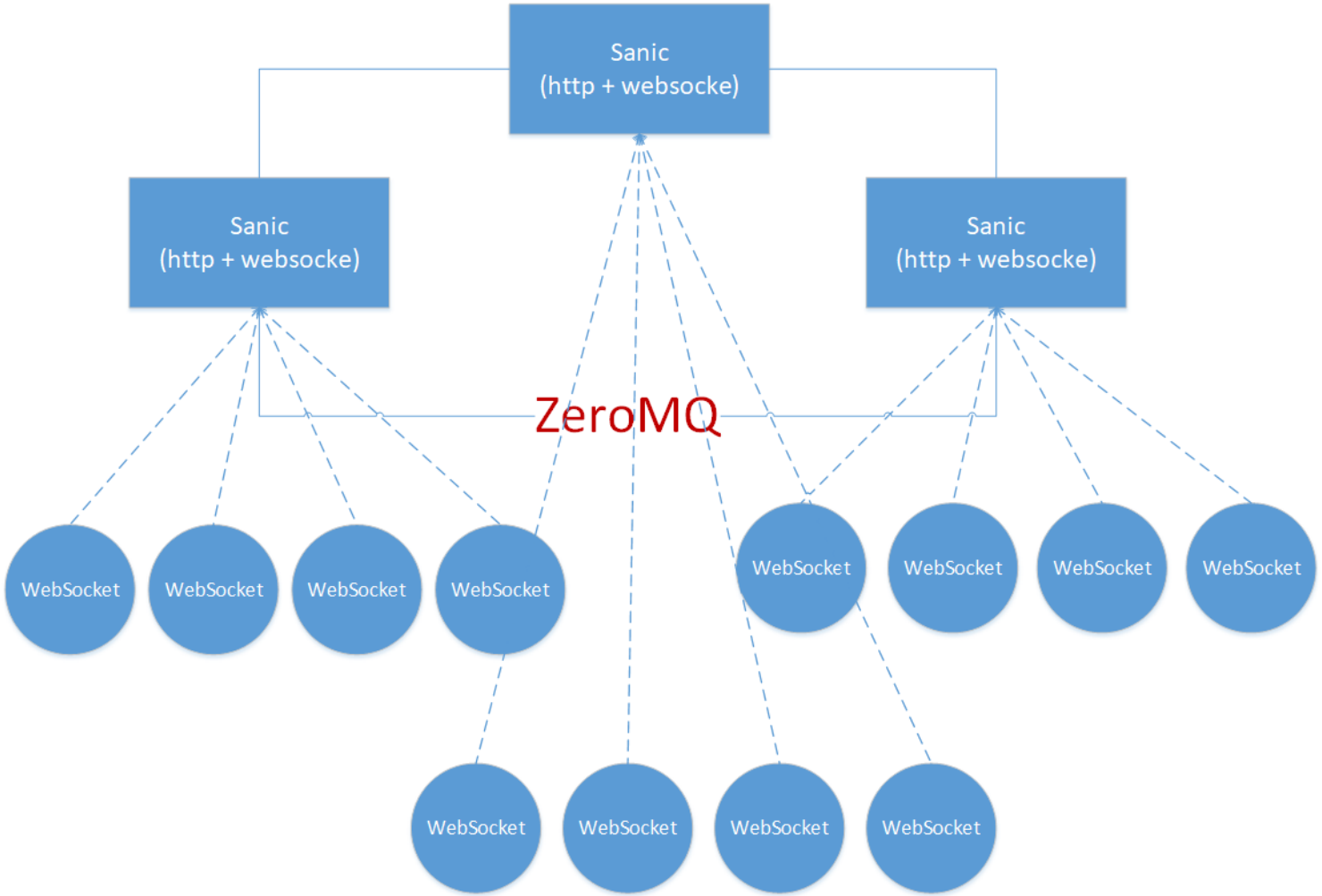
Admin Server

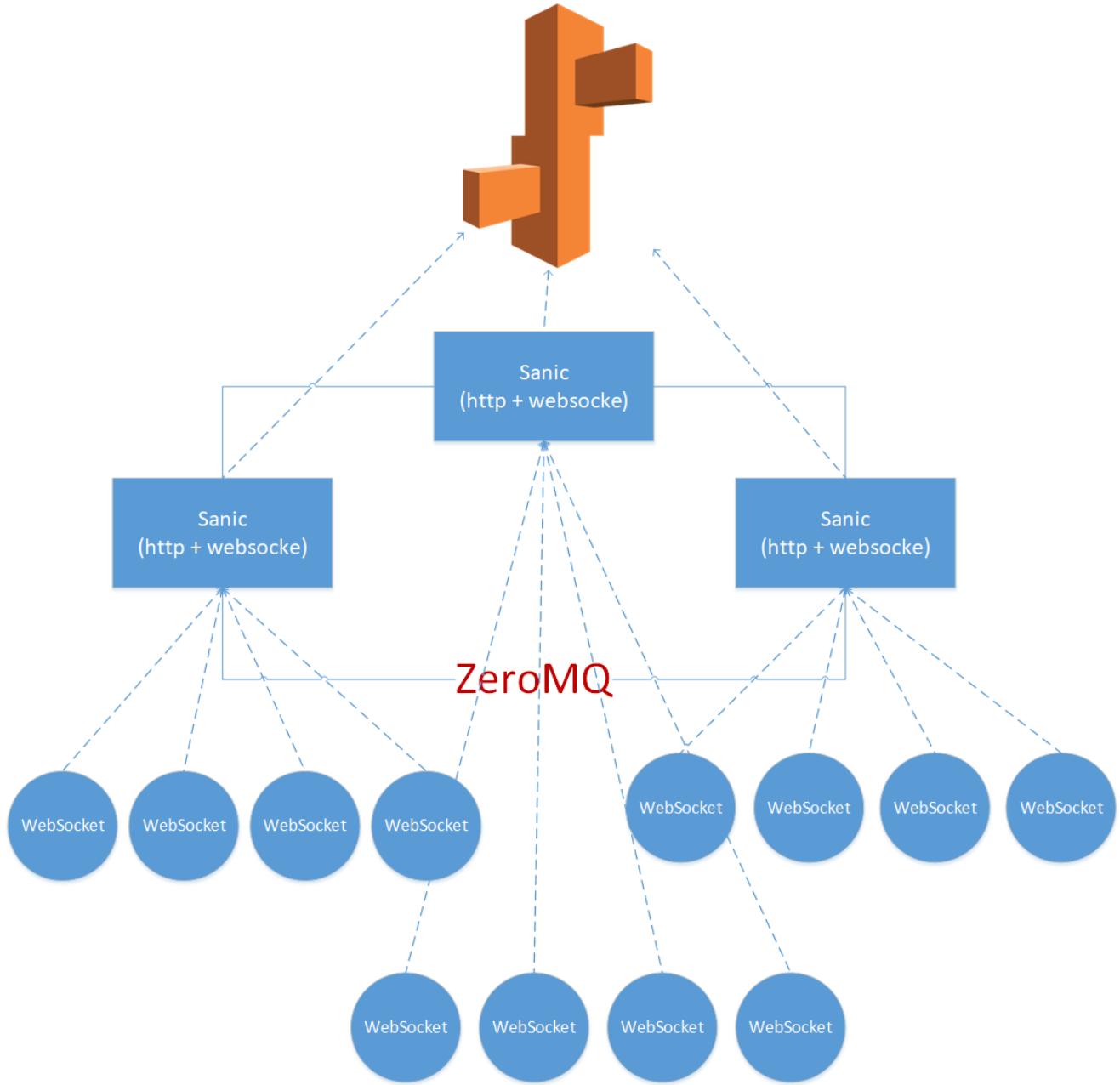
- 서버 상태 체크 (Green/Yellow/Red) 표시
- 서버의 수와 RPS를 표현하는 실시간 차트 표시
- 수동 스케일 인/아웃 할 수 있는 기능

Admin Server




Sanic
(http + websocke)
Sanic
(http + websocke)
Sanic
(http + websocke)
ZeroMQ


Sanic
(http + websocke)
Sanic
(http + websocke)
Sanic
(http + websocke)
ZeroMQ
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket


Sanic
(http + websocke)
Sanic
(http + websocke)
Sanic
(http + websocke)
ZeroMQ
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket
WebSocket

시연

Github

- https://github.com/amathon-2019/real-time-ws-pubsub-baas-api
- https://github.com/amathon-2019/real-time-ws-pubsub-baas-zmqpubsub-lib
- https://github.com/amathon-2019/real-time-ws-pubsub-baas-admin