FRnOG#24

20 mars 2015

OVH

AS#16276

AntiDDoS : Threat detection with Kafka and Storm

Steven Le Roux
Infrastructure Engineer

Année de création

1999

International

12 implantations en Europe

2 succursales en Amérique du Nord

3 filiales en Afrique

Leader mondial

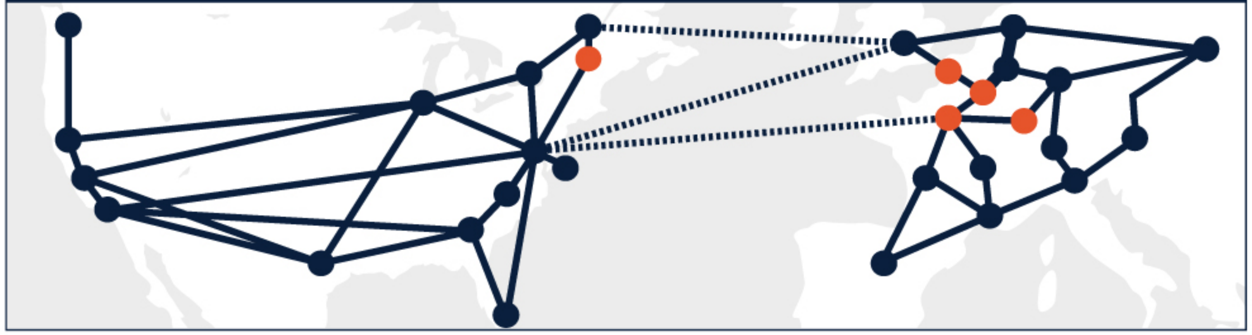
3e hébergeur Internet dans le monde*

Leader européen

1er hébergeur Internet en Europe et en France*

700 000
clients dans le monde

30 points de présence (POP) connectés à notre réseau mondial en fibre optique



2 POP en Asie

Bande passante

3 Tbps

180 000
serveurs

Levée de fonds

140 millions
d'euros

Nos centres de données

17 centres de données
En activité

2 centres de données
En projet

Web
Dedicated
Telecom
Cloud

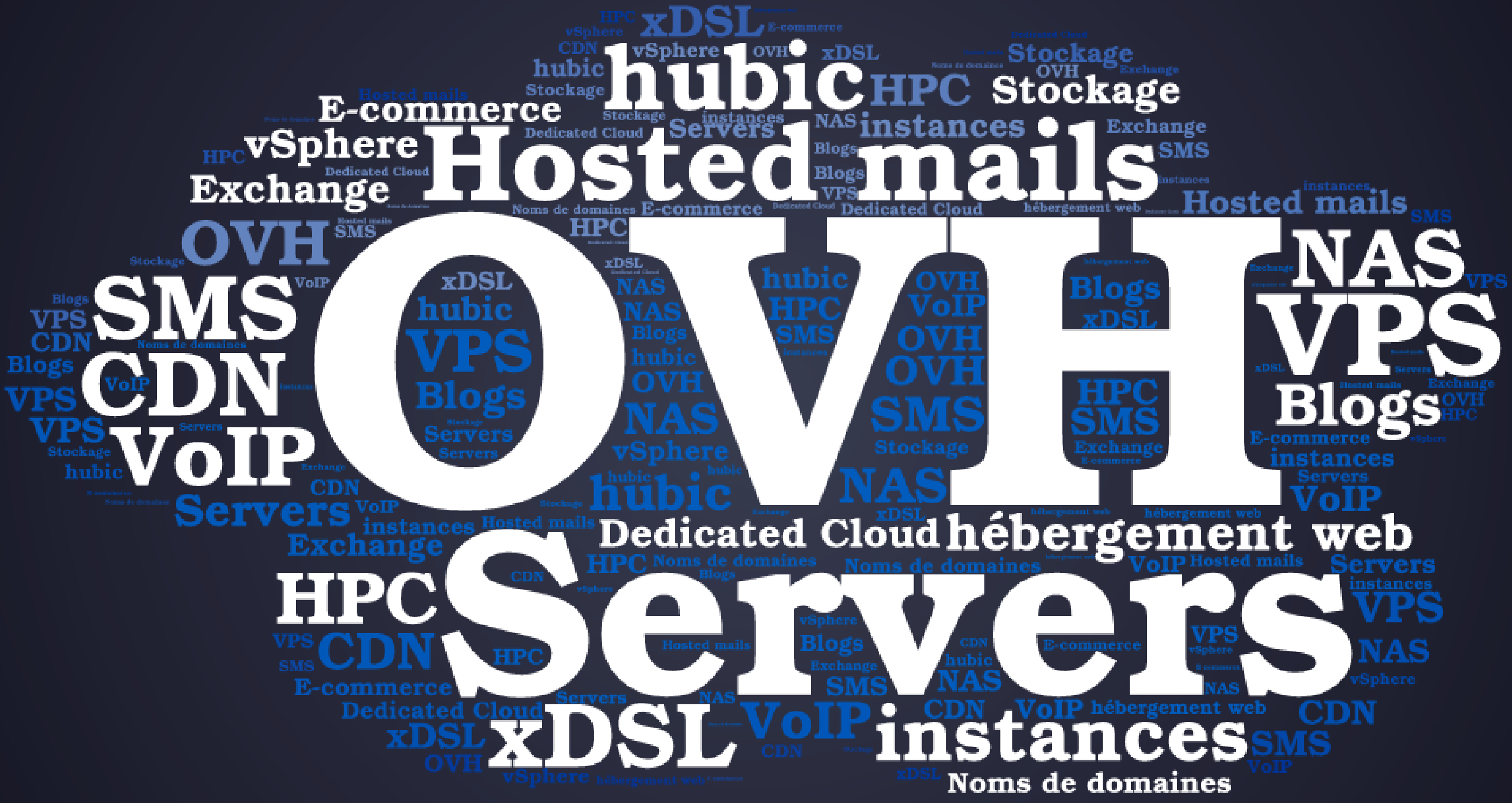
Kimsufi

So you Start.com

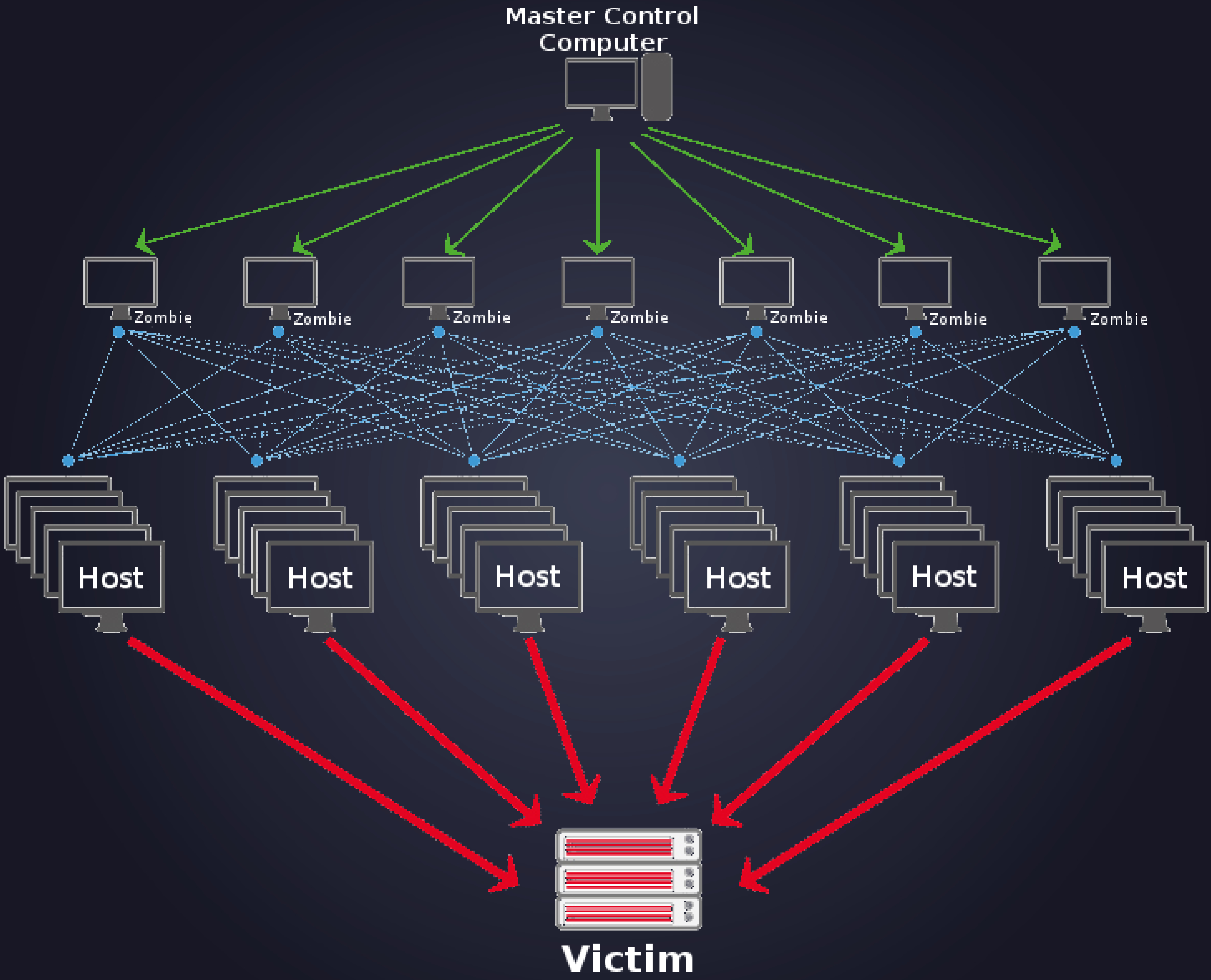
hubiC

oxalya

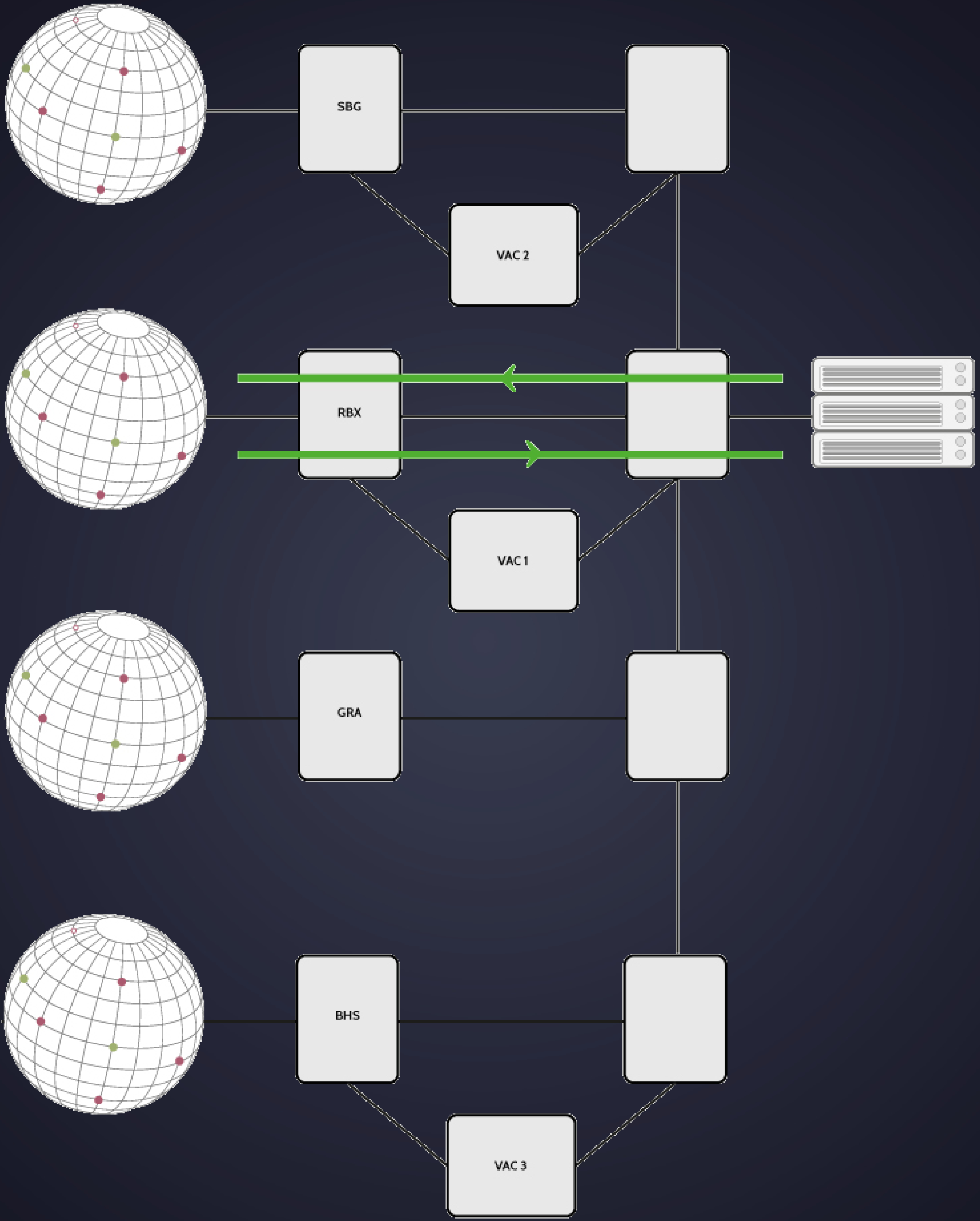
HPC by OVH.COM

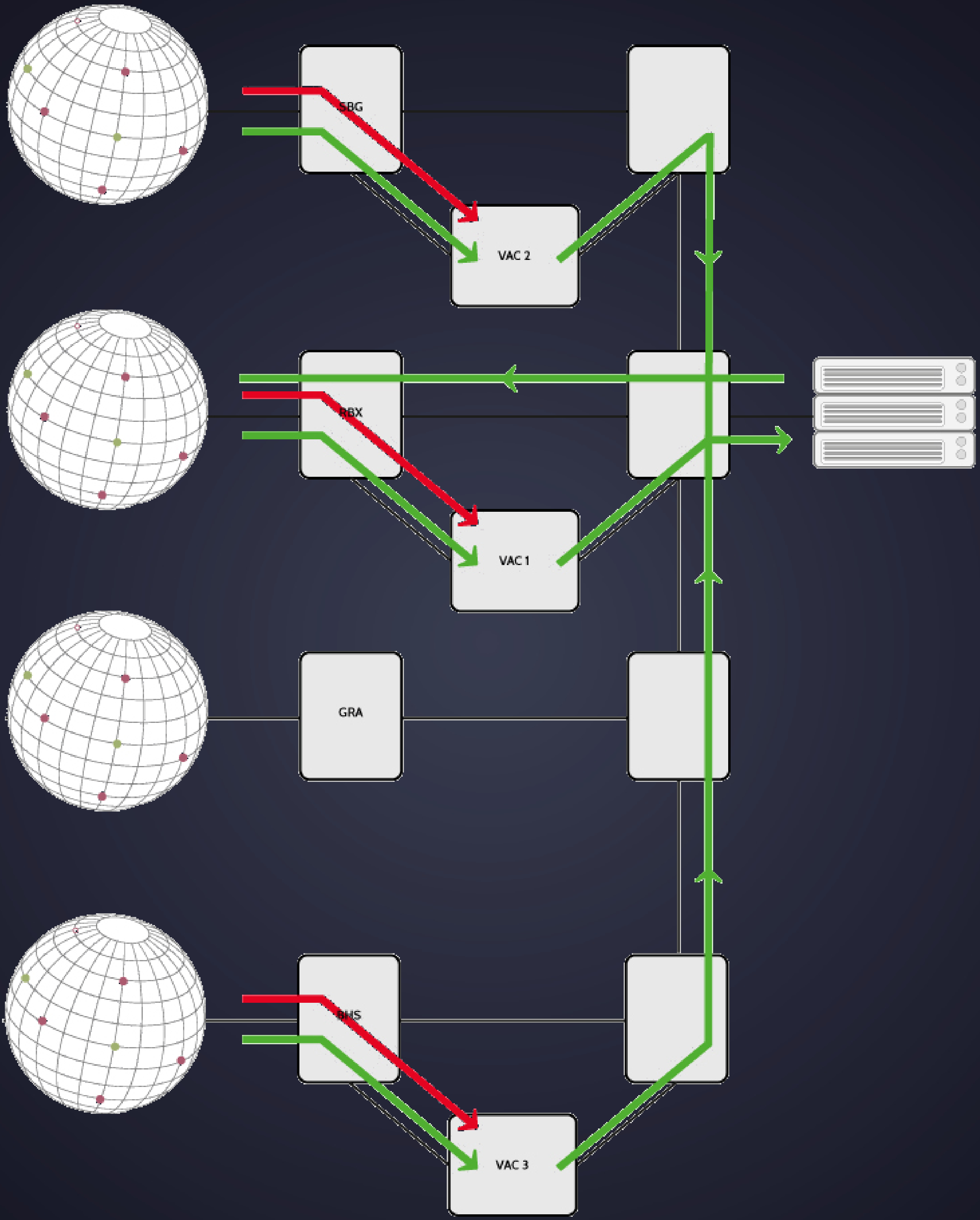

OVH
Servers
Hosted mails
hubic
Stockage
E-commerce
vSphere
Exchange
SMS
CDN
VoIP
NAS
VPS
Blogs
Dedicated Cloud
hébergement web
HPC
xDSL
instances
Noms de domaines

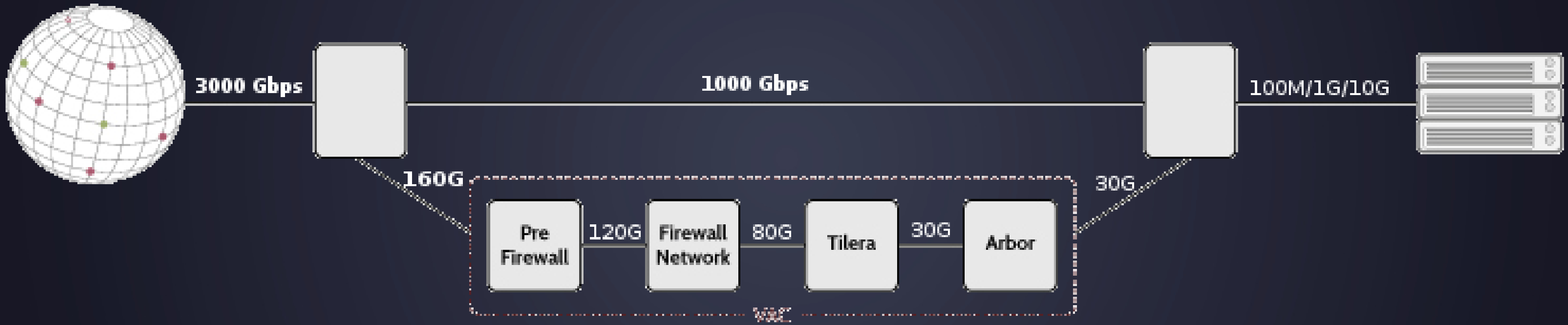
OVH Anti-DDoS

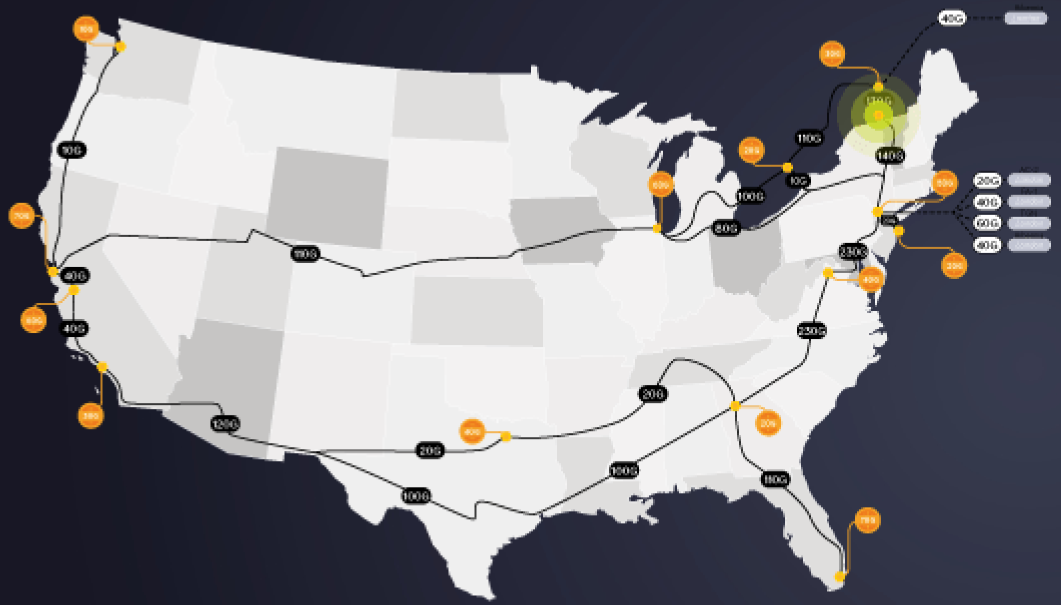
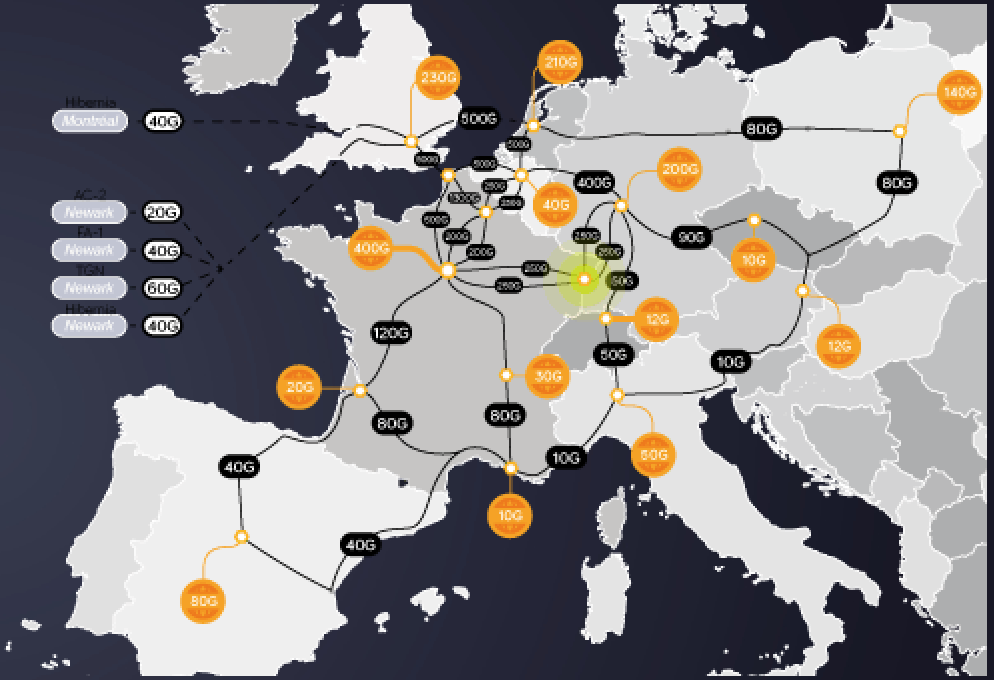

Master Control Computer
Zombie
Zombie
Zombie
Zombie
Zombie
Zombie
Zombie
Host
Host
Host
Host
Host
Host
Victim

VAC


SBG
VAC 2
RBX
VAC 1
GRA
BHS
VAC 3

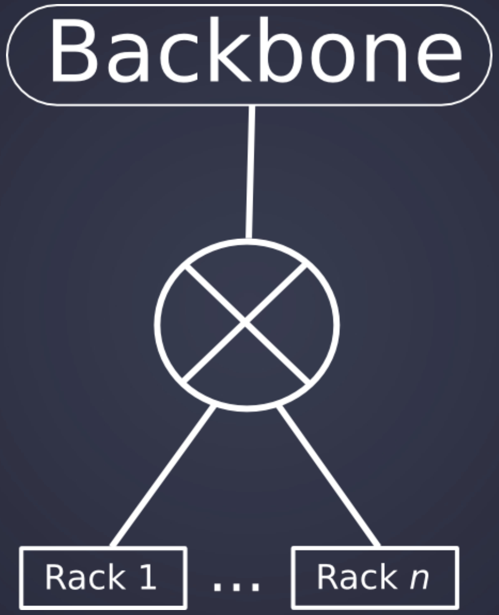

SBG
VAC 2
RBX
VAC 1
GRA
BHS
VAC 3

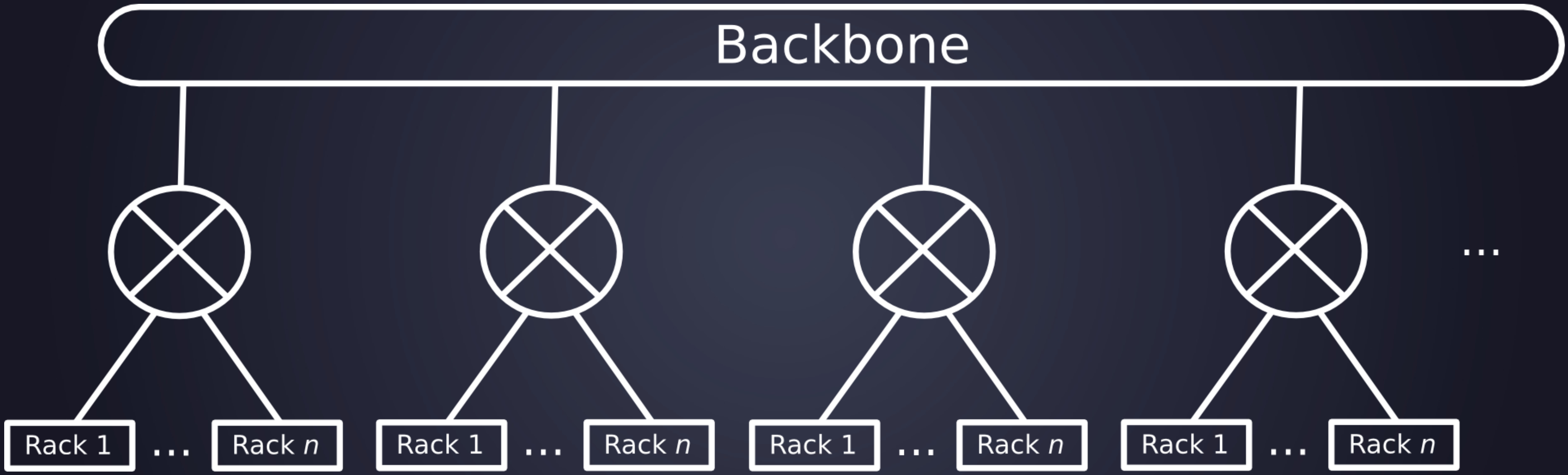

3000 Gbps
1000 Gbps
100M/1G/10G
160G
30G
Pre Firewall
120G
Firewall Network
80G
Tilera
30G
Arbor
VAC

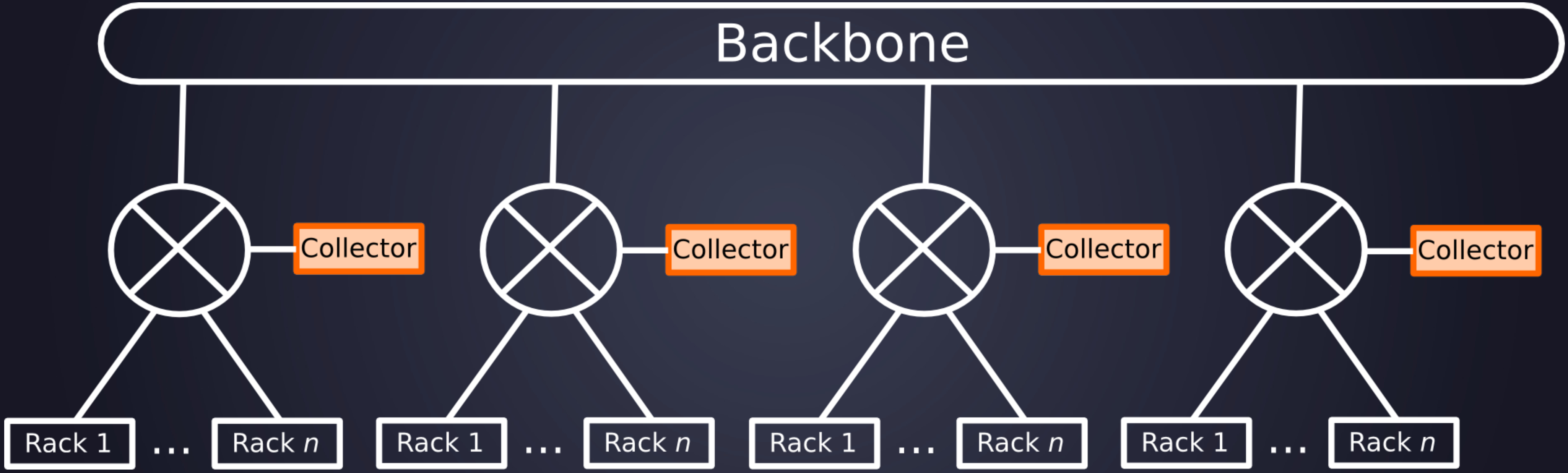



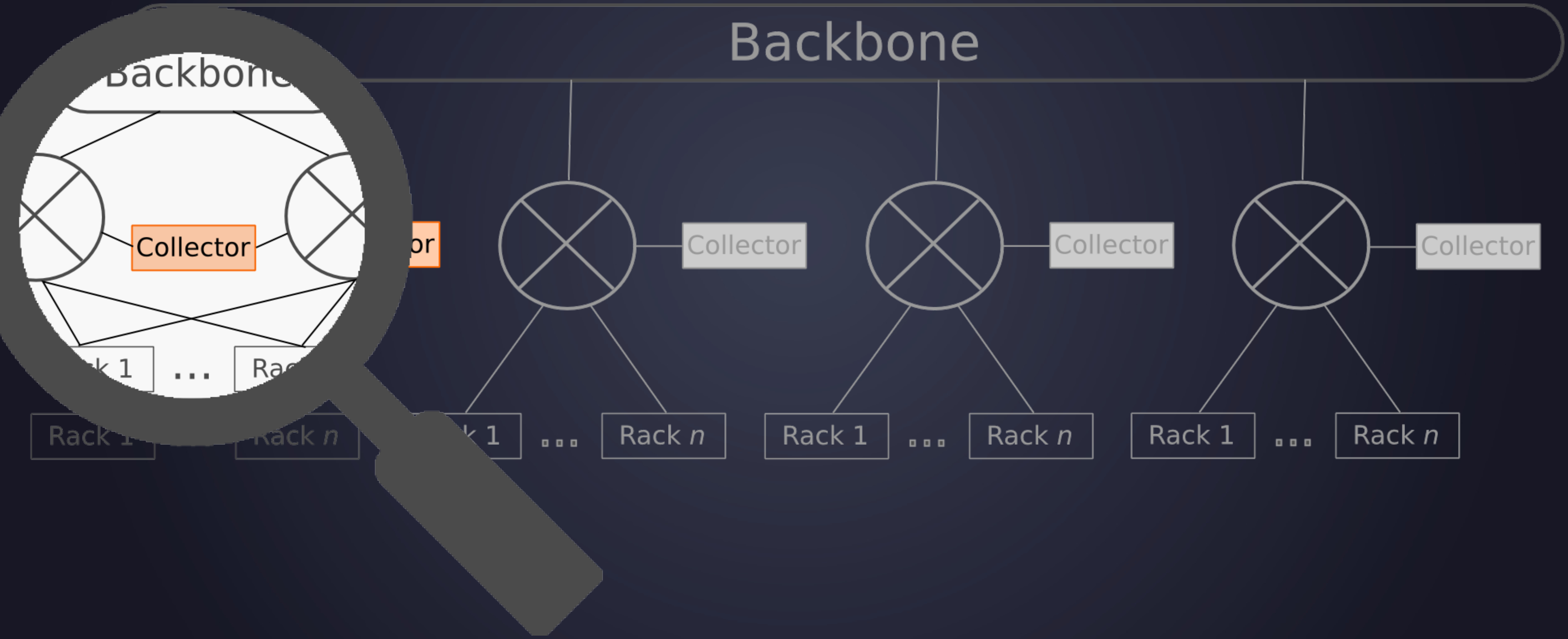
- 3 Tbps

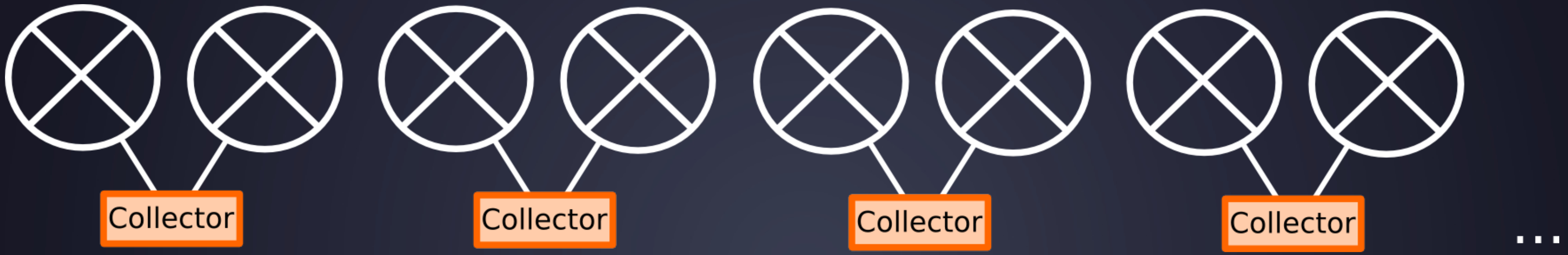
- 17 Datacenters
- 32 PoPs

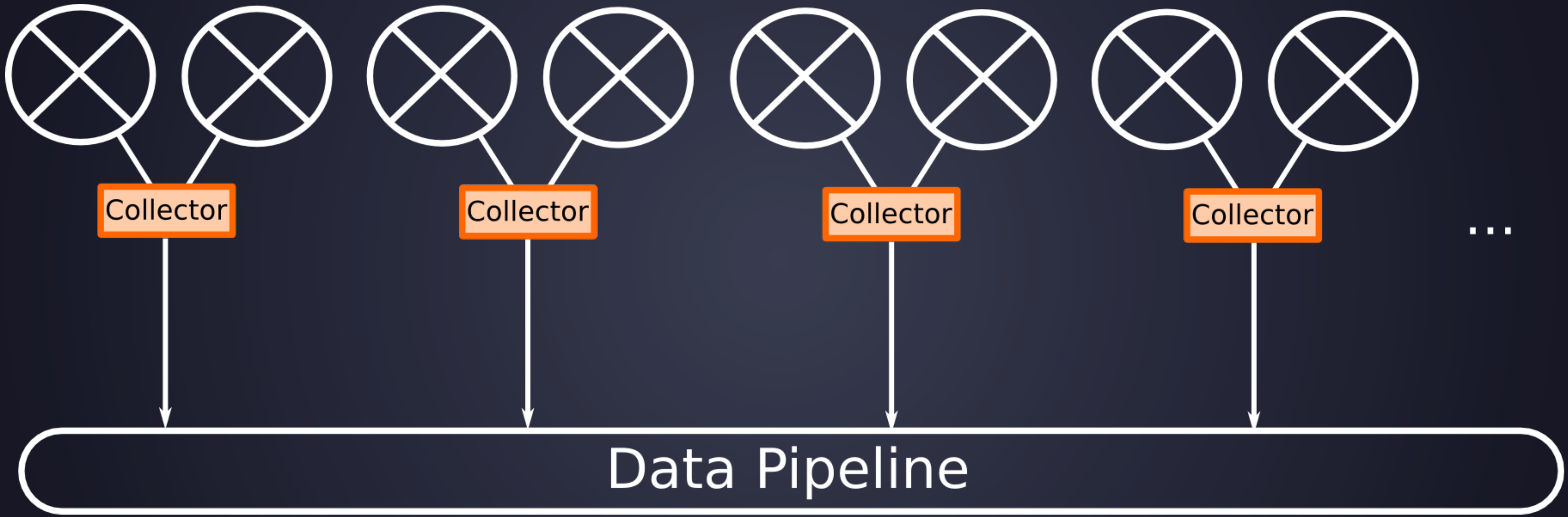

Backbone
Rack 1
...
Rack *n*

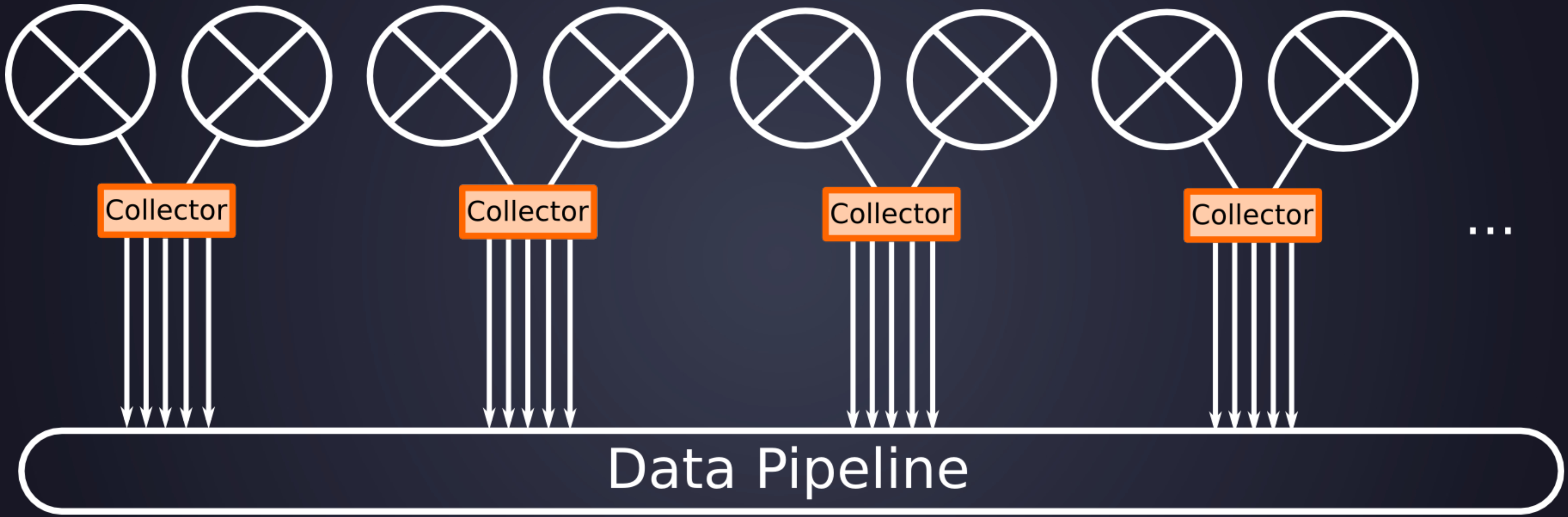

Backbone
Rack 1
...
Rack *n*
...


Backbone
Collector
Collector
Collector
Collector
Rack 1 … Rack *n*
Rack 1 … Rack *n*
Rack 1 … Rack *n*
Rack 1 … Rack *n*


Backbone
Collector
Collector
Collector
Collector
Rack 1
...
Rack n
Rack 1
...
Rack n
Rack 1
...
Rack n
Rack 1
...
Rack n


Collector
Collector
Collector
Collector
...


Collector
Collector
Collector
Collector
...
Data Pipeline


Collector
Collector
Collector
Collector
...
Data Pipeline

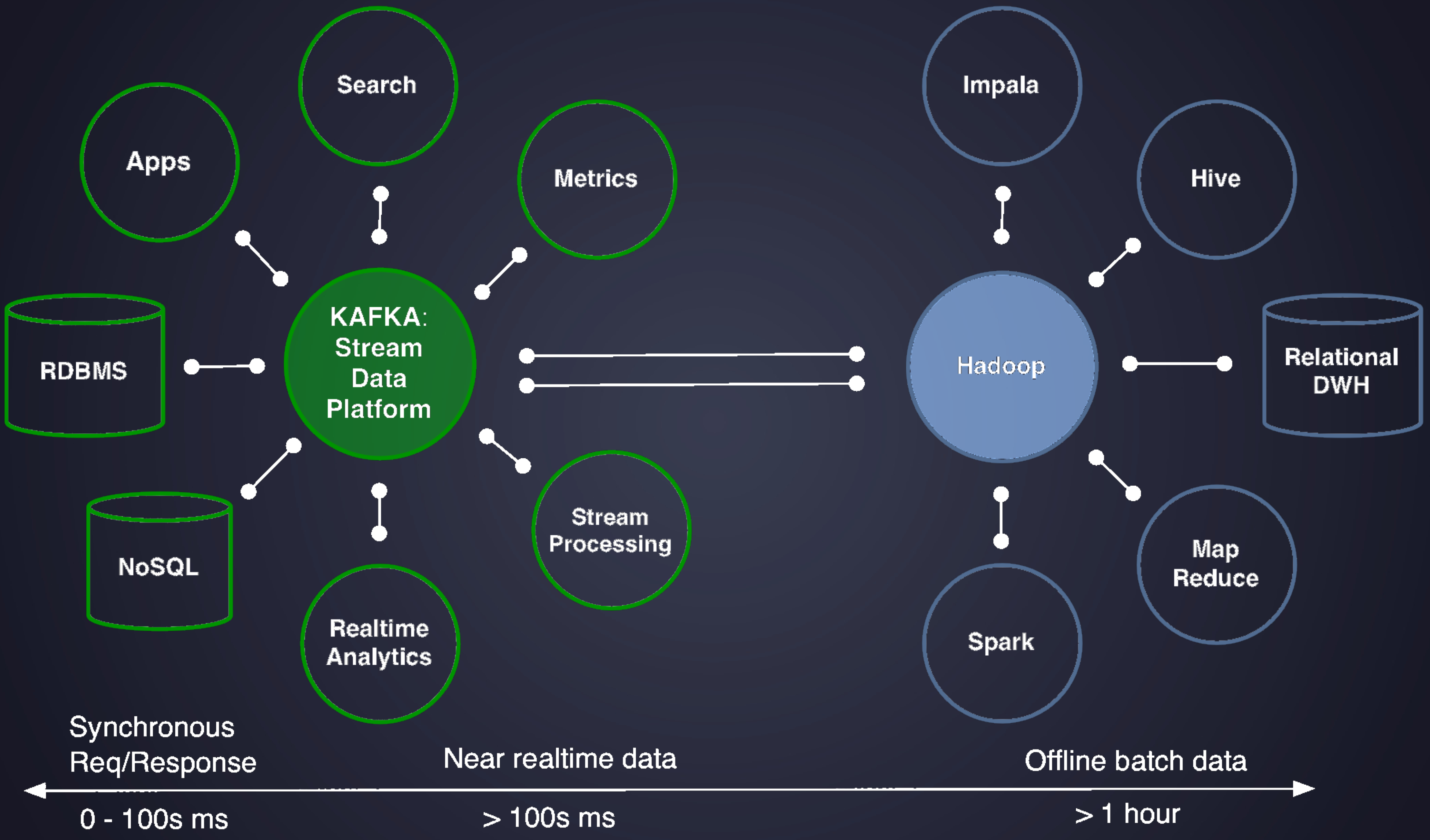
Data Pipeline

Apache Kafka

A high-throughput distributed messaging system.

Linked in

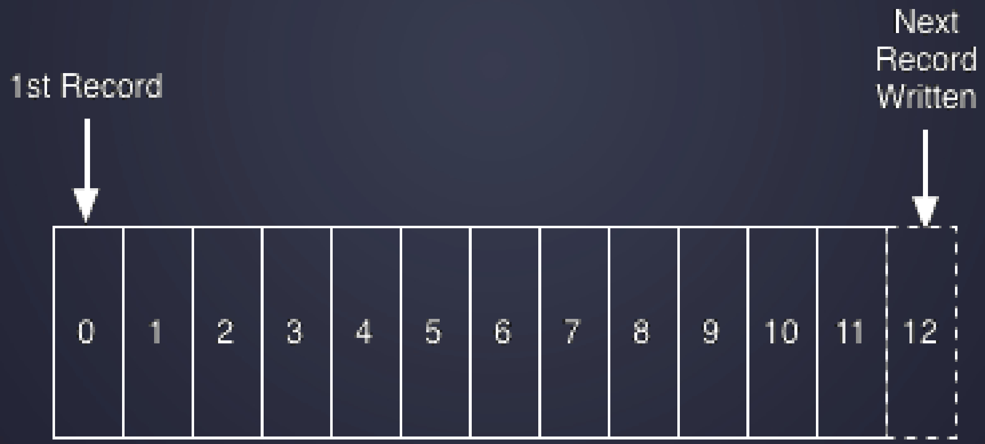
/ kafka


Search
Apps
Metrics
RDBMS
KAFKA: Stream Data Platform
NoSQL
Realtime Analytics
Stream Processing
Impala
Hive
Hadoop
Relational DWH
Spark
Map Reduce
Synchronous Req/Response
Near realtime data
Offline batch data
0 - 100s ms
> 100s ms
> 1 hour

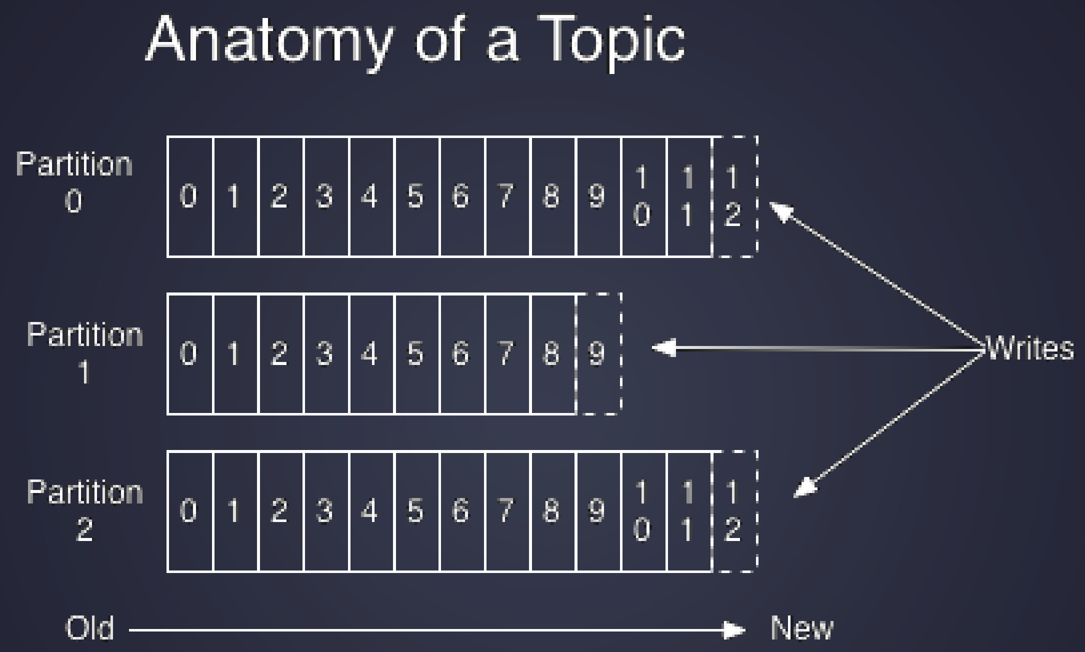
/ kafka

- Clients
  - Producers
  - Consumers
- Brokers
  - Topics
  - Partitions
  - Replicas

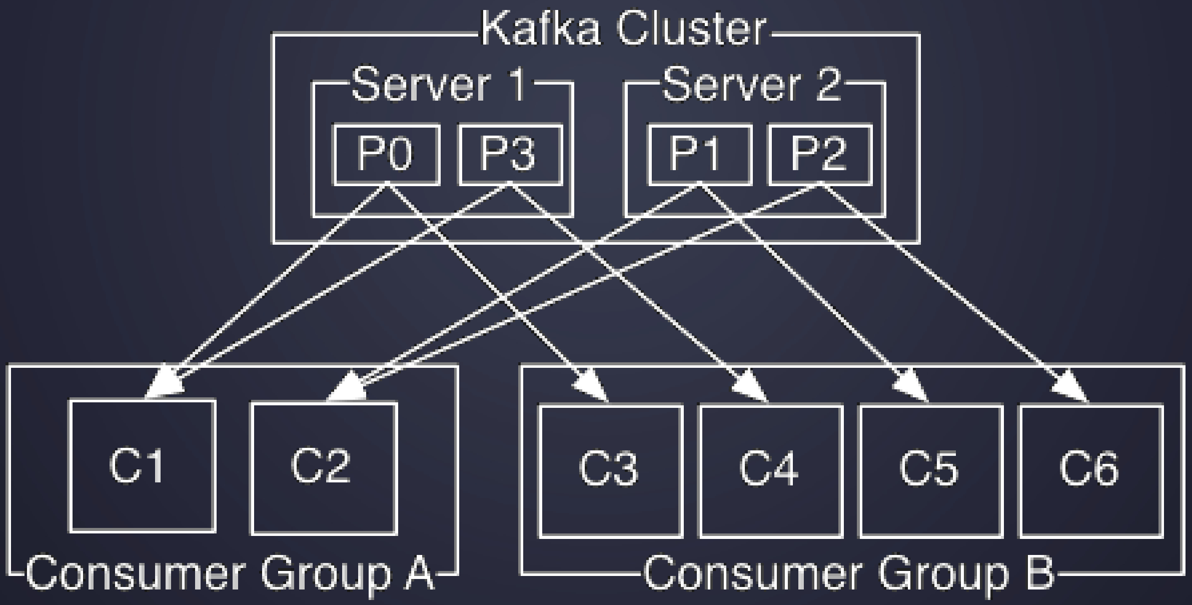
#commitLog



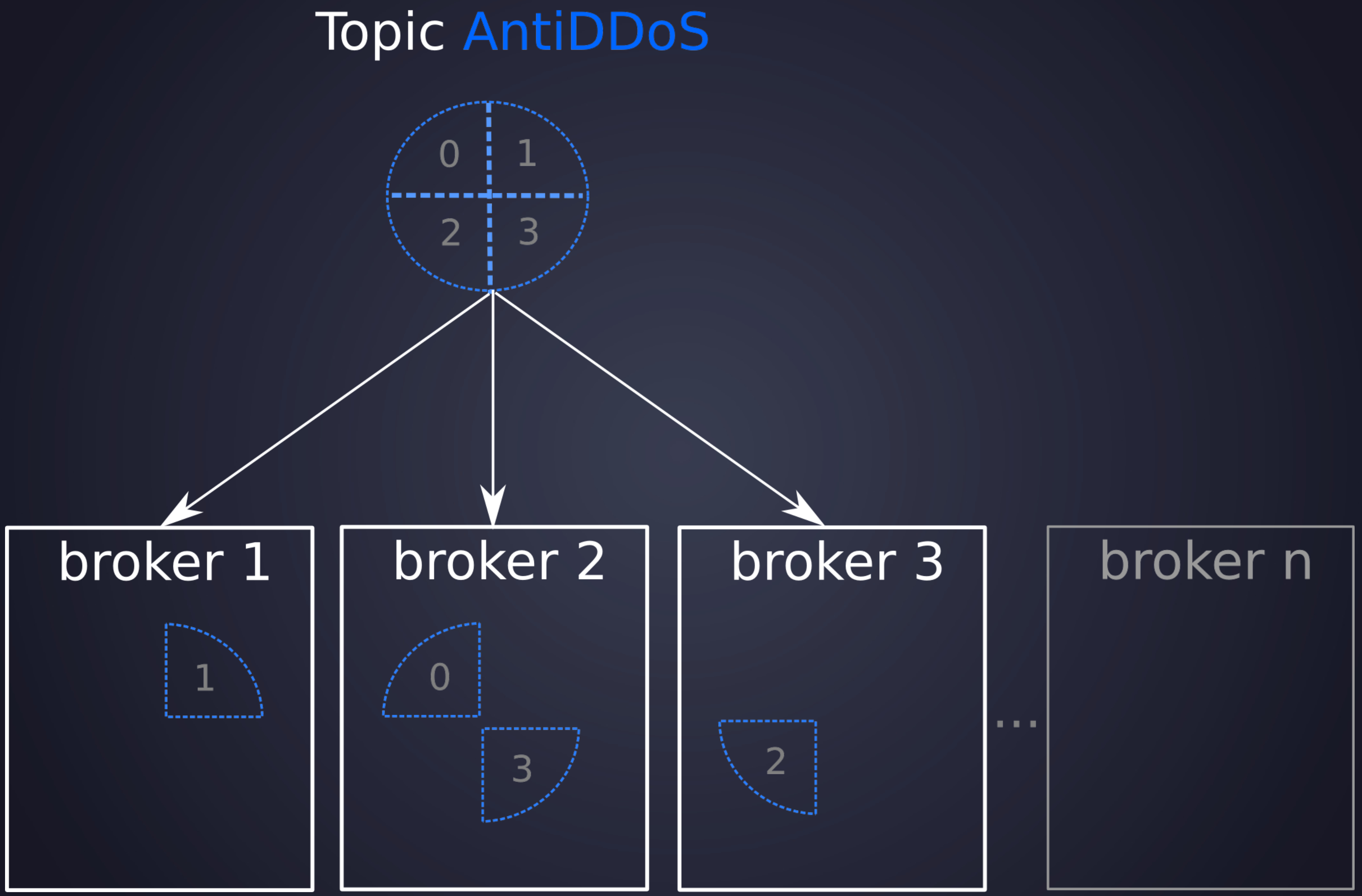
/ kafka


Anatomy of a Topic
Partition 0
Partition 1
Partition 2
0 1 2 3 4 5 6 7 8 9 10 11 12
Writes
Old
New

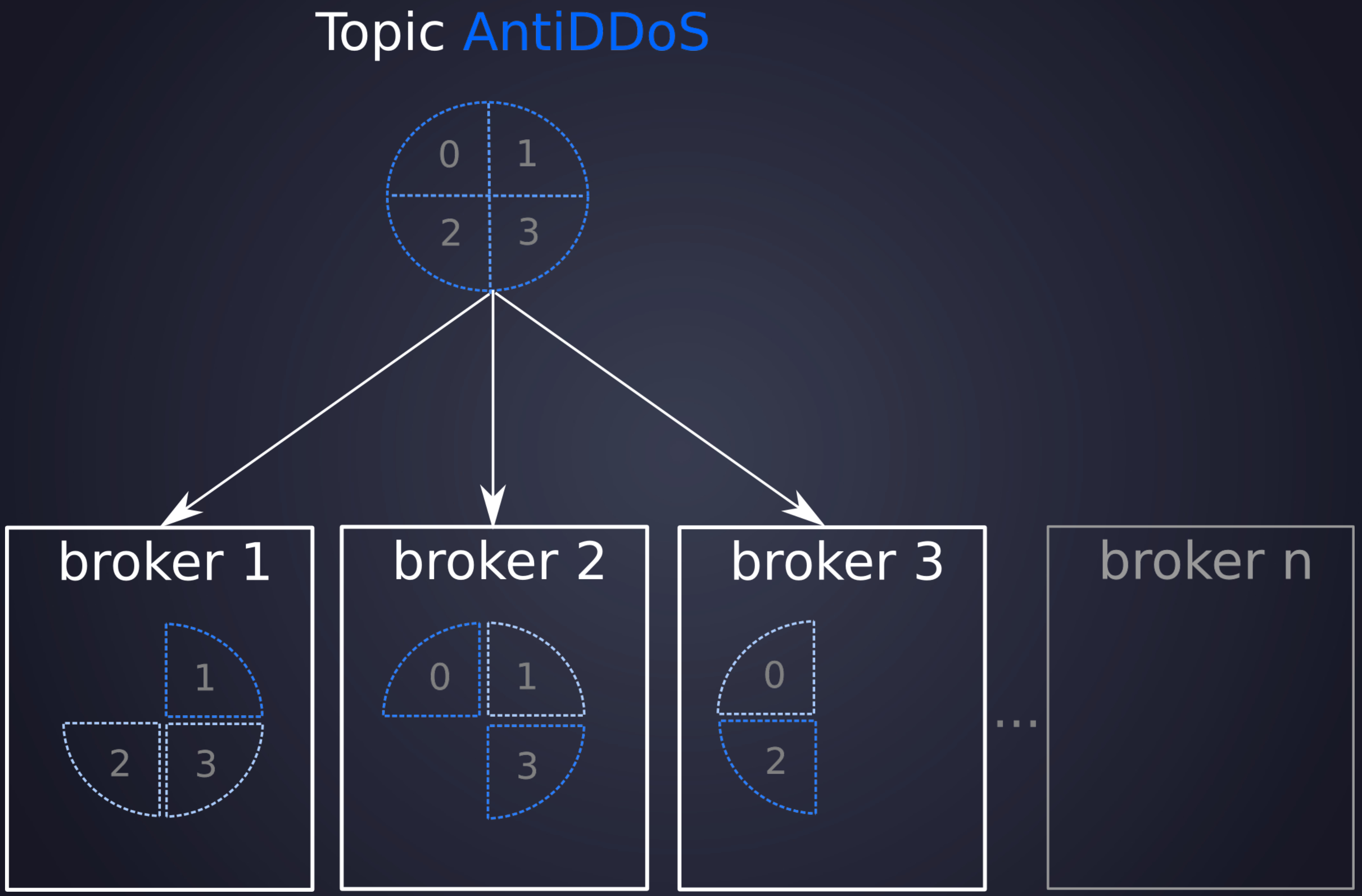
#pubsub
#multicast or #scaleup



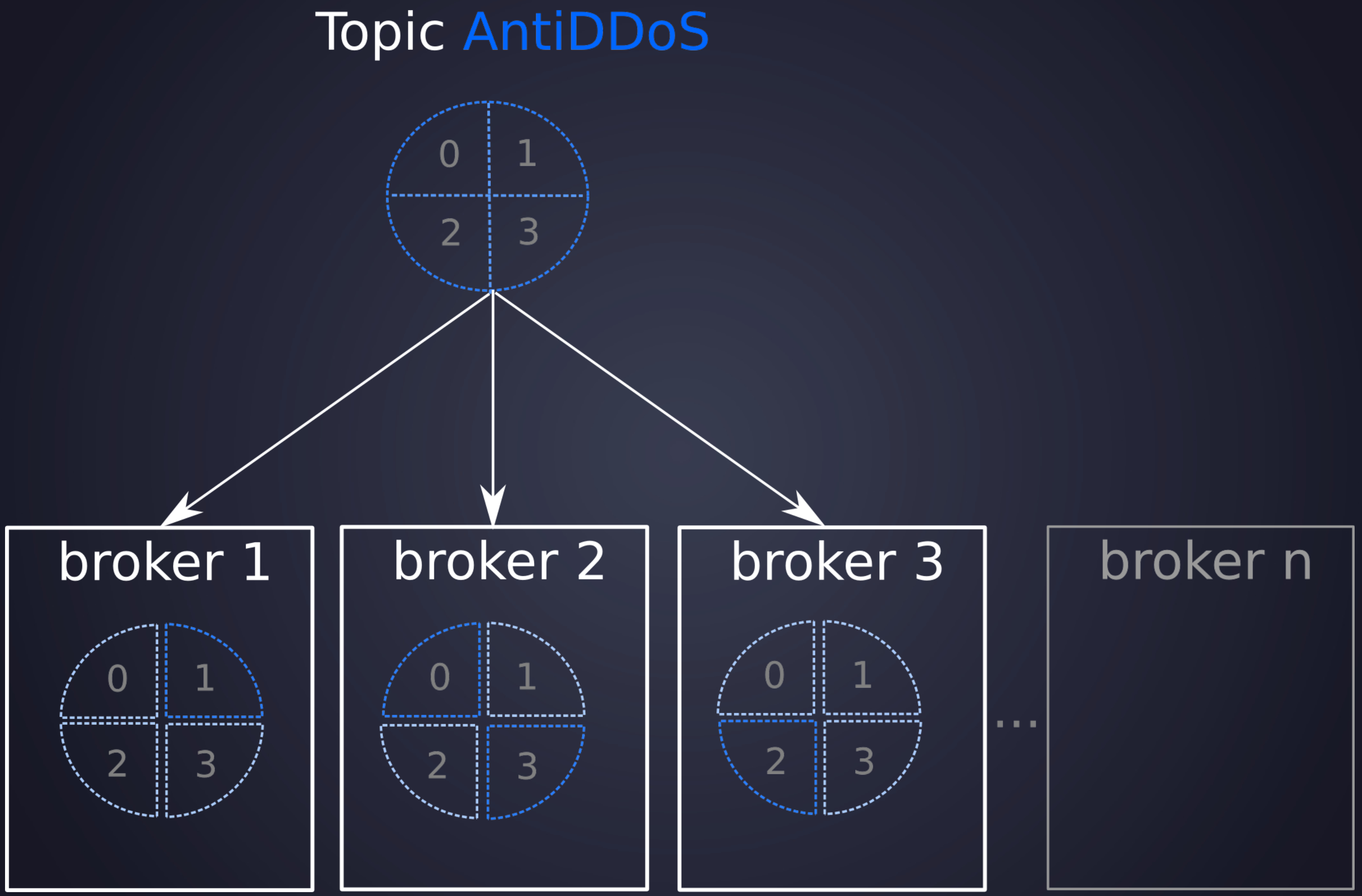
/ kafka / topic



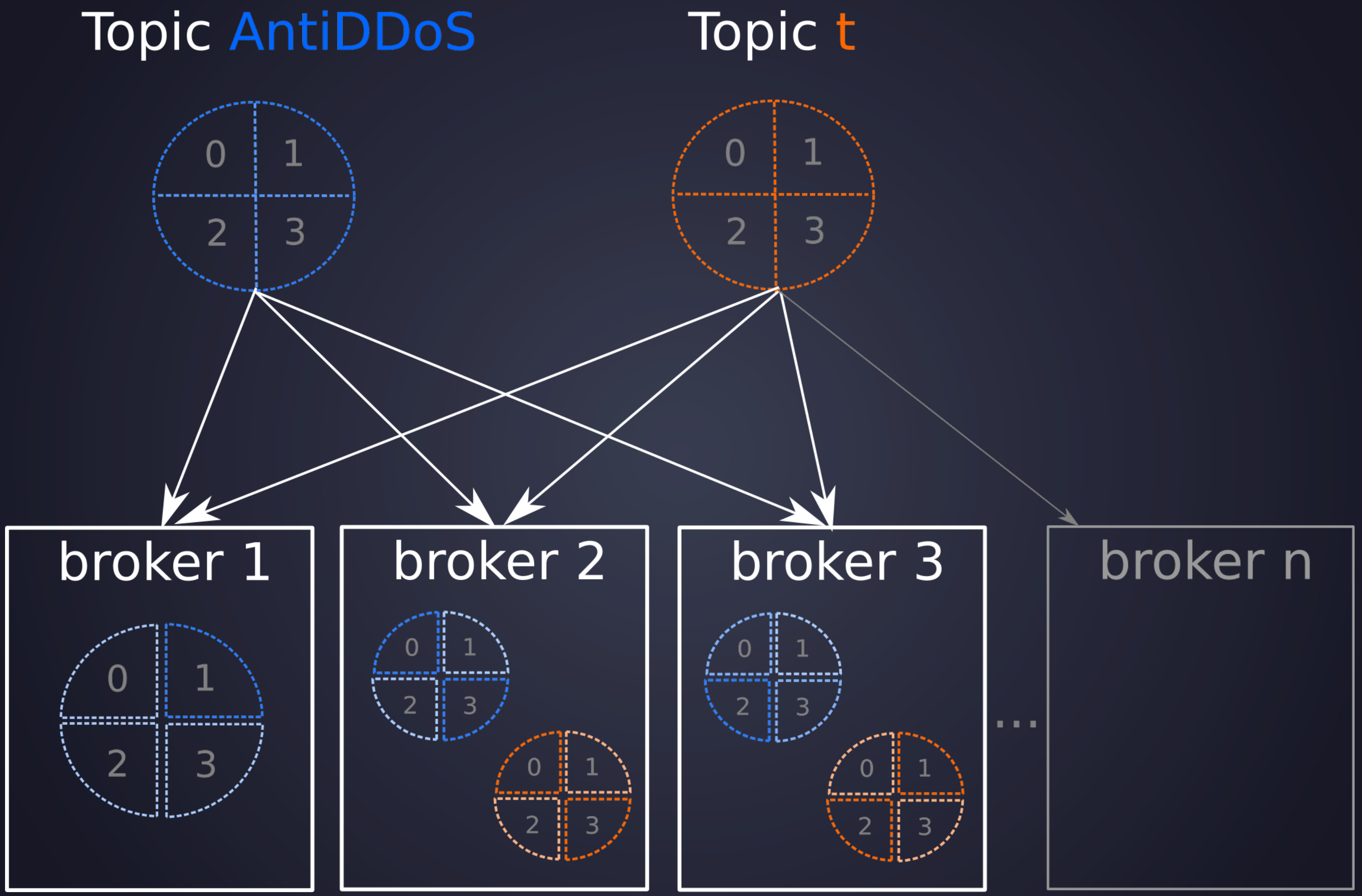
/ kafka / topic / replicas

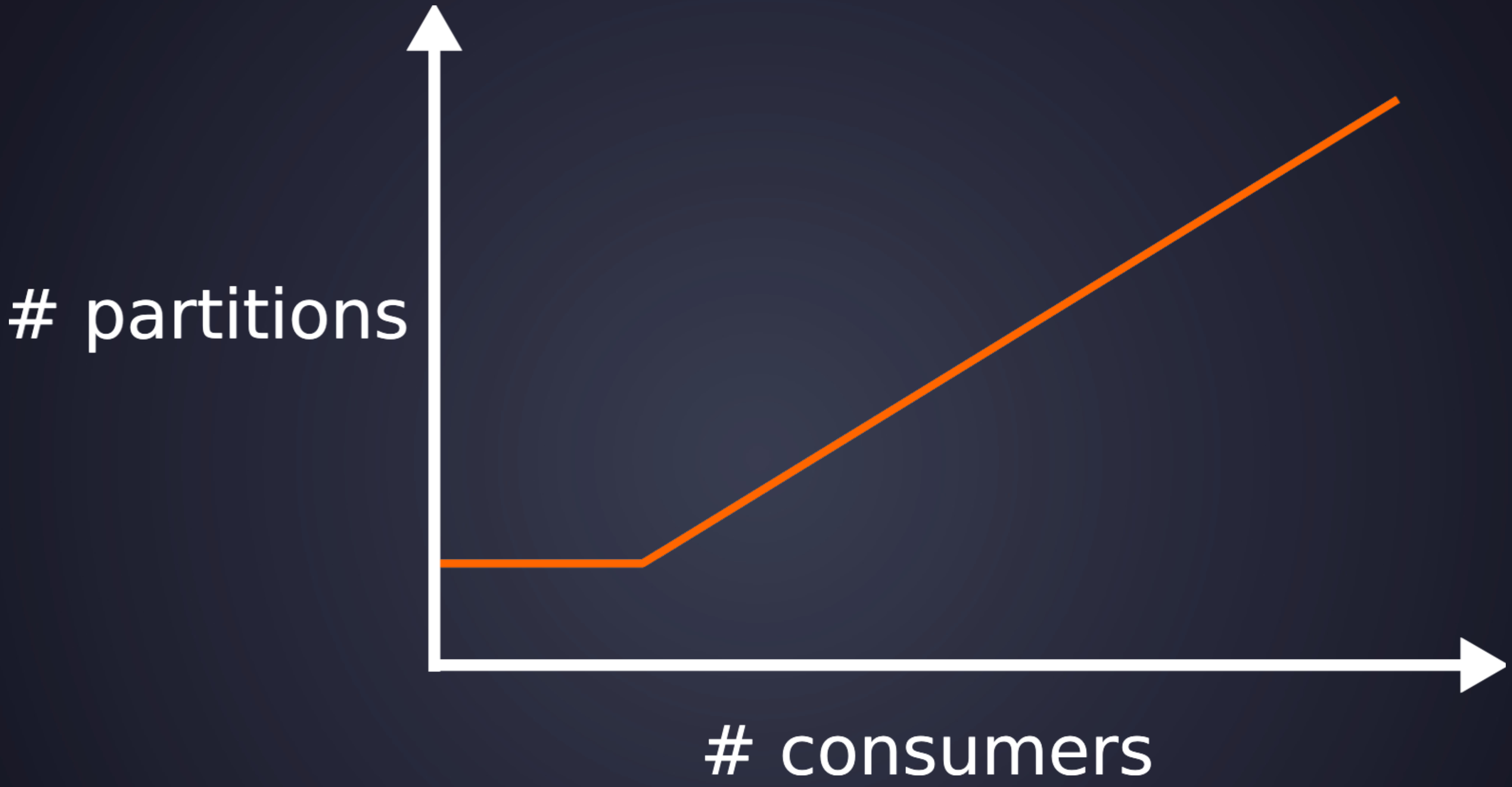


/ kafka / topic / replicas / factor / 3



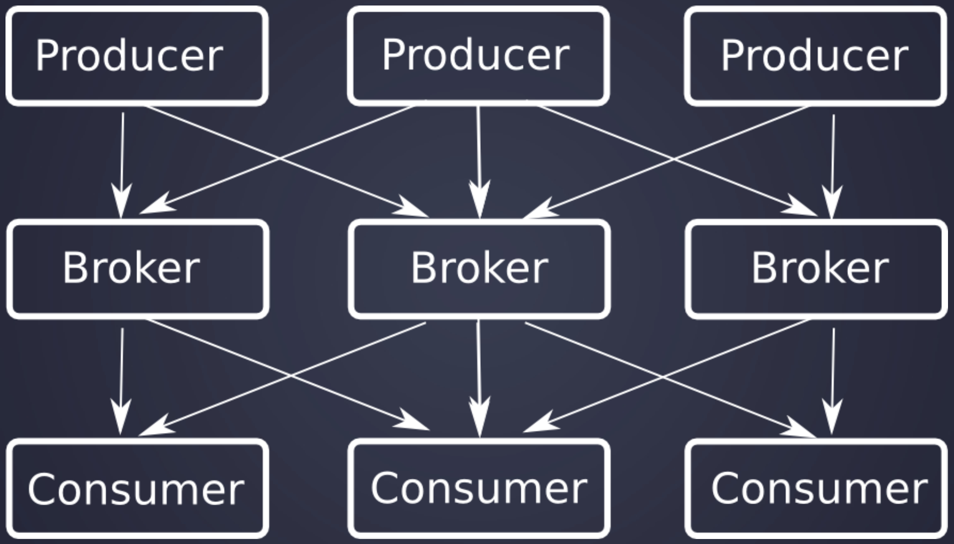
/ kafka / topics



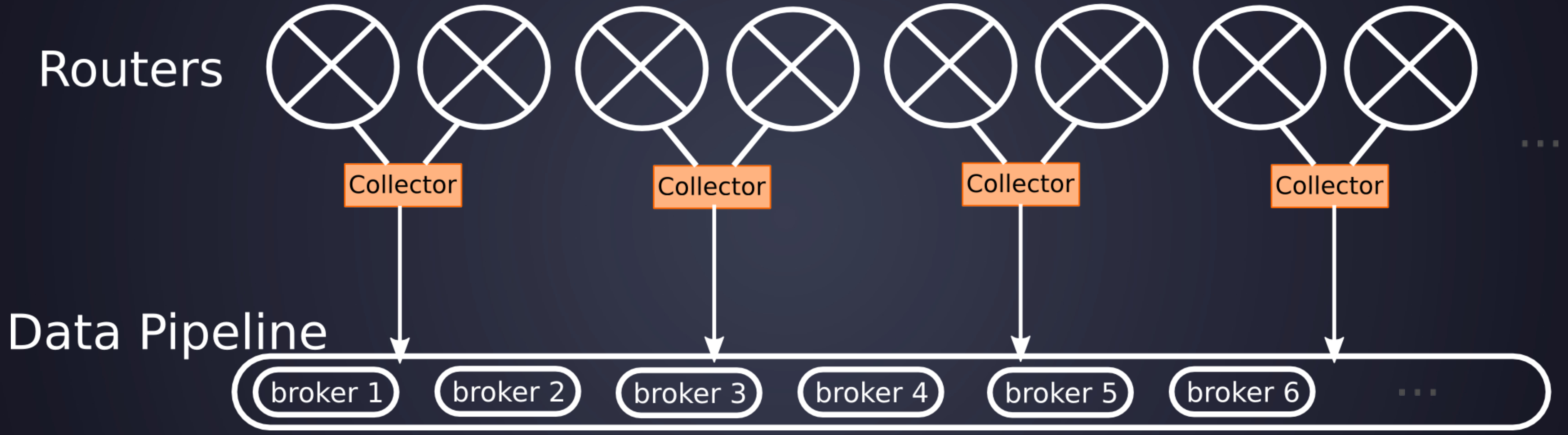


The partition is the unit of scalability

/ kafka


Producer
Producer
Producer
Broker
Broker
Broker
Consumer
Consumer
Consumer

/ kafka / producers



Stream Processing

STORM

/ storm

- **Topology** (DAG)
  - **Spouts**
  - **Bolts**
  - **Tuples**
    - **Fields**
- **Cluster**
  - **Nimbus**
  - **Supervisors**
  - **Workers**

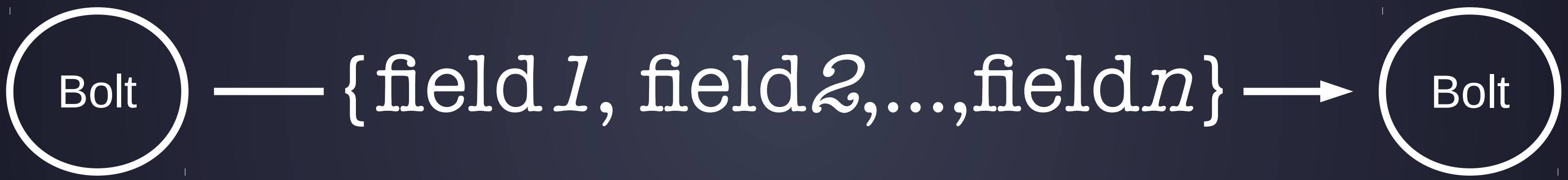
/ storm / tuple

field

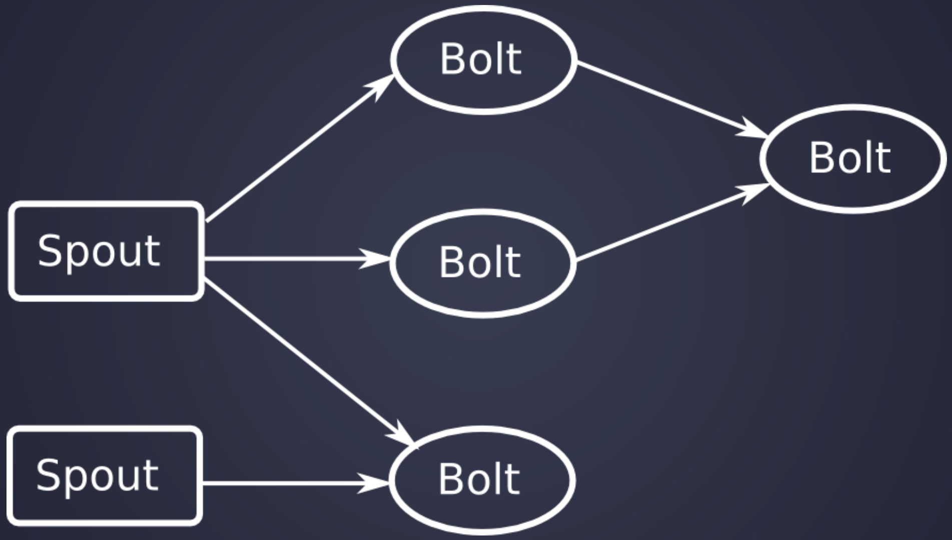
/ storm / tuple

{field*1*, field*2*,...,field*n*}

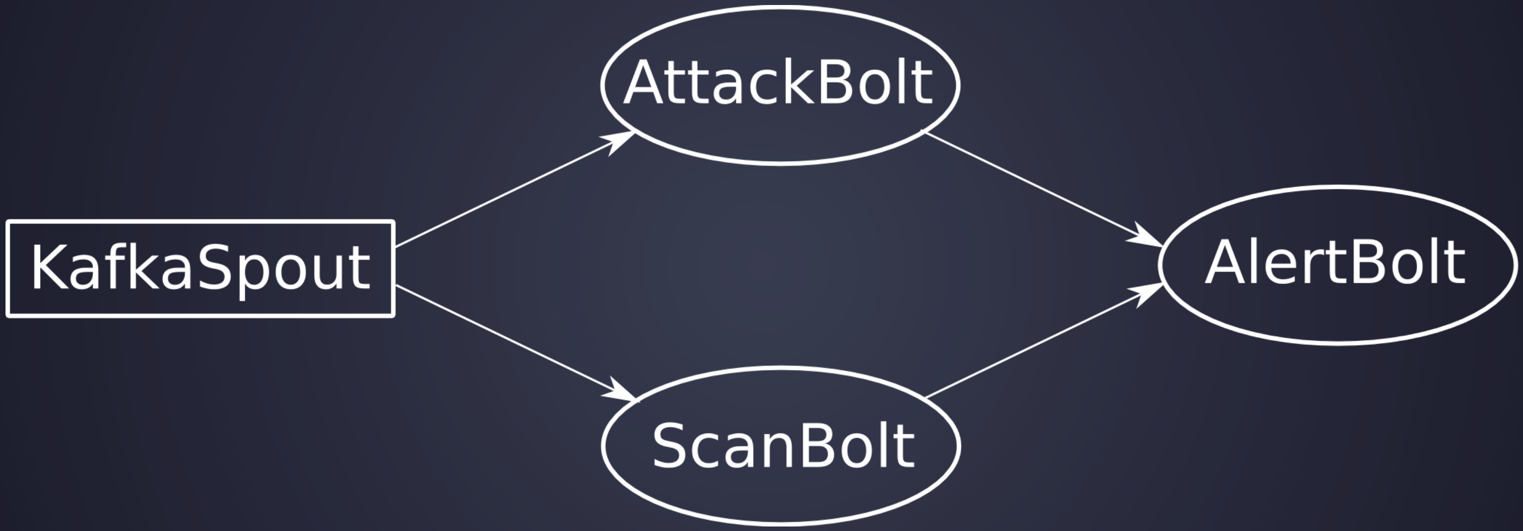
/ storm / tuple



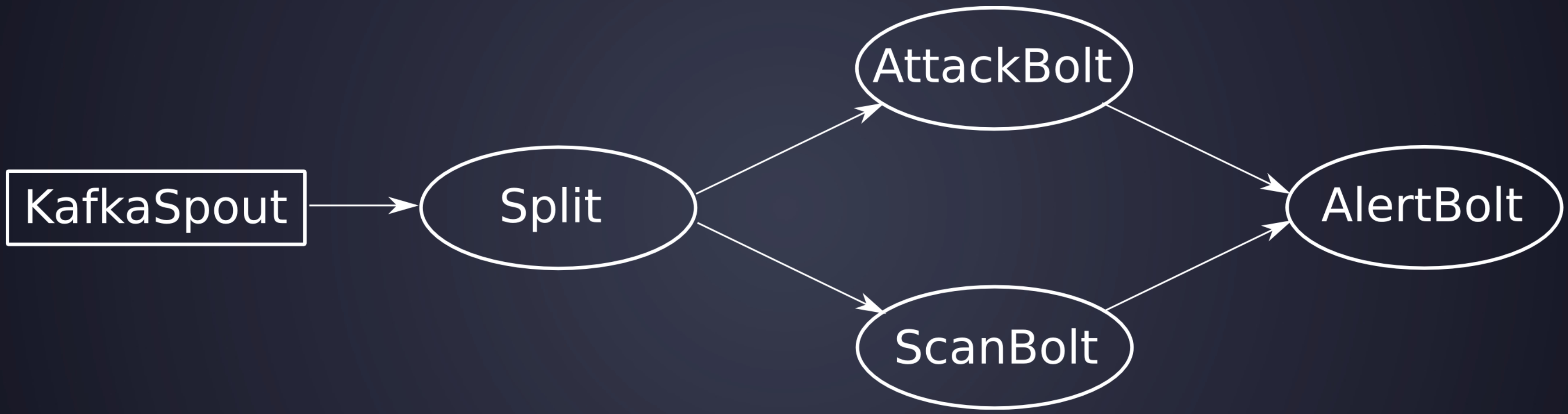
/ storm / topology



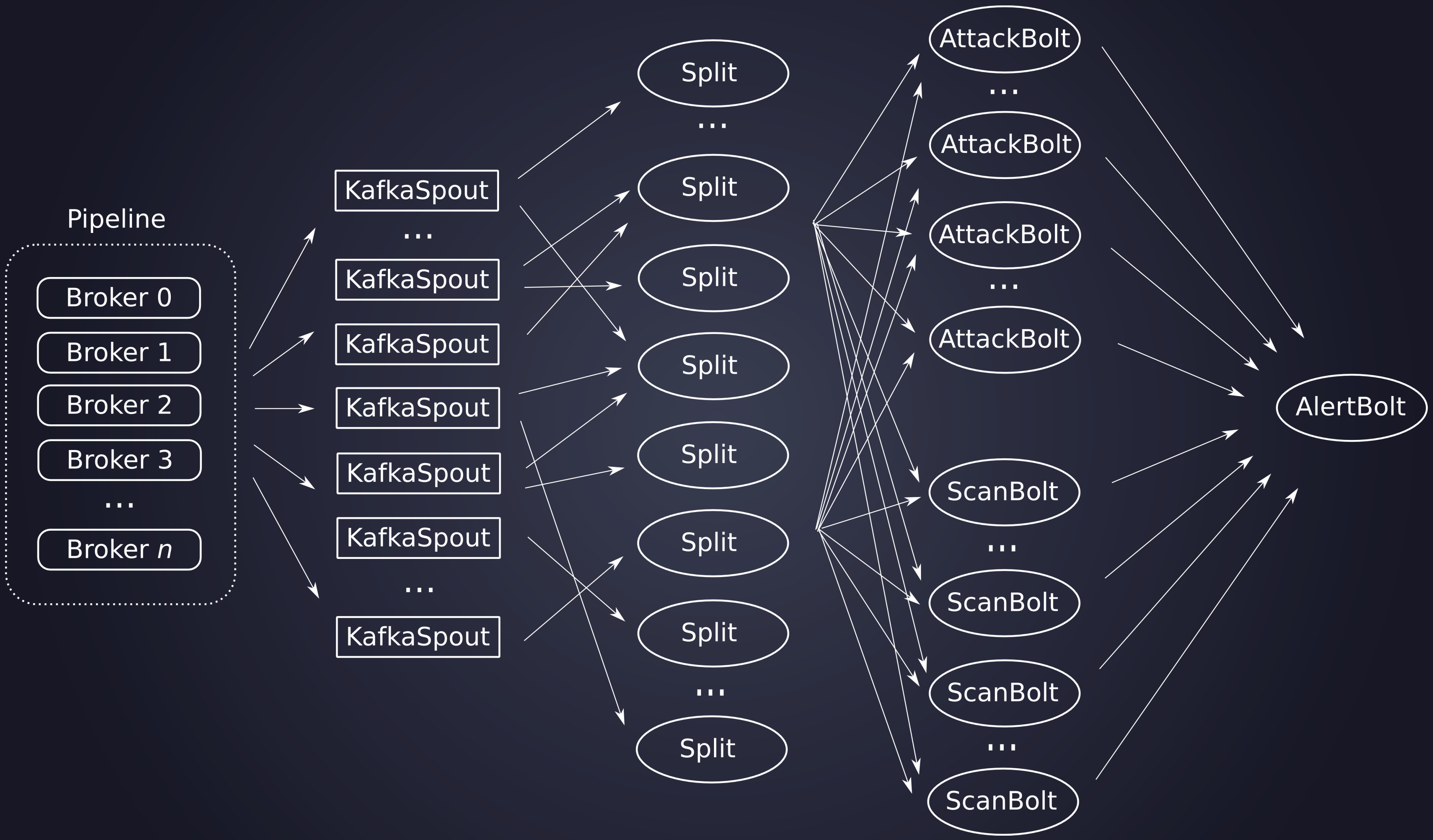
/ storm / topology / antiddos

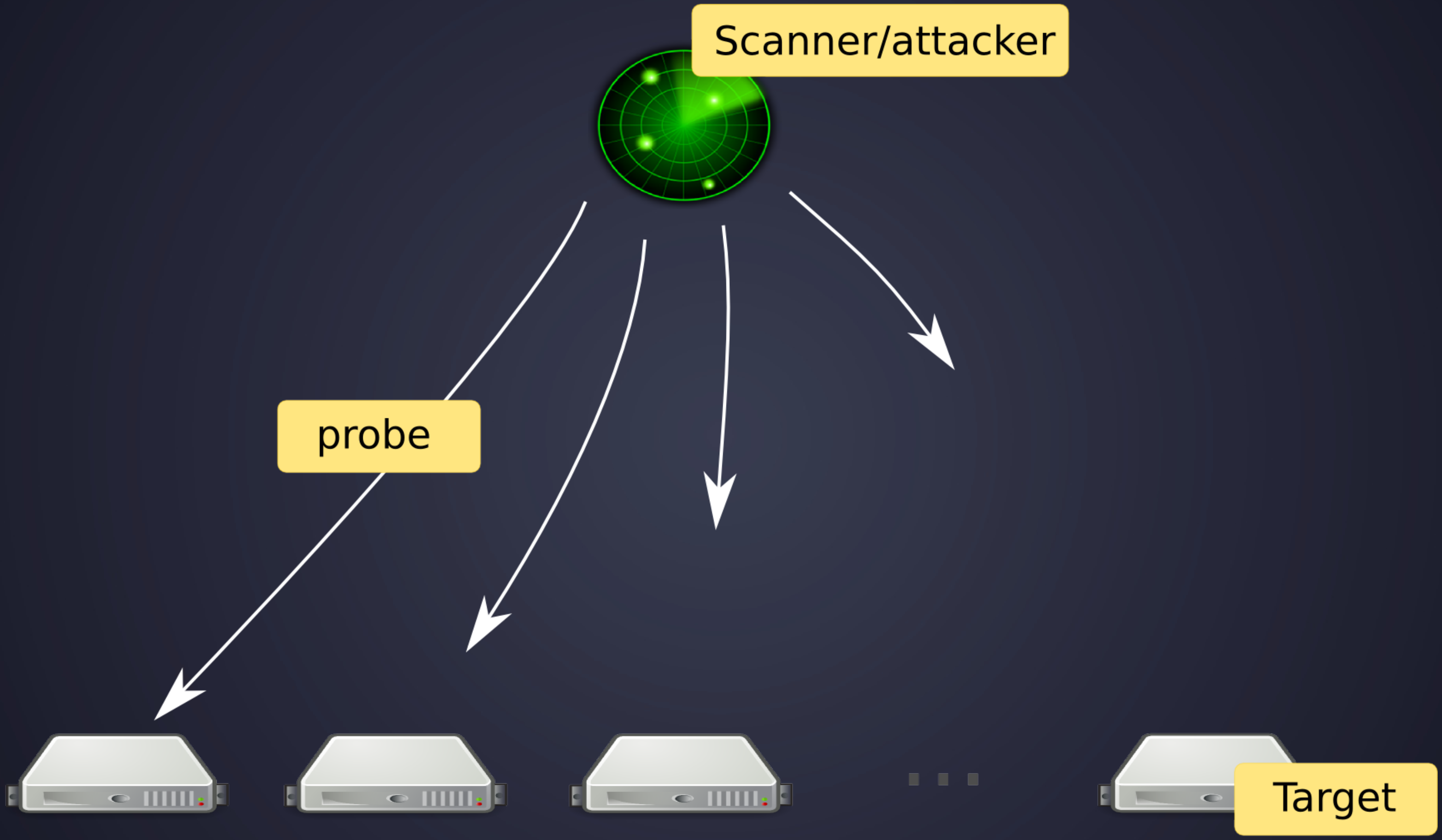


/ storm / topology / antiddos / scale



/ storm / topology / antiddos / scale / parallelism




Scanner/attacker
probe
Target

Stream Grouping

/ storm

Shuffle Grouping

Field Grouping

Direct Grouping

Other Grouping

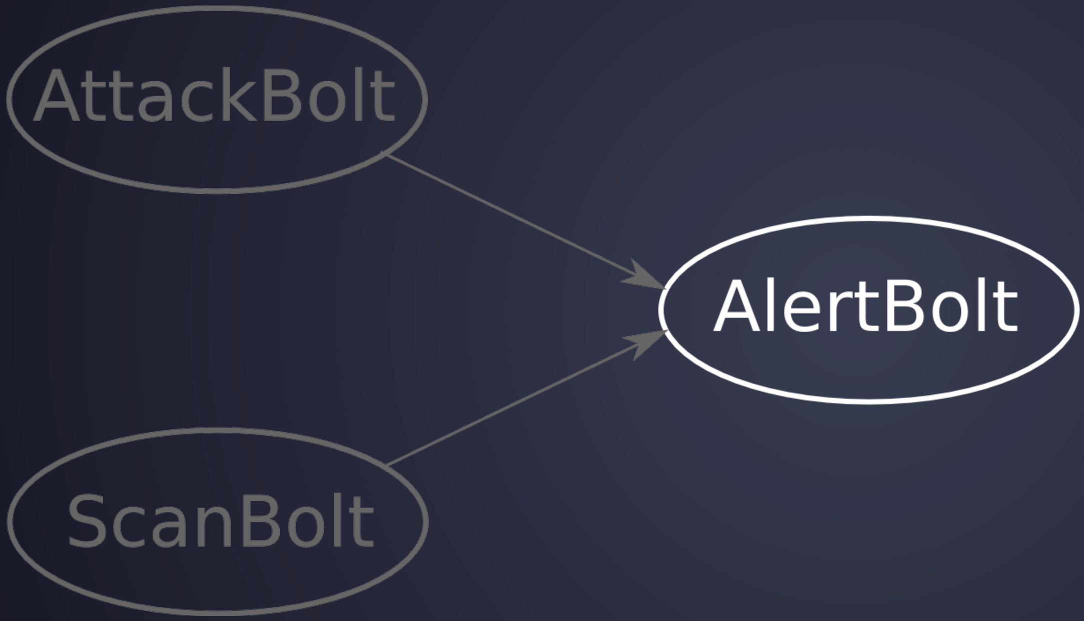
/ storm / grouping

- **Attacks**
  - **Router Grouping**
- **Scans**
  - **IP src Grouping**

/ storm / features

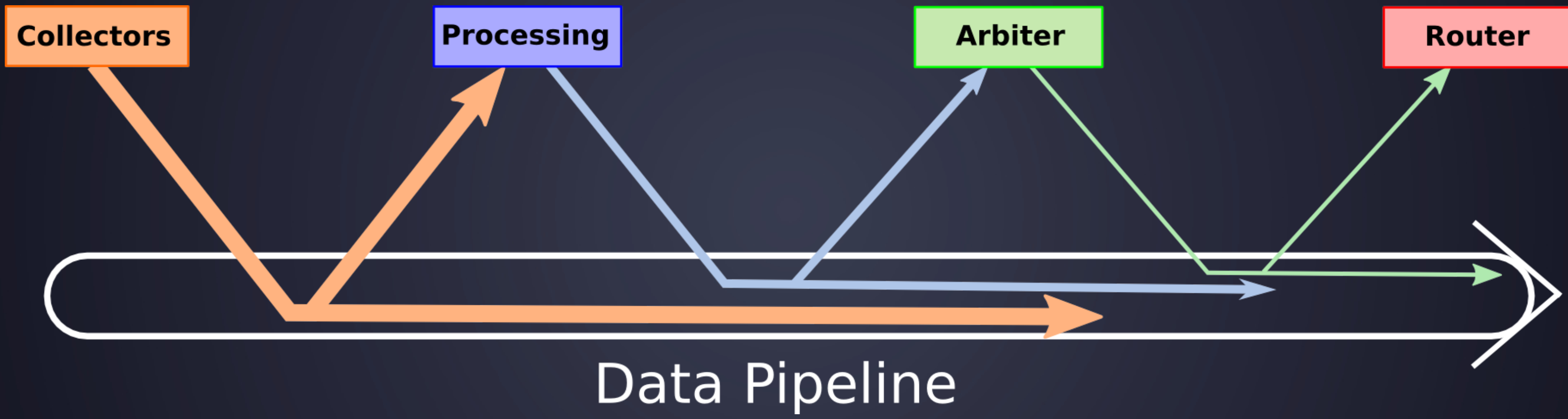
- **Attacks**
  - **≈ 1s detection time**
  - **Scoring with**
    - **Filters**
    - Burst tolerance
- **Scans**
  - **per IP**
  - **per Proto**

/ storm / event

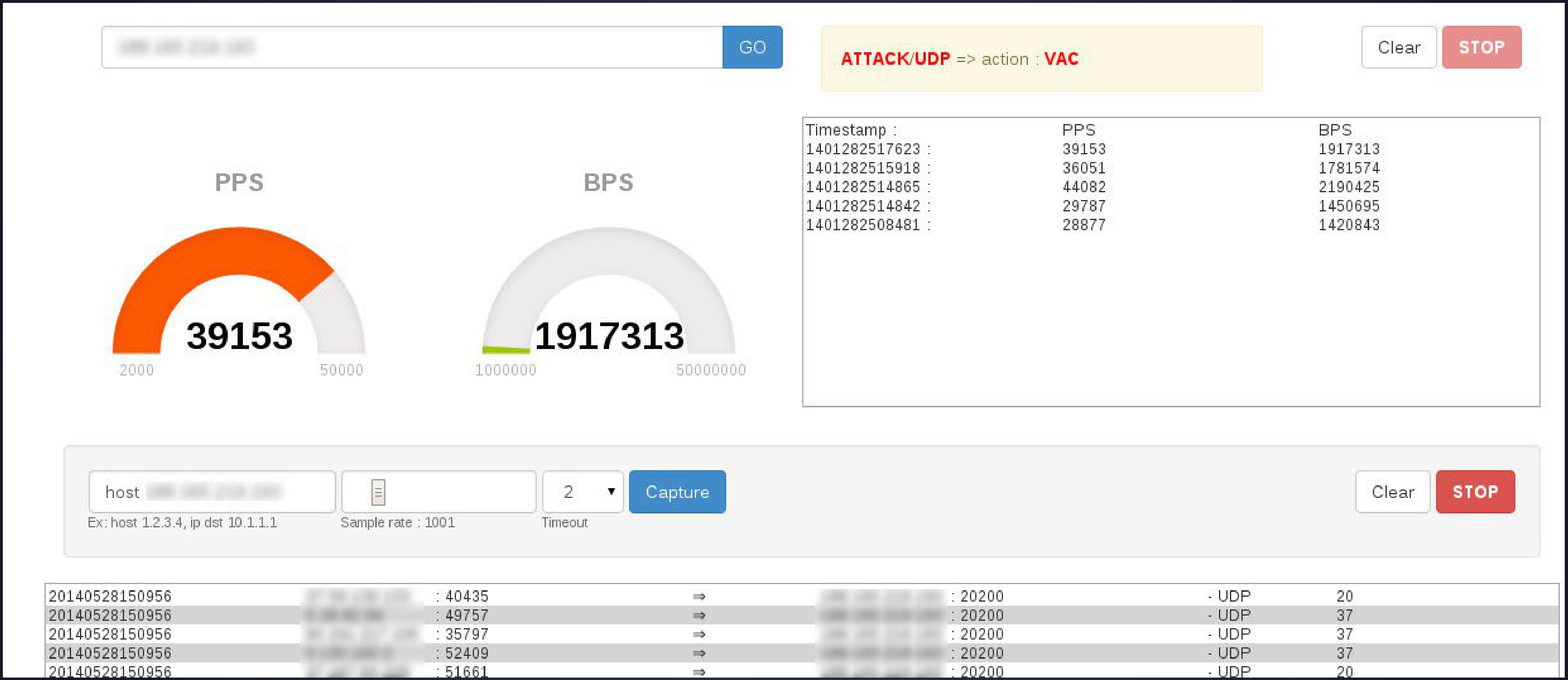


- ☐ **Indexing**
- ☐ **Prooving**
- ☐ **Producing**

#lifecycle



#dataviz



ATTACK/UDP => action : VAC

39153

1917313

Timestamp :	PPS	BPS
1401282517623 :	39153	1917313
1401282515918 :	36051	1781574
1401282514865 :	44082	2190425
1401282514842 :	29787	1450695
1401282508481 :	28877	1420843

Ex: host 1.2.3.4, ip dst 10.1.1.1

Sample rate : 1001

Timeout

```
20140528150956   [redacted] : 40435   ⇒   [redacted] : 20200   - UDP   20
20140528150956   [redacted] : 49757   ⇒   [redacted] : 20200   - UDP   37
20140528150956   [redacted] : 35797   ⇒   [redacted] : 20200   - UDP   37
20140528150956   [redacted] : 52409   ⇒   [redacted] : 20200   - UDP   37
20140528150956   [redacted] : 51661   ⇒   [redacted] : 20200   - UDP   20
```

Nice speech...

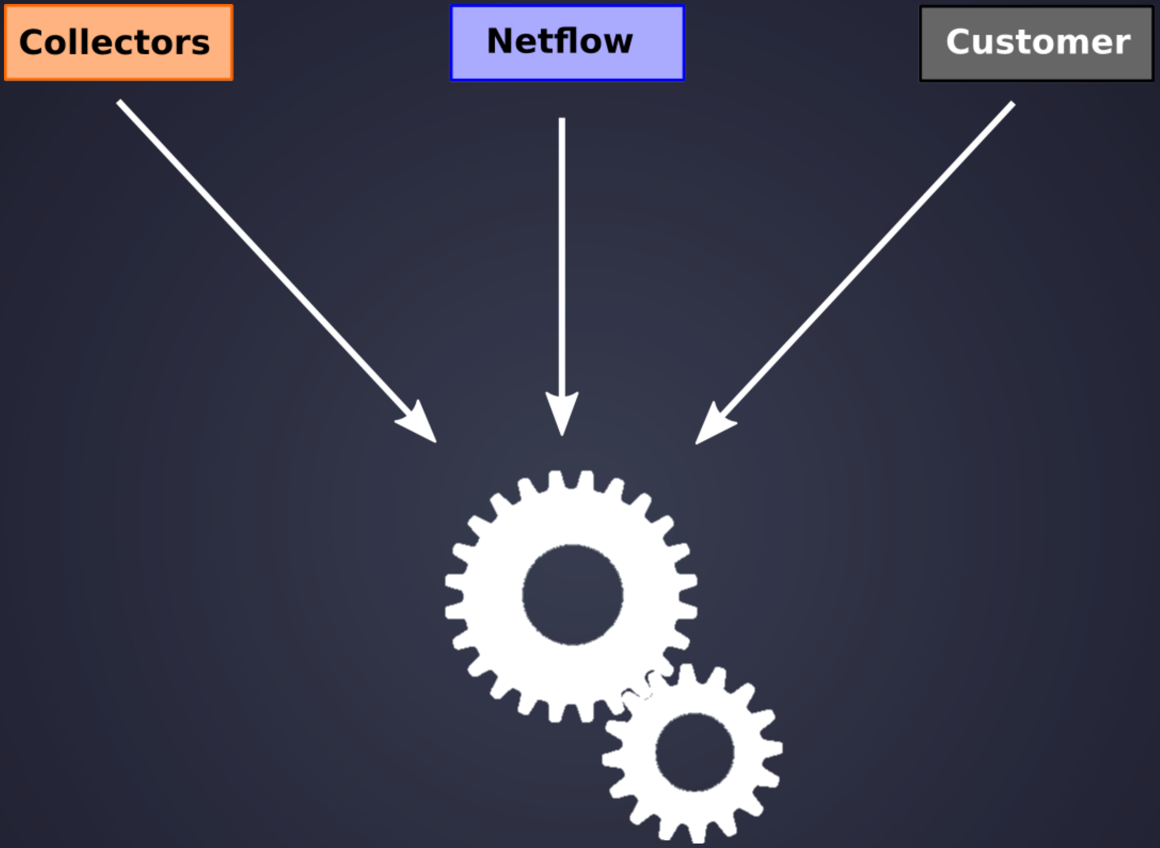
... so what ?

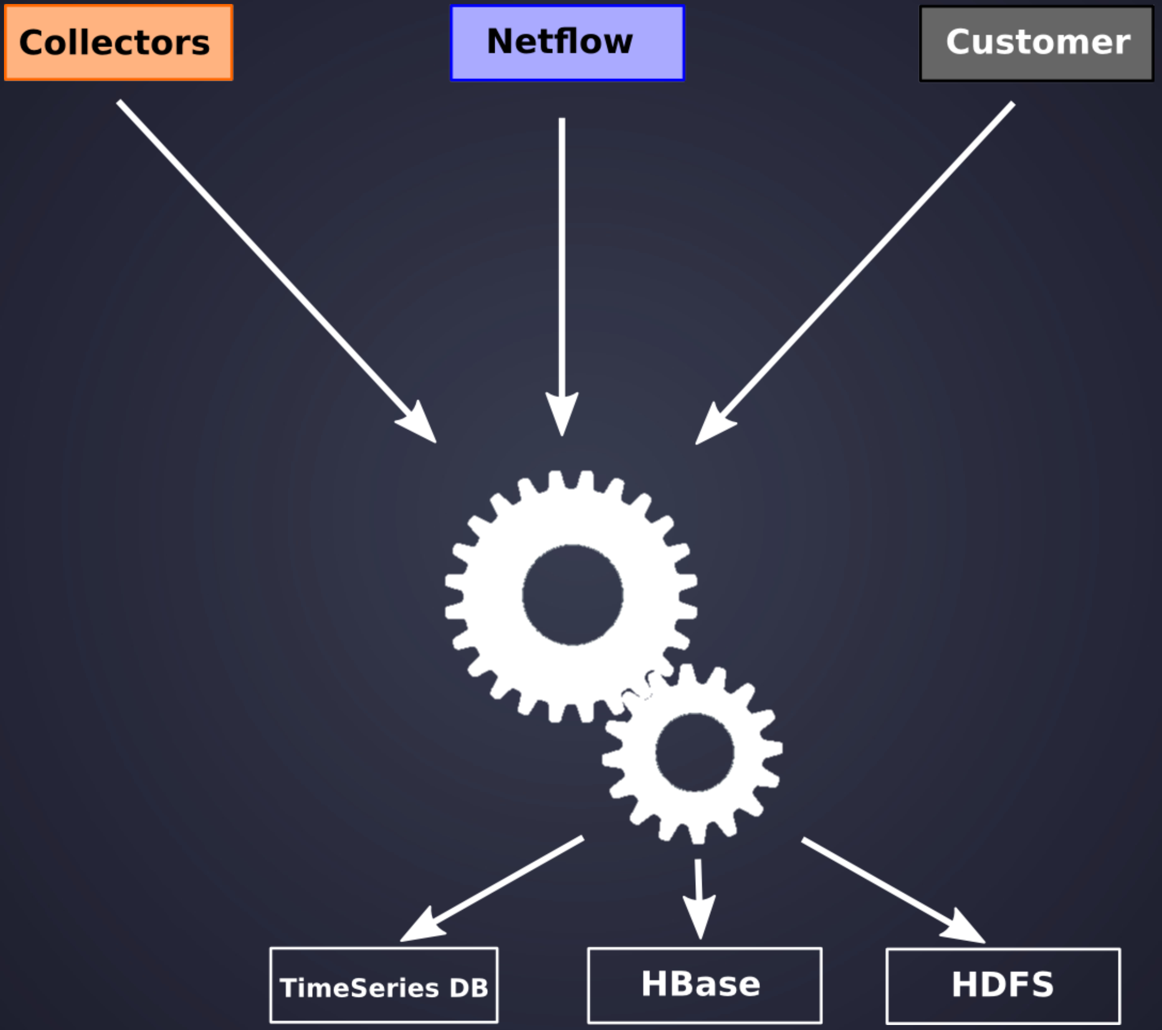
#issues

- **False positives**
- **Strange behaviours from customers**
  - e.g. DB sync without connection pool
- **Application centric**
  - i.e. UDP protocols

#solutions

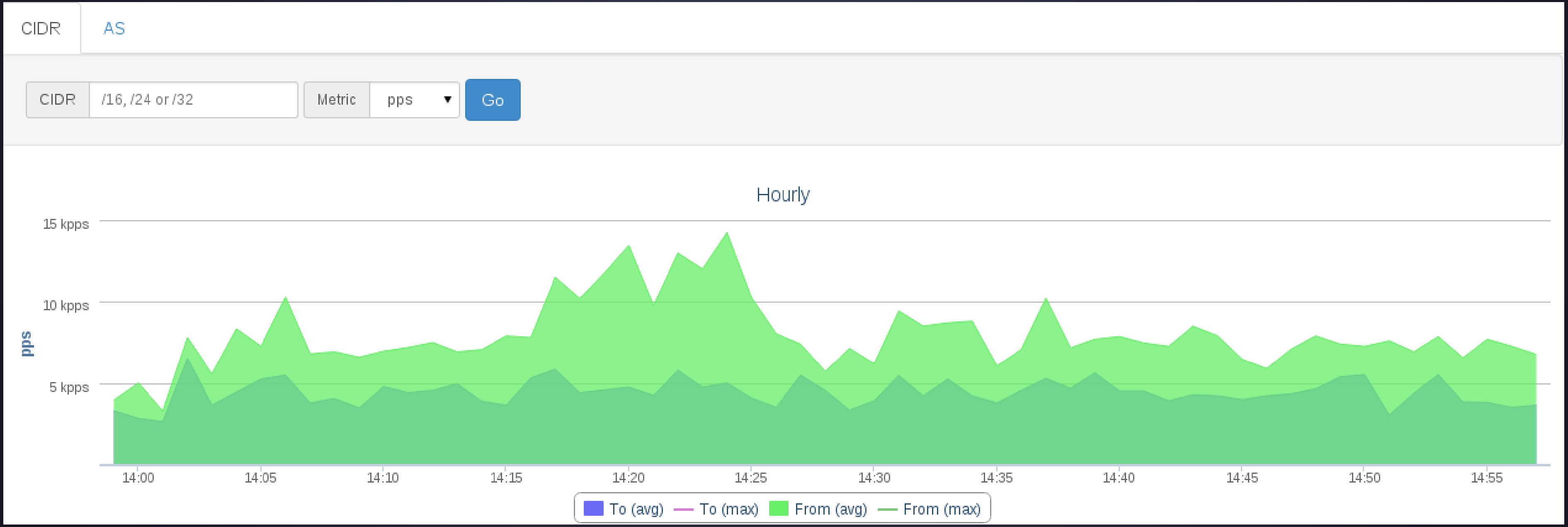
- Add other sources
- Application Anti-DDoS
  - Game
    - Half Life/Source
    - CS:GO
    - TeamSpeak / Mumble
    - GTA
    - SA:MP
    - …
  - More to come (any special need ?)


Collectors
Netflow
Customer



#datalake

#aggregations



#hardware

- **Nodes - Hardware**
  - **CPU 16c/32t**
  - **RAM 256GB**
  - **Disks :**
    - OS : Raid 1
    - Data : 10 disks
  - **per node**
    - 200 MB/s ~ 1,5-2 Gbps

#config

- **Storm**
  - **CPU/RAM bound**
  - **M+ tuples/s**
  - **No ackers**
  - **Break SRP**
  - **Minimal workers**
    - Avoid transfer buffer

- **Kafka**
  - **I/O bound**
  - **Bench (1node)**
    - 1M+ msg/s
  - **No compression**
  - **No ackers**
  - **80MB/s**
  - **Tuning**
    - num.io.thread
    - num.network.thread
    - socket.*.buffer.*

OpenSOC

#Thanks

- **Clément Sciascia** - @csciasci
- **Magnus Edenhill** - @edenhillm
  - https://github.com/edenhill/librdkafka
- **LinkedIn** - Apache Kafka
- **Nathan Marz** - Apache Storm

#more

- **Storm basic training** – Mickael G. Noll
  - http://fr.slideshare.net/miguno/apache-storm-09-basic-training-verisign
- **Kafka documentation & basic Training** – Mickael G. Noll

Thank you !

@StevenLeRoux
steven.le-roux@ovh.net

Appendices

Sample Producer

```
from kafka import SimpleProducer, KafkaClient, KafkaConsumer

kafka = KafkaClient("localhost:9092")
producer = SimpleProducer(kafka)


producer = SimpleProducer(kafka,
   async=False,
   req_acks=ACK_AFTER_(LOCAL_WRITE/CLUSTER_COMMIT),
   ack_timeout=2000,
   batch_send...)


producer.send_messages("topic", "message")
```

Sample Consumer

```
from kafka import SimpleProducer, KafkaClient, KafkaConsumer

kafka = KafkaClient("localhost:9092")

consumer = KafkaConsumer(
  "topic",
   group_id="groupid",
   metadata_broker_list=["localhost:9092"]
)


for message in consumer:
   print(message)
```

Sample Topology

```
TopologyBuilder builder = new TopologyBuilder();
builder.setSpout("integers", new genInteger(), 10);
builder.setBolt("print", new DoubleAndTrippleBolt(), 3)
        .shuffleGrouping("integers");
```

Sample Bolt

```
public class DoubleAndTripleBolt extends BaseRichBolt {
    private OutputCollectorBase _collector;

    @Override
    public void prepare(Map conf, TopologyContext context,
OutputCollectorBase collector) {
        _collector = collector;
    }

    @Override
    public void execute(Tuple input) {
        int val = input.getInteger(0);
        _collector.emit(input, new Values(val*2, val*3));
        _collector.ack(input);
    }

    @Override
    public void declareOutputFields(OutputFieldsDeclarer declarer) {
        declarer.declare(new Fields("double", "triple"));
    }
}
```

http://ovh.careers