# THE ELK STACK

## SANTIAGO JAVIER CALCAGNO

## KARLSRUHE INSTITUTE OF TECHNOLOGY

### SEMINAR „BIG DATA TOOLS“

# WHAT IS THE ~~ELK~~ ELASTIC STACK?[1]

**LOGSTASH**

Data processing

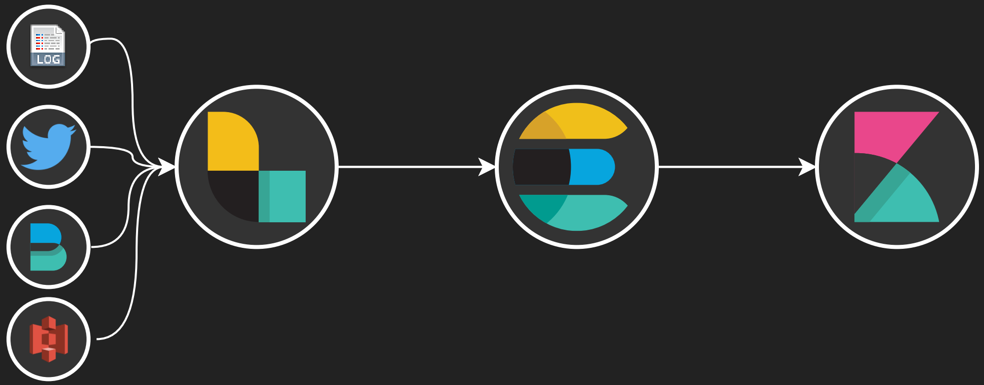
**ELASTICSEARCH**

Search engine

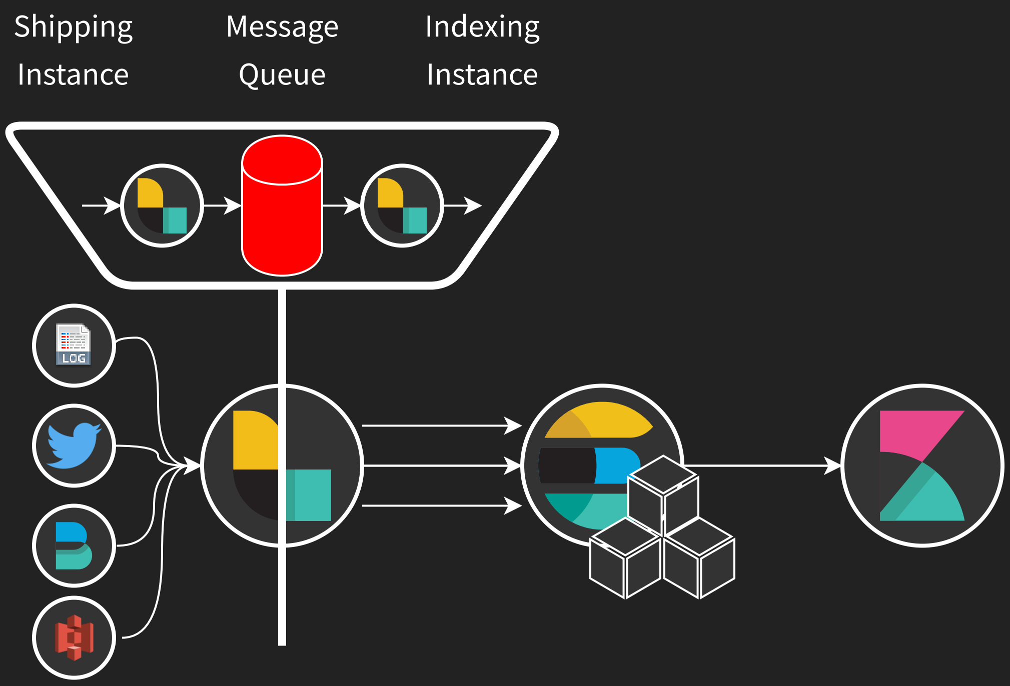
**KIBANA**

Visualization tool

# GENERAL ARCHITECTURE



# SCALING THE STACK$^2$



Shipping Instance

Message Queue

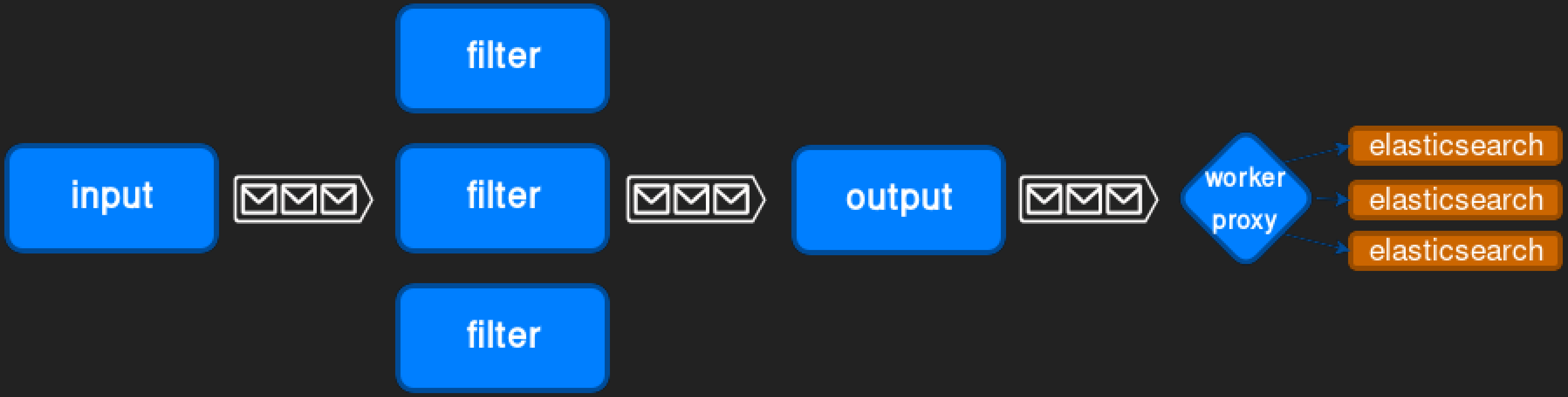
Indexing Instance

# USE CASES

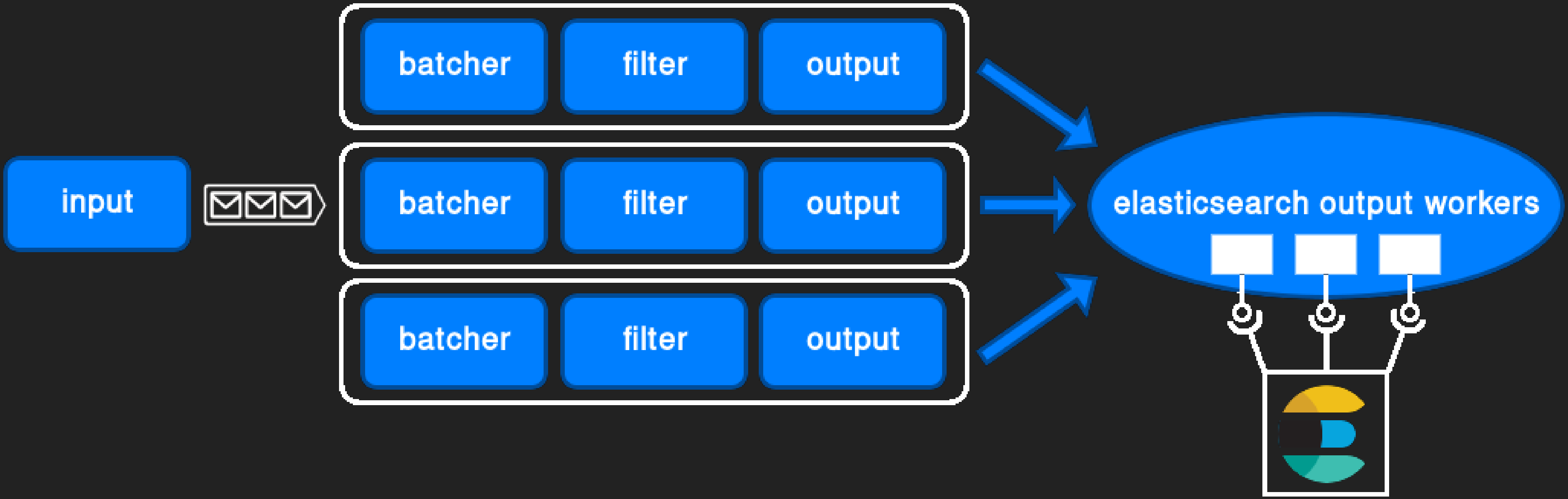
More use cases here.[3]

Adobe
salesforce
XING
CISCO
NETFLIX
ebay
Microsoft
Goldman Sachs
mercado Libre
SOUNDCLOUD
IBM

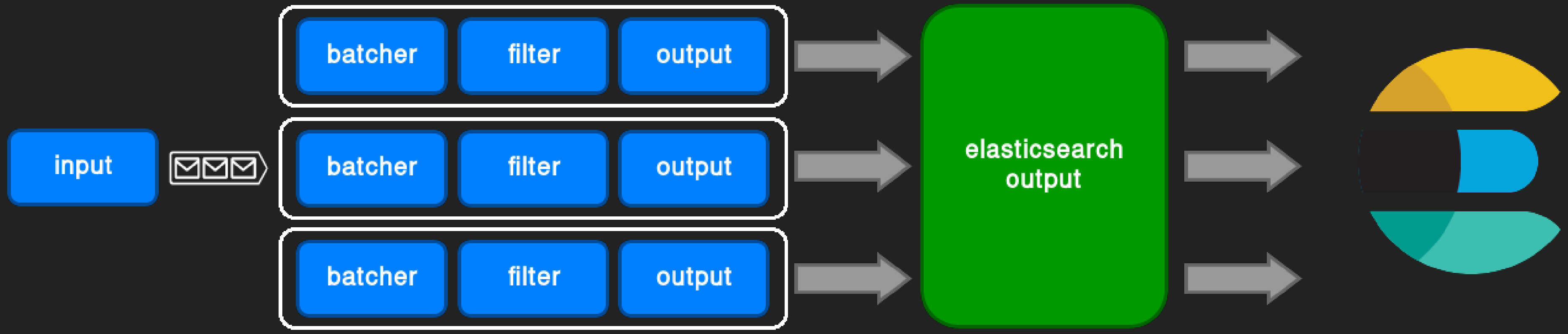
# EVOLUTION OF THE LOGSTASH PIPELINE[4]



Logstash pipeline, versions 1.2.2 to 2.1



Logstash pipeline, version 2.2



Logstash pipeline, versions 5.0 and newer

# GOAL

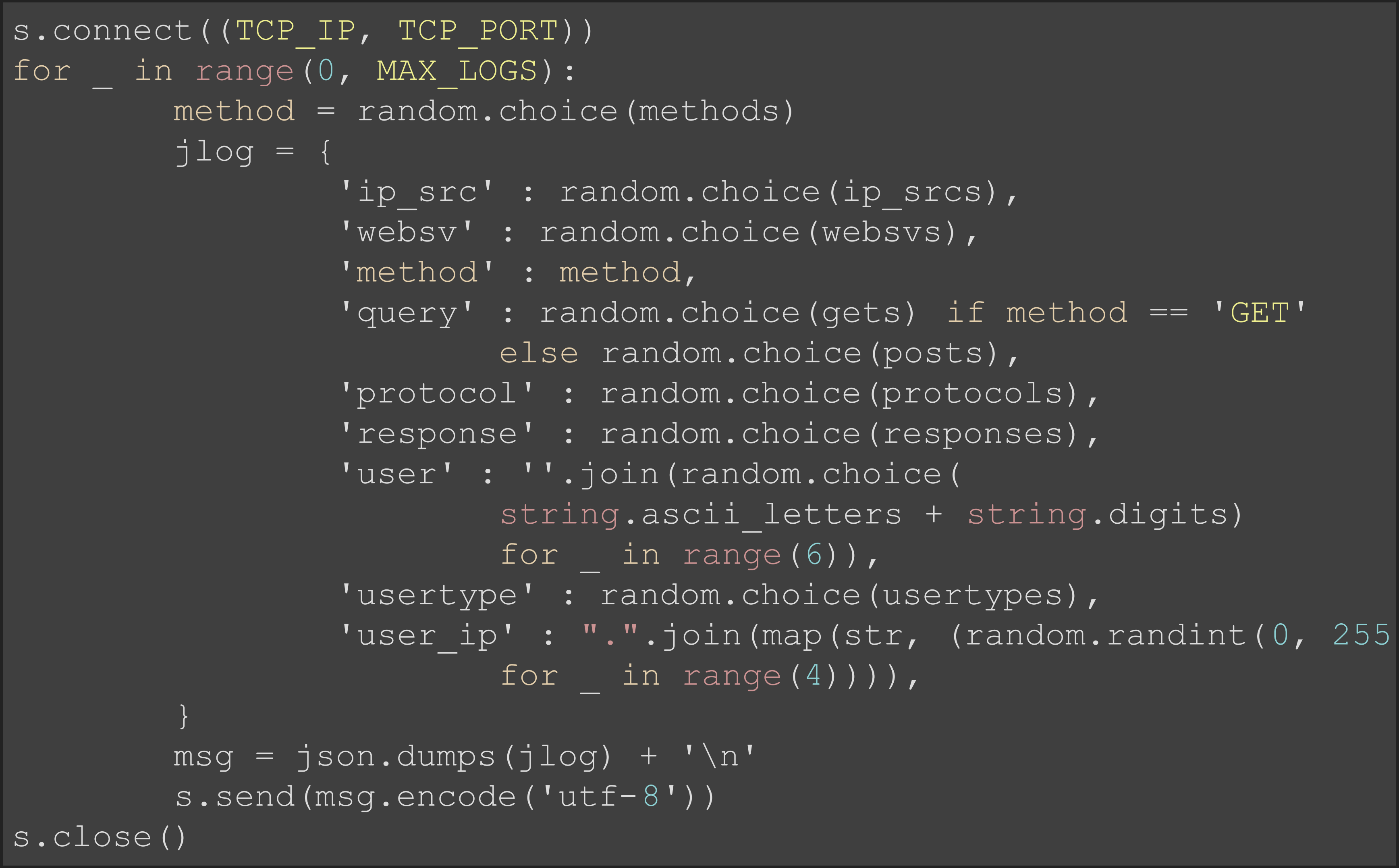
Analyze how the number of pipeline workers and the batch size affect the indexing rate (in a specific system).

- Intel® Core™ i5-2520M
- 16GB RAM DDR3-1866
- Samsung® EVO™ 250GiB mSATA SSD
- Arch Linux, kernel 4.8.13-1
- Elastic stack version 5.1

# TESTS AND TOOLS

```
for w in "${W_VALUES[@]}"
do
        for b in "${B_VALUES[@]}"
        do
                sed -i -e "s/-w [0-9]*/-w $w/" docker-compose.yml
                sed -i -e "s/-b [0-9]*/-b $b/" docker-compose.yml
                docker-compose up &
                DOCKER_PID=$!
                sh ./gatherdata.sh &
                GATHER_PID=$!
                python jlog.py
                kill $GATHER_PID &&
                curl -s -XDELETE 'http://localhost:9200/_all'
                kill $DOCKER_PID
        done
done
```

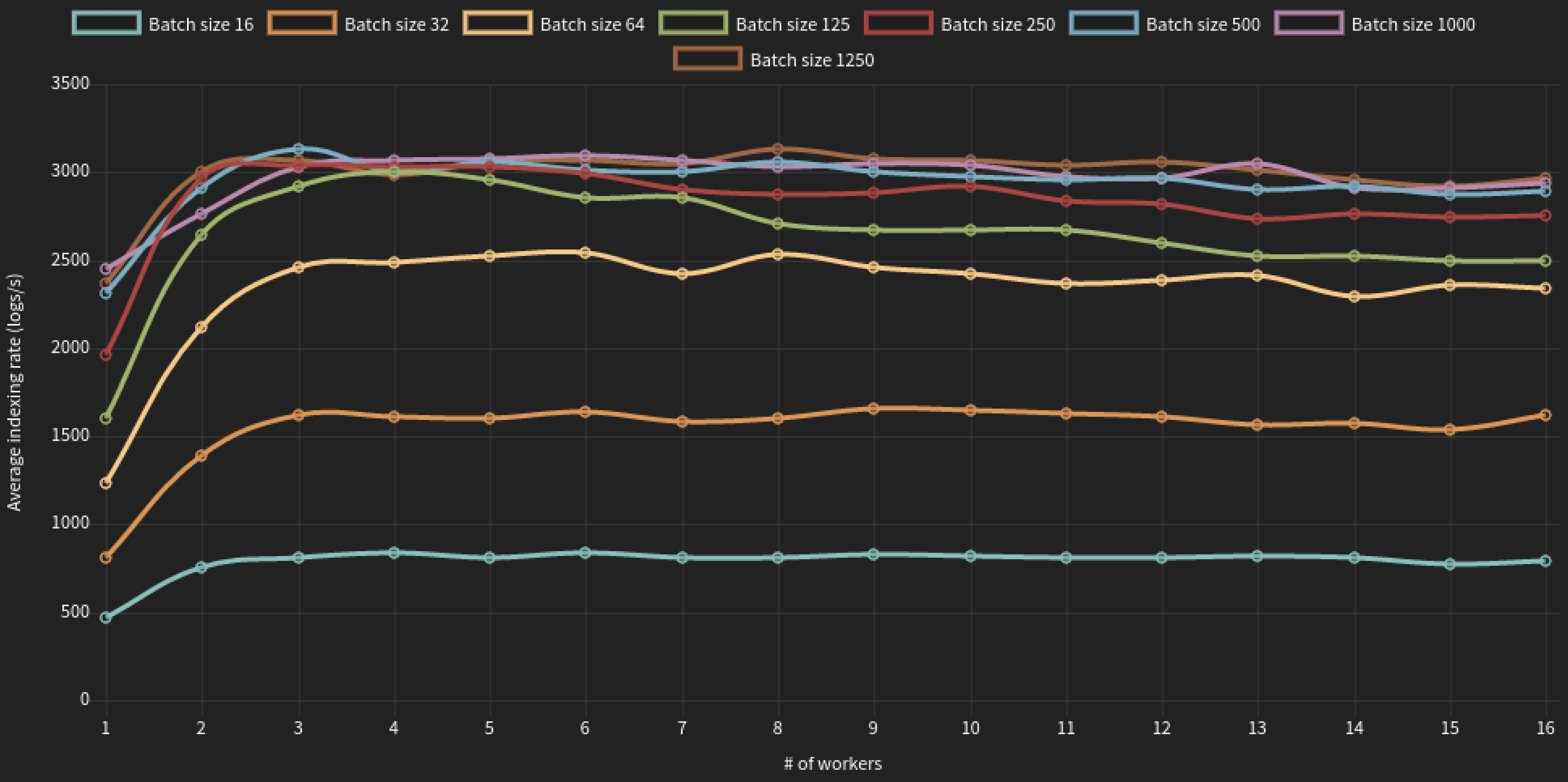
run.sh



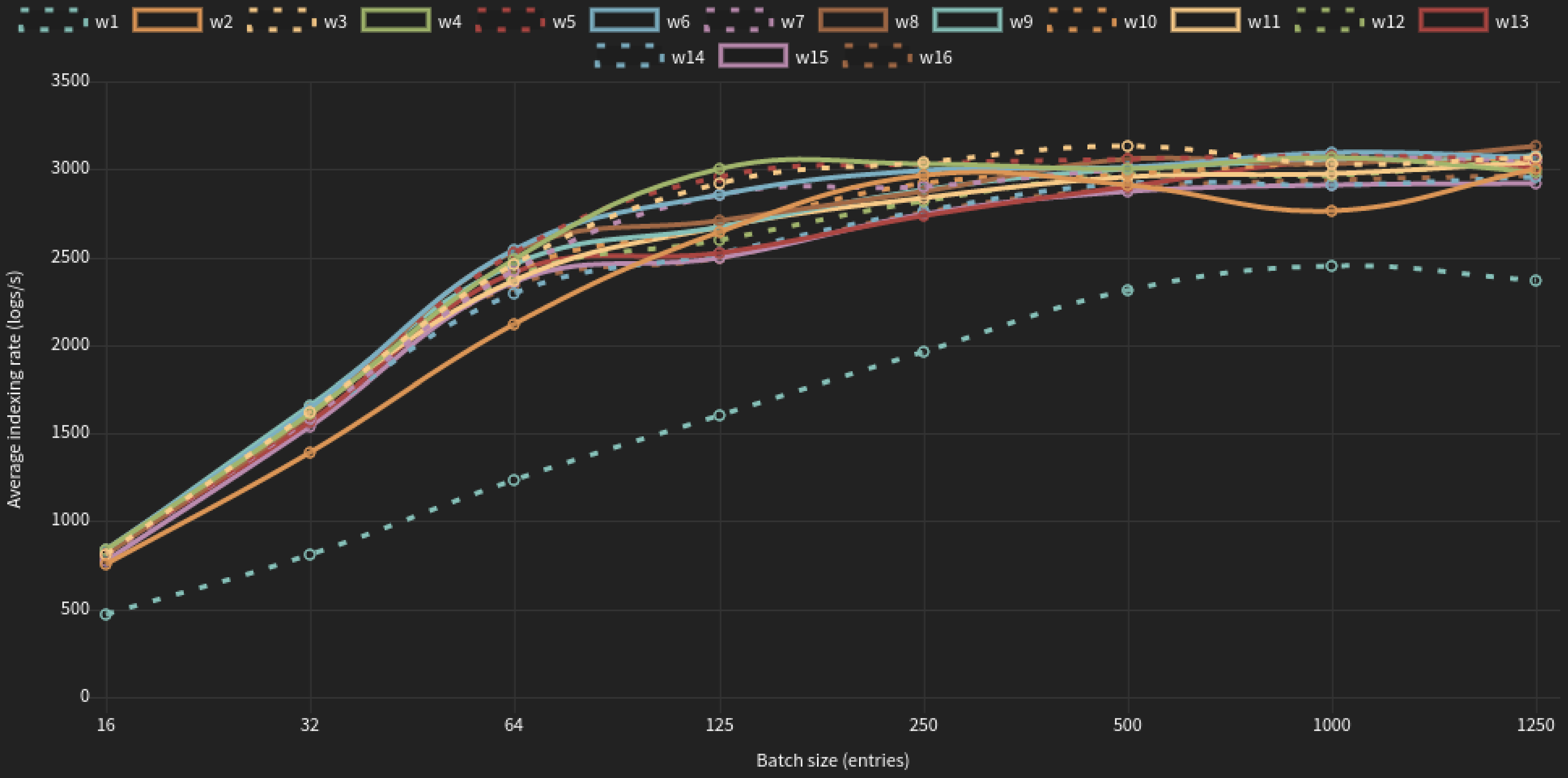
```
s.connect((TCP_IP, TCP_PORT))
for _ in range(0, MAX_LOGS):
        method = random.choice(methods)
        jlog = {
                'ip_src' : random.choice(ip_srcs),
                'websv' : random.choice(websvs),
                'method' : method,
                'query' : random.choice(gets) if method == 'GET'
                        else random.choice(posts),
                'protocol' : random.choice(protocols),
                'response' : random.choice(responses),
                'user' : ''.join(random.choice(
                        string.ascii_letters + string.digits)
                        for _ in range(6)),
                'usertype' : random.choice(usertypes),
                'user_ip' : ".".join(map(str, (random.randint(0, 255
                        for _ in range(4)))),
        }
        msg = json.dumps(jlog) + '\n'
        s.send(msg.encode('utf-8'))
s.close()
```

jlog.py

# RESULTS

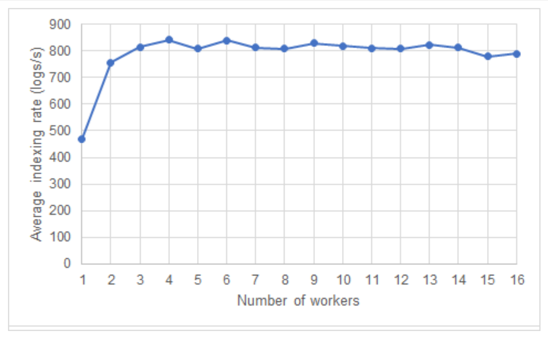

Batch size 16
Batch size 32
Batch size 64
Batch size 125
Batch size 250
Batch size 500
Batch size 1000
Batch size 1250
Average indexing rate (logs/s)
3500
3000
2500
2000
1500
1000
500
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
# of workers

# Batch size comparison

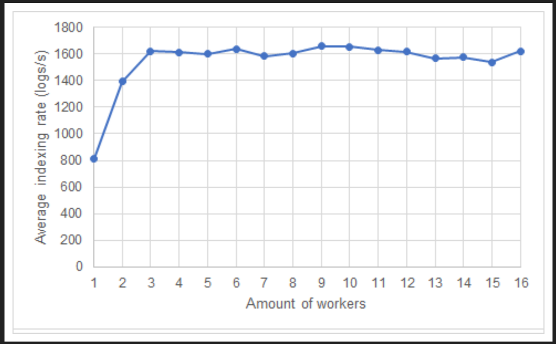


# Output workers comparison

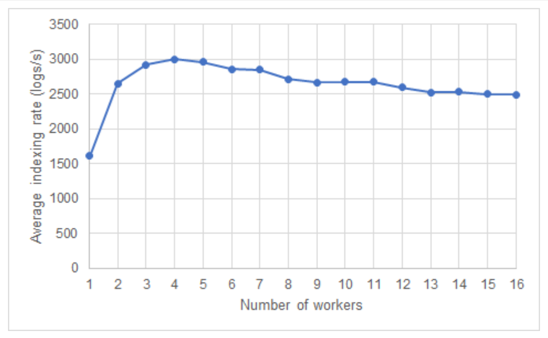
# SOME SPECIAL CASES



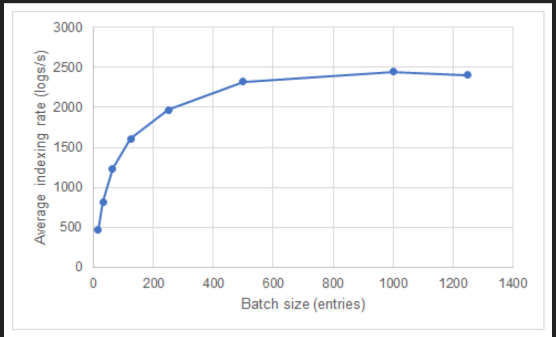
Batch size 16



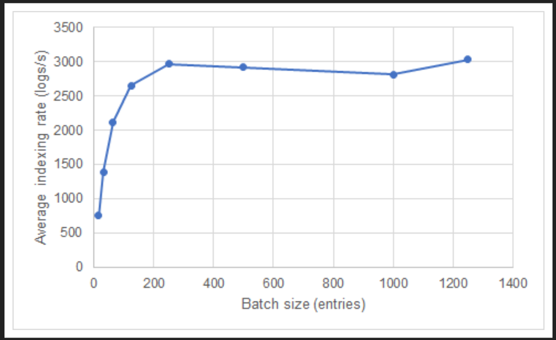
Batch size 32



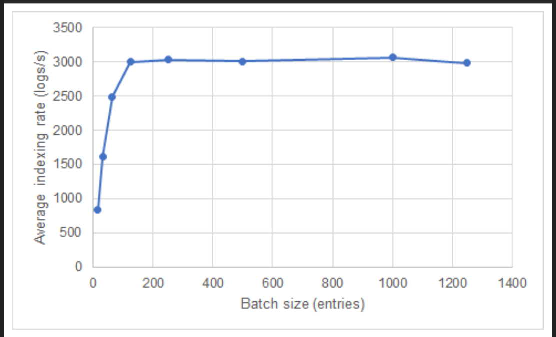
Batch size 125



1 worker



2 workers



4 workers

# CONCLUSION

- For this system, w ~ 4 and b ~ 150
- Effect of message queue and more Elasticsearch nodes
- Generic testbed for more complex scenarios
- Try it yourself!

# QUESTIONS?

# THANK YOU!

References

1. Product Overview. (n.d.). Retrieved January 03, 2017, from https://www.elastic.co/products
2. Deploying and Scaling Logstash | Logstash Reference [5.1] | Elastic. (n.d.). Retrieved January 03, 2017, from https://www.elastic.co/guide/en/logstash/current/deploying-and-scaling.html
3. Use Cases. (n.d.). Retrieved January 08, 2017, from https://www.elastic.co/use-cases
4. Logstash Pipeline Architecture Discussion. (2016, July 21). Retrieved January 03, 2017, from https://www.youtube.com/watch?v=FPLHS9Pmgk0