SCALE WITH **NSQ**

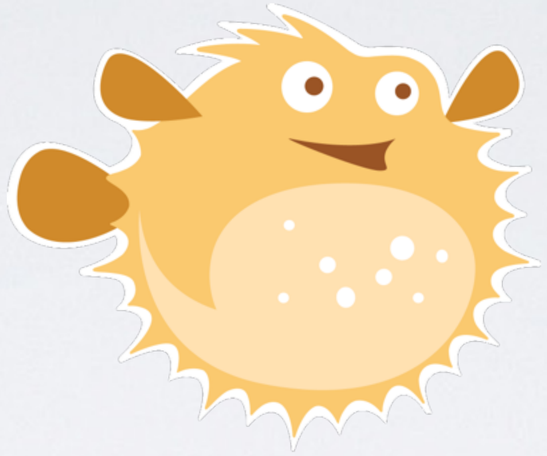
a realtime distributed messaging platform

@GeorgiCodes

Georgi Knox - Software Engineer @ Bitly

Berlin Buzzwords June 2, 2015

bitly



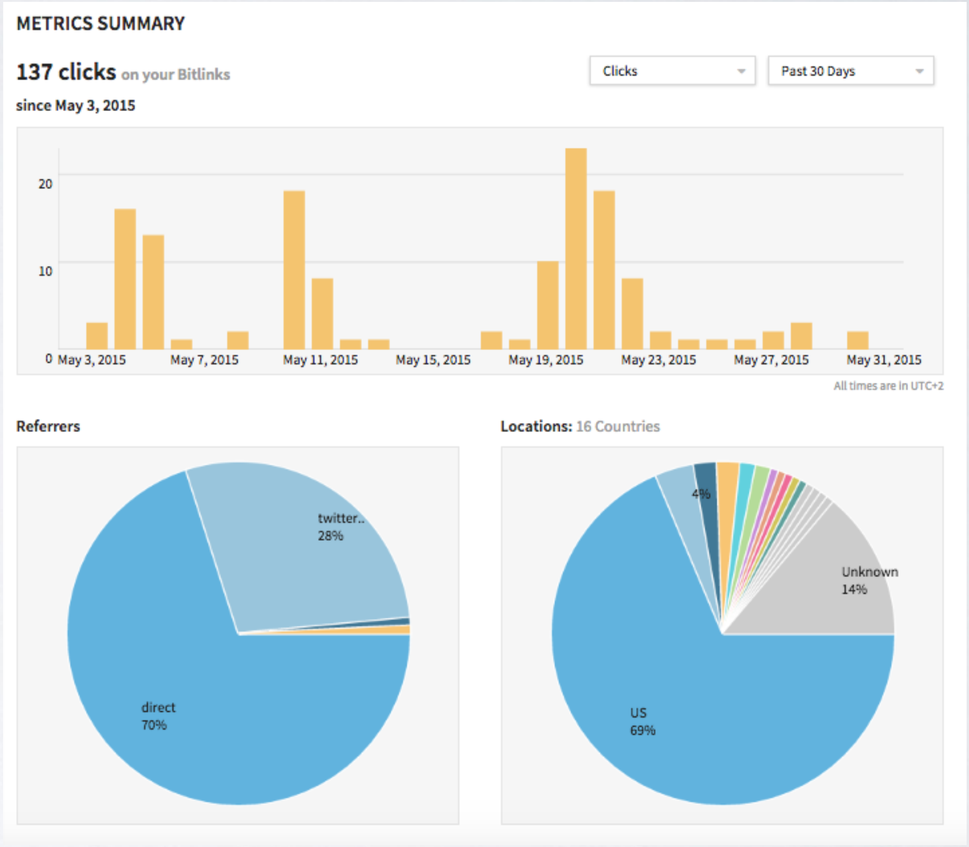
G'DAY, MY NAME IS GEORGI

@GeorgiCodes

bitly

WHAT IS **BITLY**?

- Popular URL shortener
- analytics around **how** and **where** those links were shared
- **10 billion** clicks per month
- **8000** requests per second



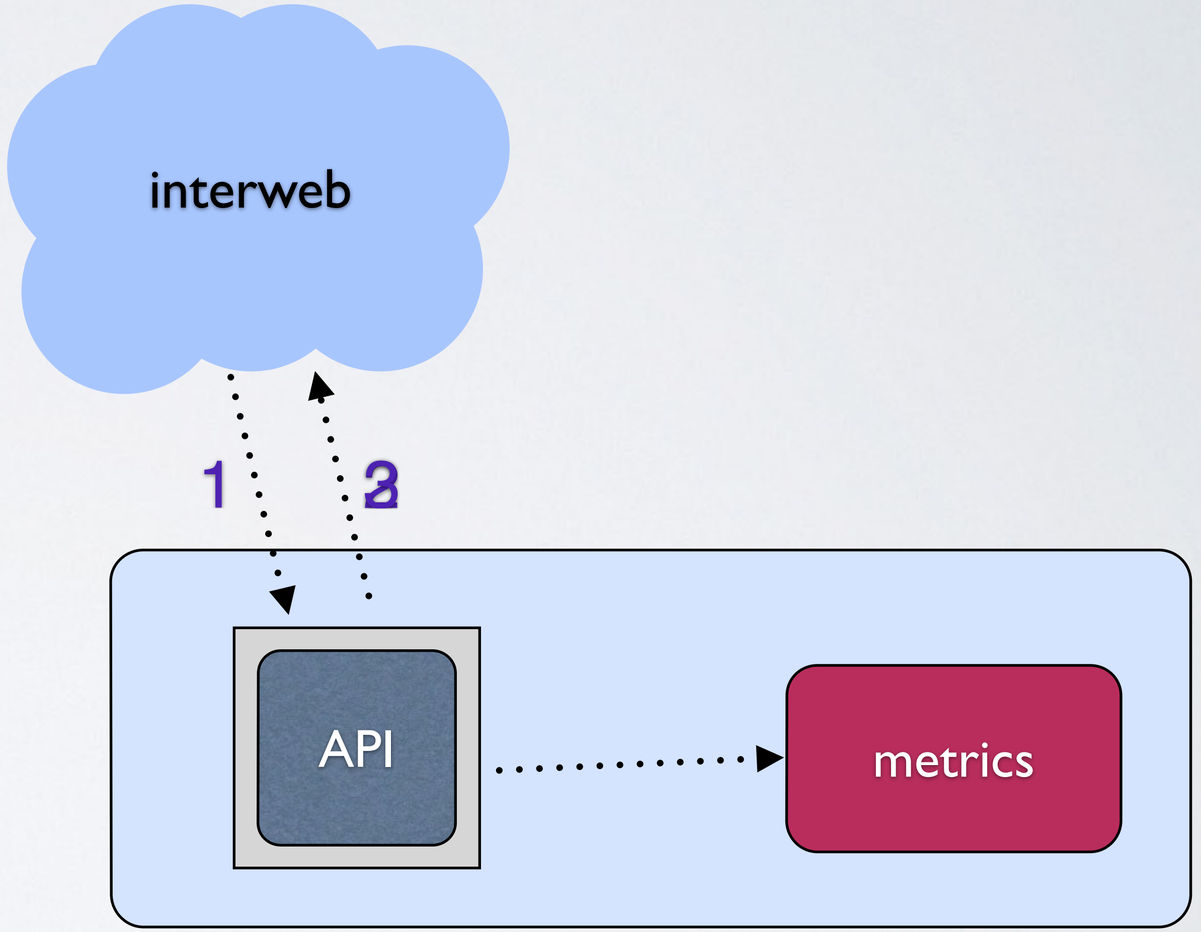
WHAT IS **NSQ**?

- At version: 0.3.5
- Open-sourced by Bitly
- written in Go
- 19 client libraries, 11 languages
- > 3 years in production

bitly

GROWING PAINS

- **single** host
- **synchronous** writes
- new feature: **track metrics**

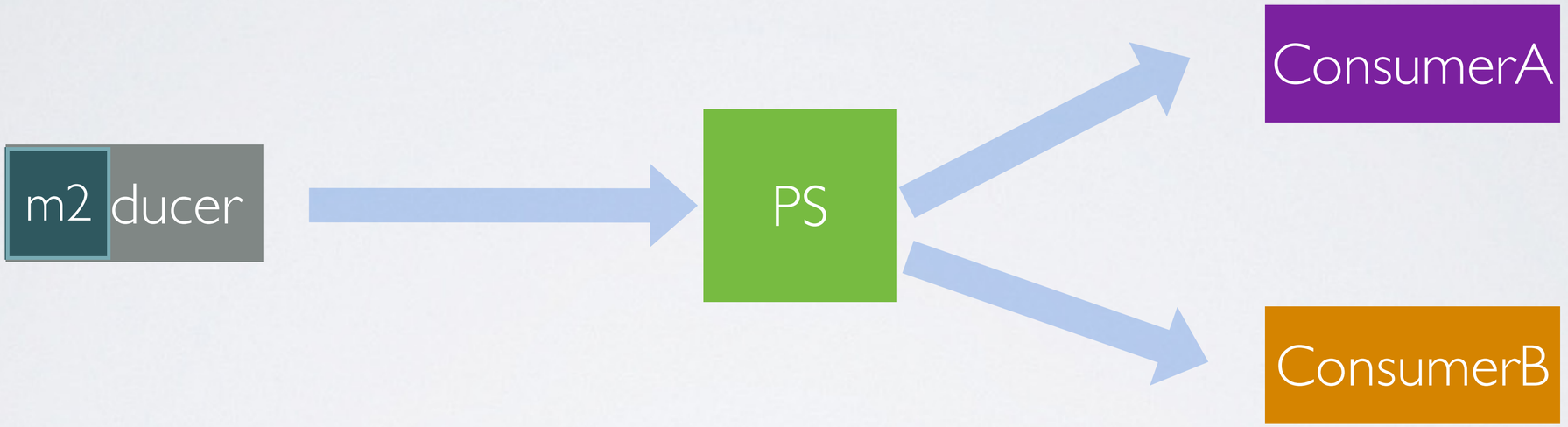


bitly

MESSAGING PATTERNS

bitly

BROADCAST

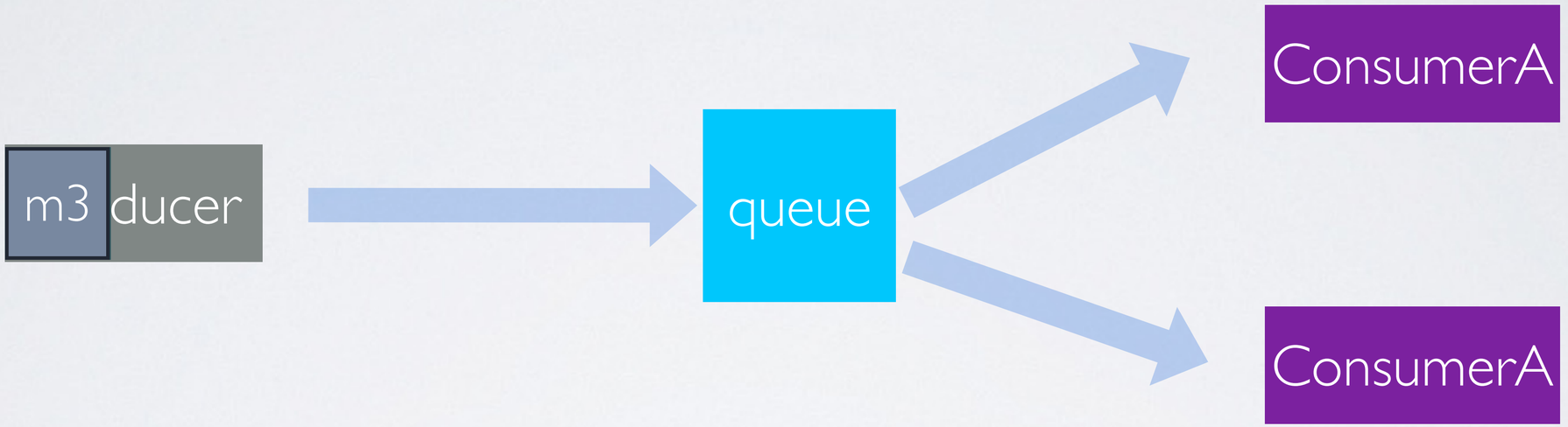


de-coupling of producers and consumers

multicast: message copied and delivered to n consumers

bitly

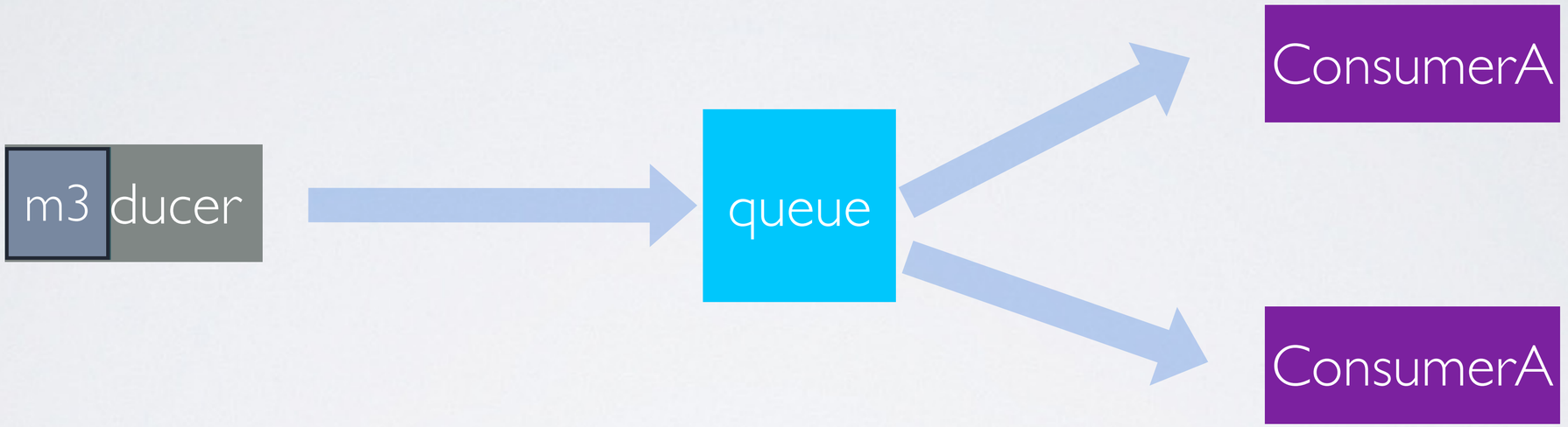
DISTRIBUTION



horizontal scalability

bitly

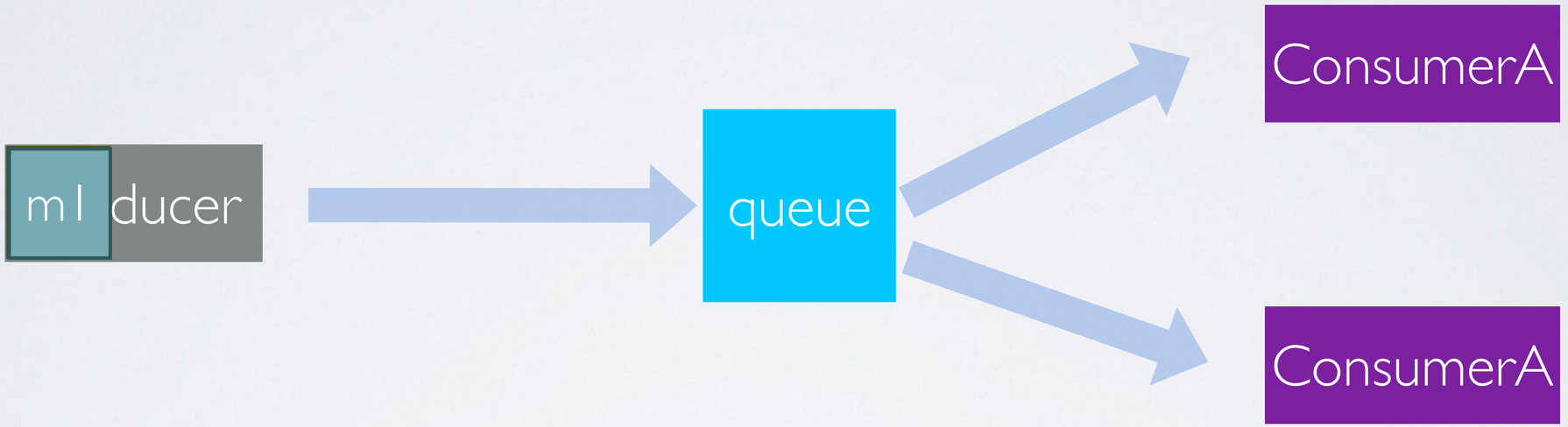
FAILURE



fault tolerance

bitly

EVEN MORE FAILURE



when *all the things* fail

bitly

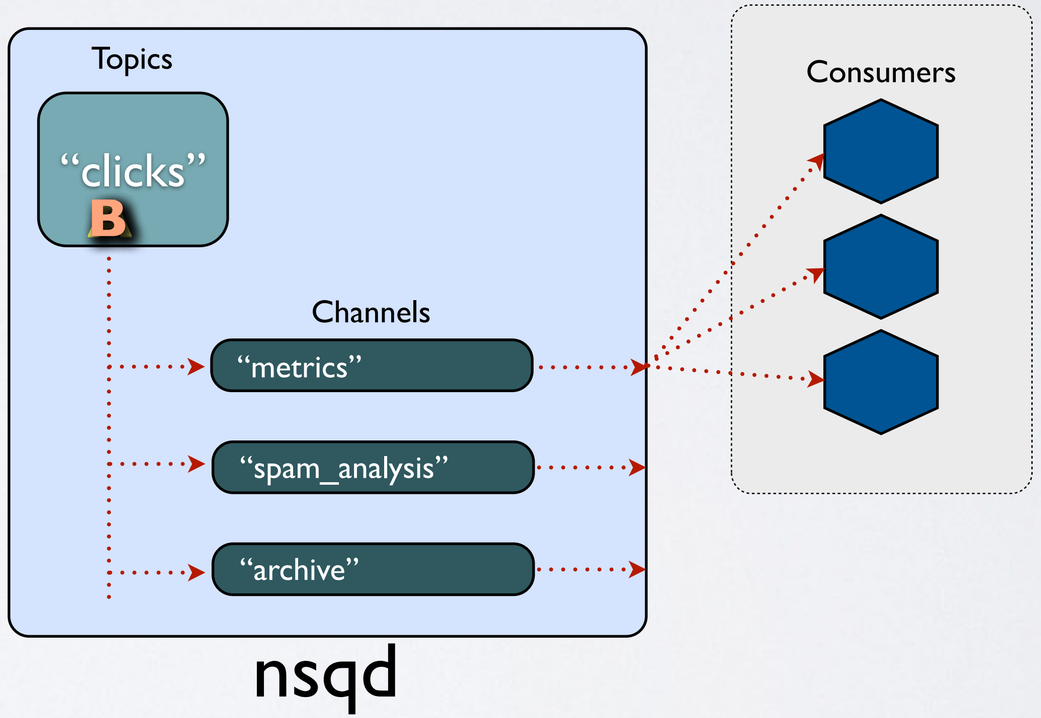
NSQD

bitly

TOPICS AND CHANNELS

combine pubsub, distribution, and queueing

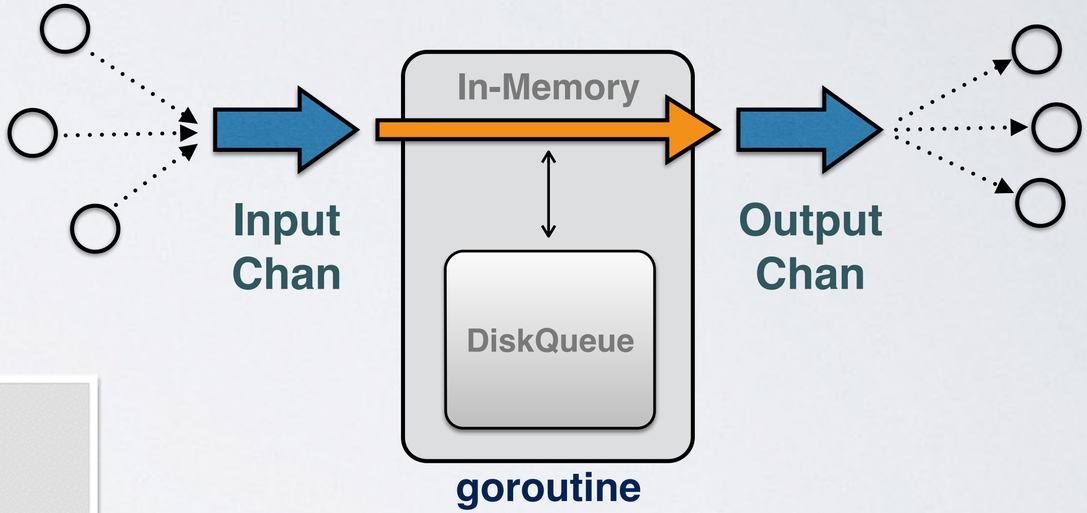
- a **topic** is a distinct stream of messages
- a **topic** has one or more **channels**
- topics and channels are created at **runtime**
- messages are **pushed** to consumers



bitly

UNDER THE HOOD

- **topics** and **channels** are *independent*
- configurable high water mark (disk persistence)

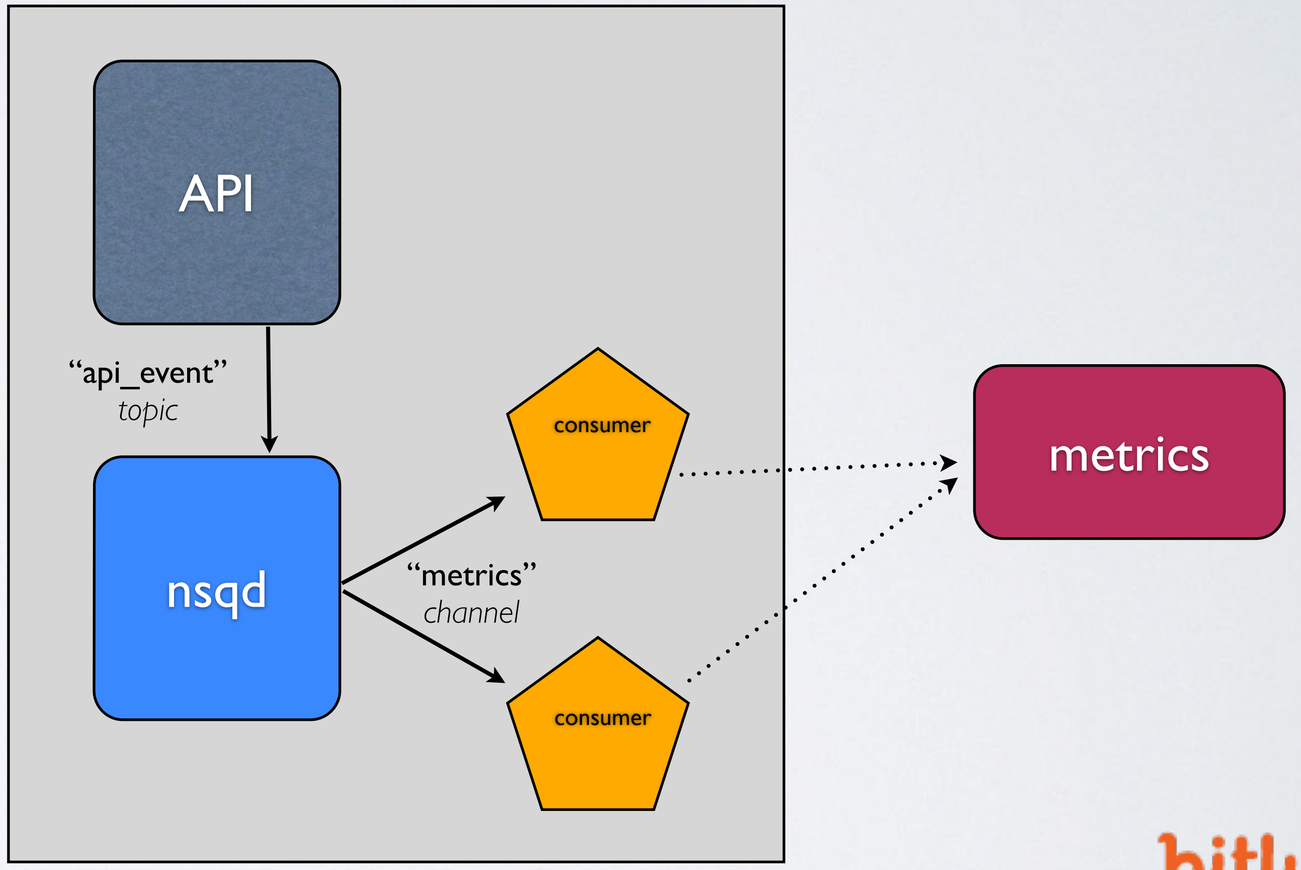



```
for msg := range c.incomingMsgChan {
    select {
    case c.memoryMsgChan <- msg:
    default:
        err := WriteMessageToBackend(&msgBuf, msg, c)
        if err != nil {
            // log whatever
        }
    }
}
```

bitly

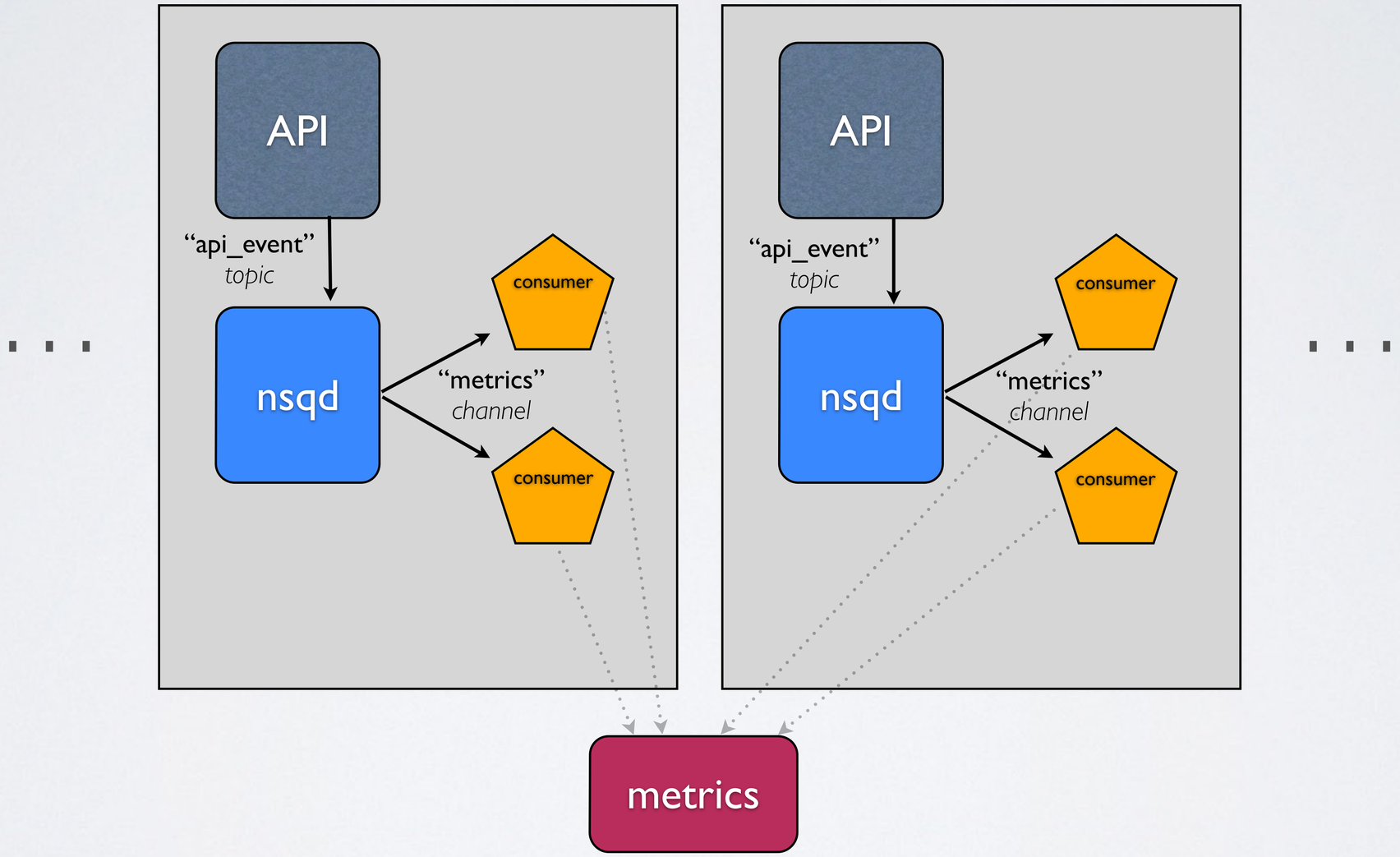
OUR SIMPLE EXAMPLE + NSQD

- introduce **nsqd**
- **de-coupled** production and consumption of data
- PUB locally to **nsqd** via HTTP
- perform work **async**
- co-locate everything (**silo**)



bitly

SCALE HORIZONTALLY

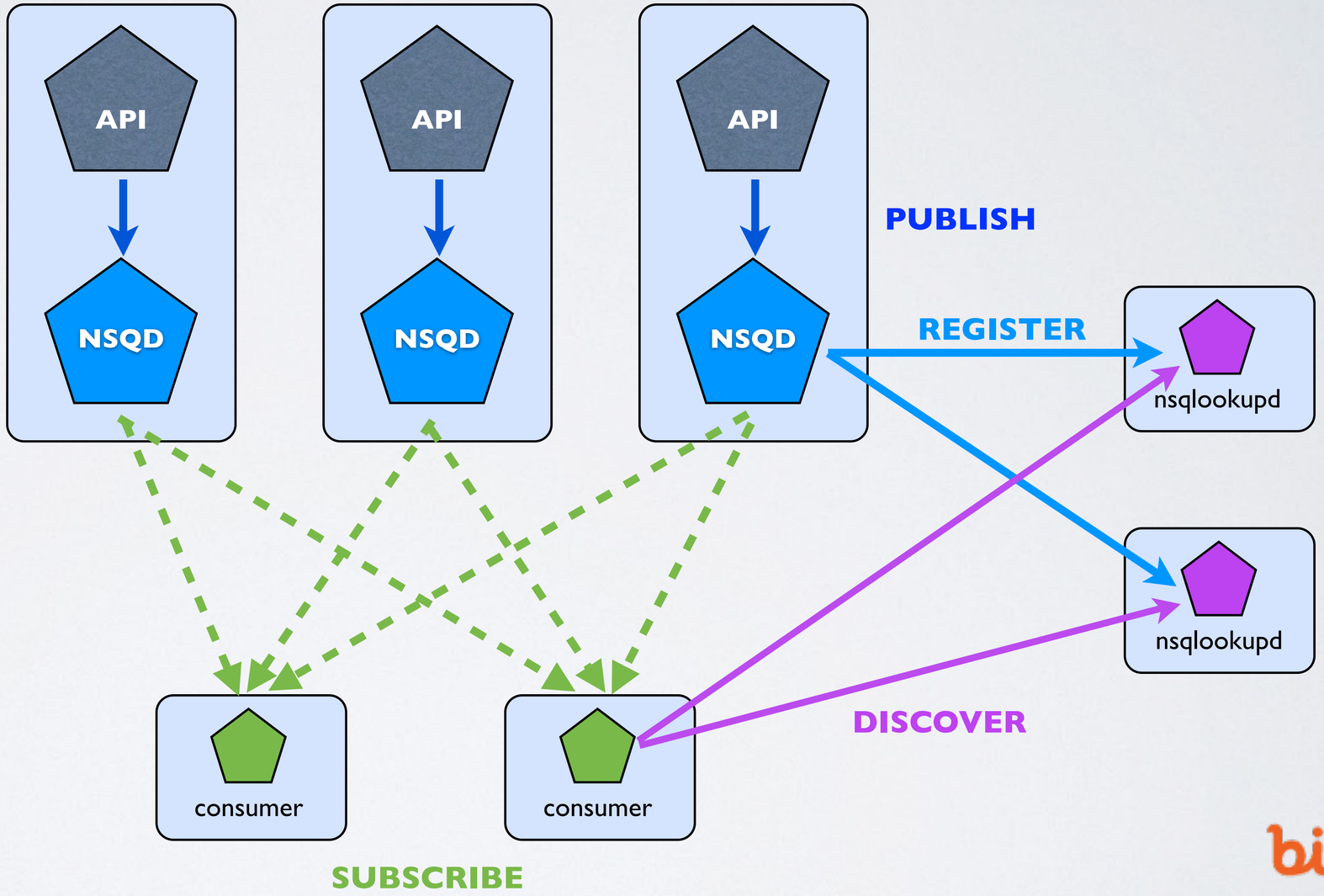


NSQLOOKUPD

bitly

TYPICAL NSQ CLUSTER

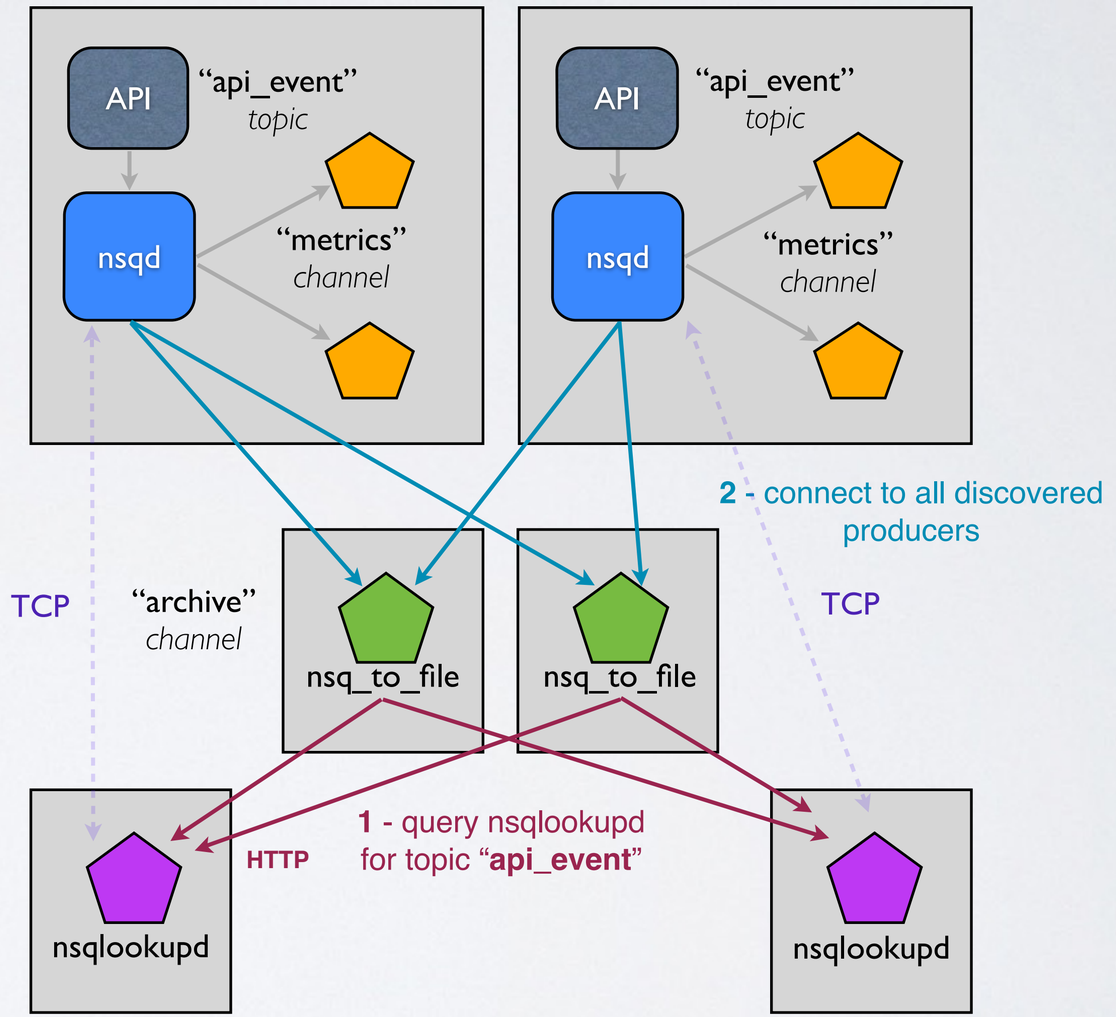
- enable *distributed* and *decentralized* topologies
- no centralized broker
- **nsqlookupd** instances are *independent* (**no coordinatation**)



bitly

OUR SIMPLE EXAMPLE + NSQLOOKUPD

- introduce **nsqlookupd**
- **discoverability**
- producers and consumers *come* and *go*
- other services can *discover* and *subscribe* to this topic



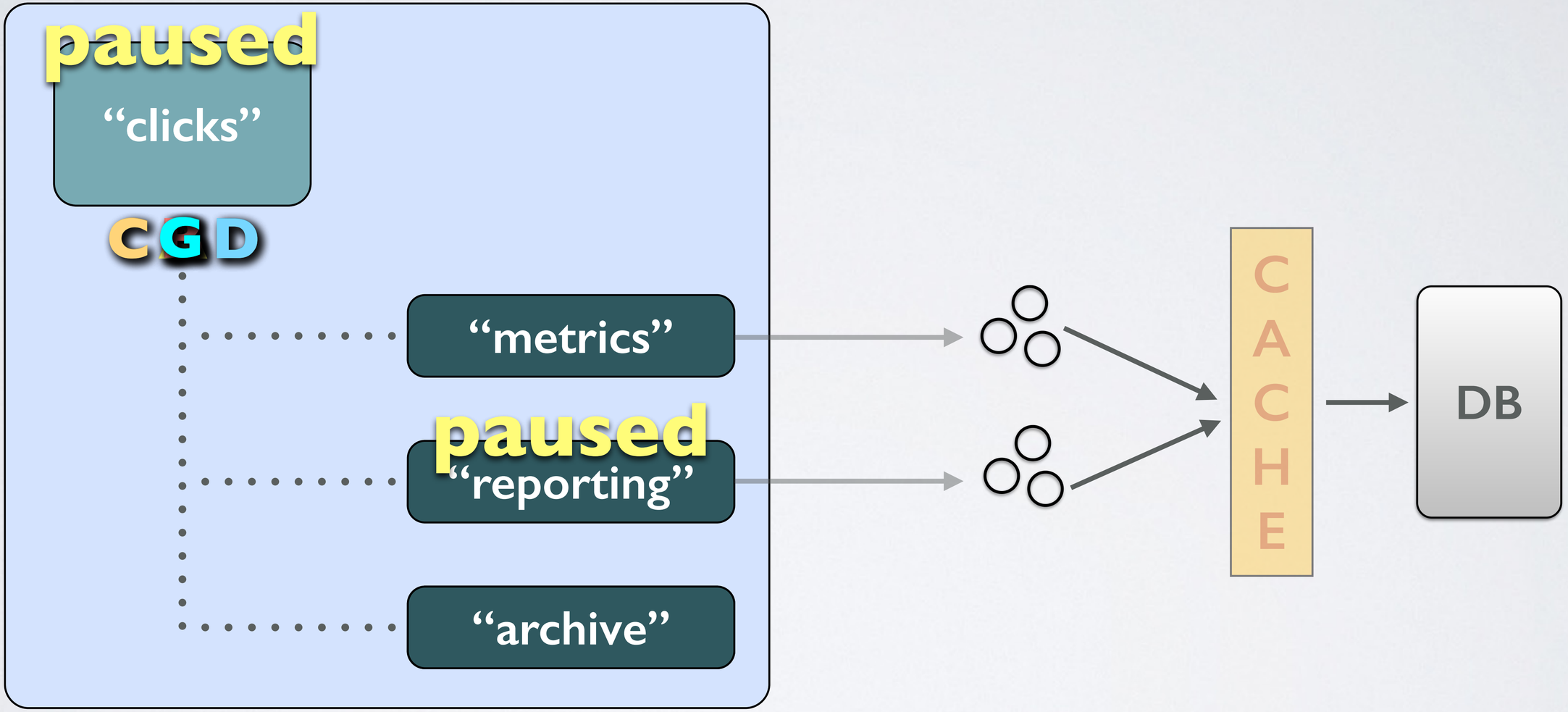
bitly

GUARANTEES

- messages are delivered *at least* once
- messages are *not* durable (by default)
- messages received are *un-ordered*
- consumers *eventually* find all topic producers

bitly

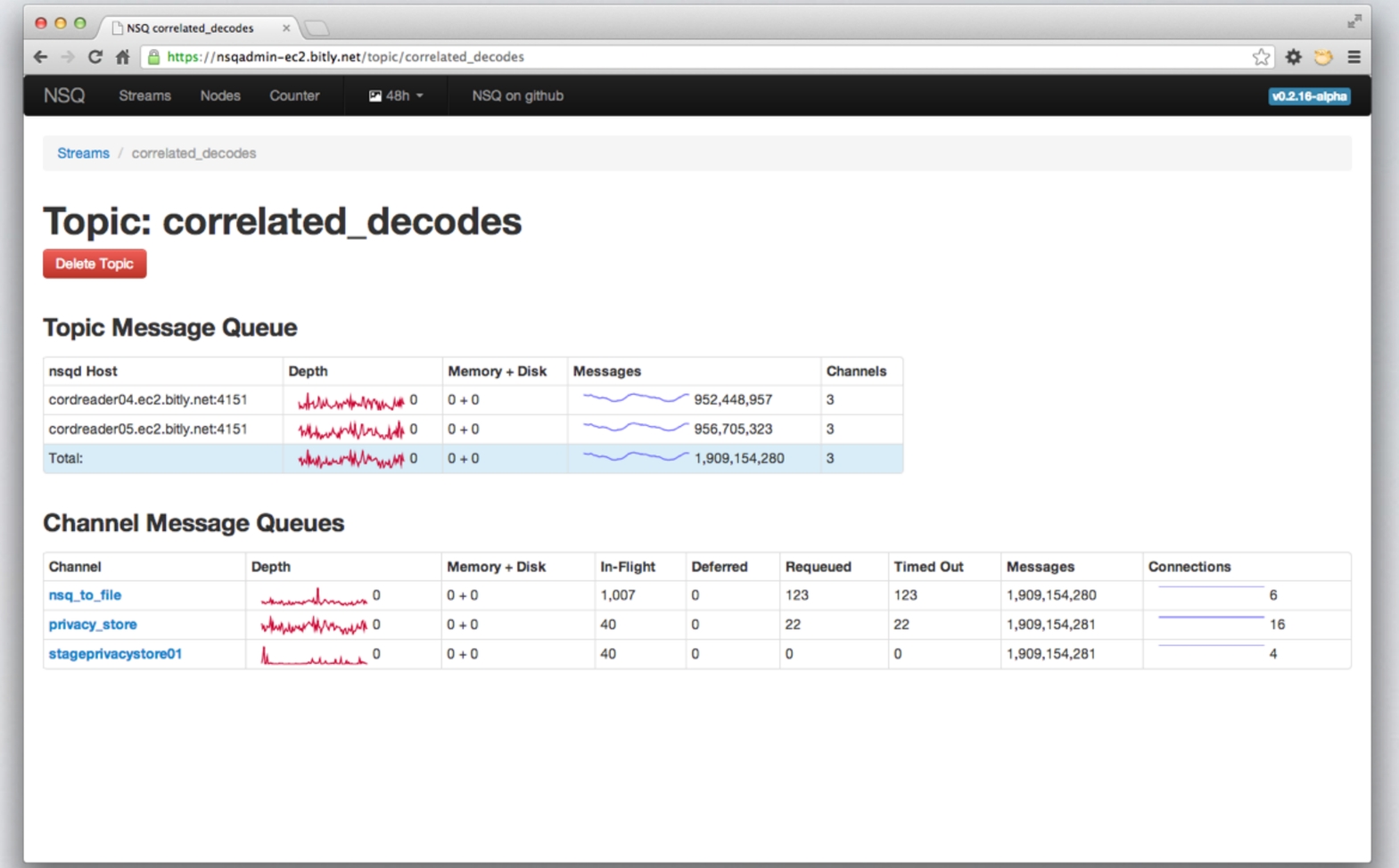
TOPIC & CHANNEL PAUSING



bitly

NSQ ADMIN

- **#ephemeral** channels



bitly

THANK-YOU!

@GeorgiCodes

big thanks to ***@jehiah*** *and* ***@imsnakes***
(authors of NSQ)

bitly