Herd of Containers

BlaBlaCar

Saâd DIF

Database Engineer

“Hello

Herd of Containers: PostgreSQL in containers at BlaBlaCar

pgDay Paris, Mar 15, 2018

Today's agenda

01
BlaBlaCar Overview

02
PostgreSQL usage at BlaBlaCar

03
Switching to a new implementation

01

BlaBlaCar Overview

Facts and Figures

60 million members

30 million mobile app downloads
Iphone and Android

Founded in 2006

15 million travellers

1 million tonnes less CO_2
In the past year

Currently in 22 countries
France, Spain, UK, Italy, Poland, Hungary, Croatia, Serbia, Romania, Germany, Belgium, India, Mexico, The Netherlands, Luxembourg, Portugal, Ukraine, Czech Republic, Slovakia, Russia, Brazil and Turkey.

Core Data Ecosystem

1

MySQL

Main Database
MariaDB 10.0+
Galera Cluster

2

Cassandra

Column Oriented
Distributed

3

Redis

In Memory
Key-Value
Optional durability

Core Data Ecosystem

4

ElasticSearch

JSON documents
FullText search
Distributed

5

PostgreSQL

ORDBMS
Extensibility
Stability

Containers

Why Containers ?

Resource allocation
Deployment Speed

On premise

Skills already there
Cost

Containers

Rkt

Why Rkt over Docker ?

CoreOS Container Linux

Linux Distrib
Simple & Secure
Only run containers

Fleet

Orchestration
By default with
CoreOS

Containers

GGN

Generate systemd units

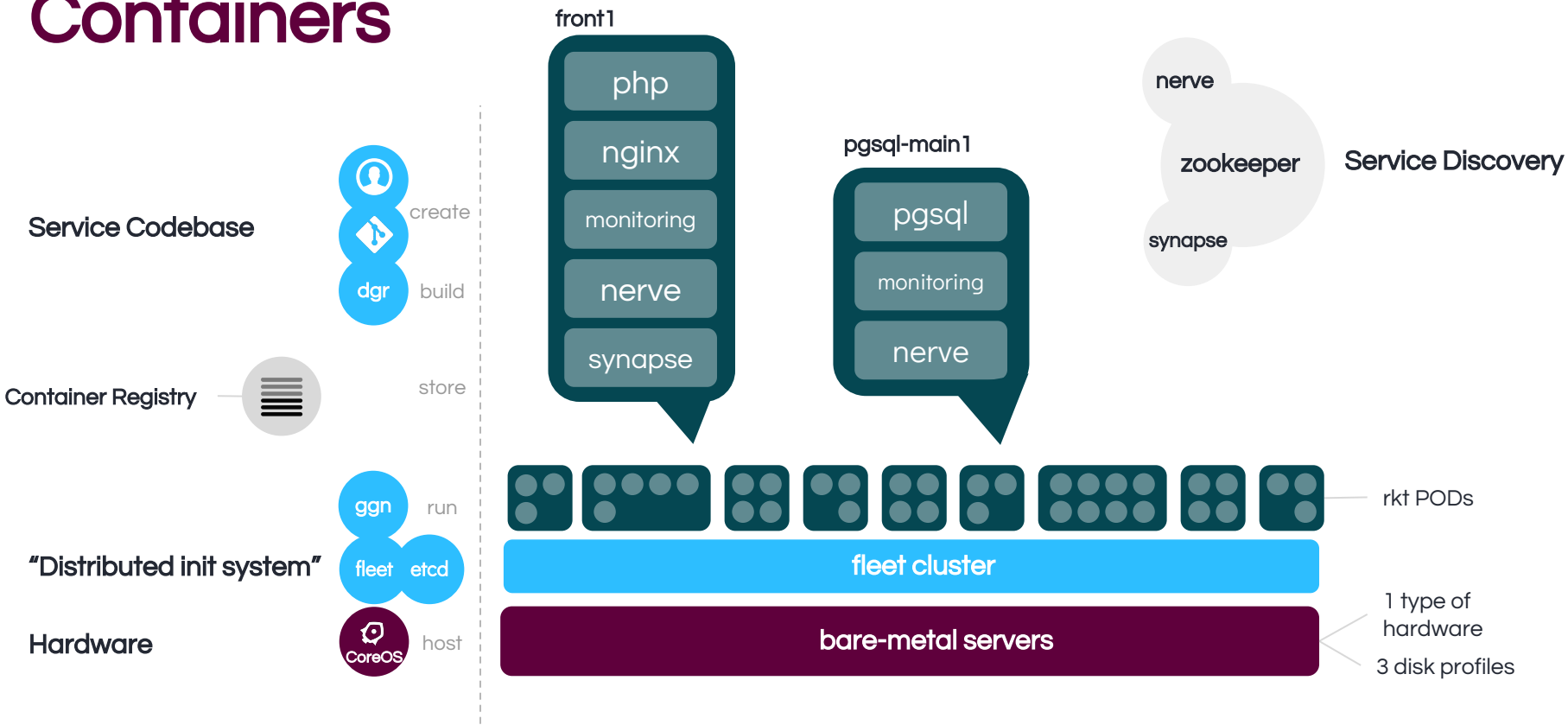
Dgr

Build and configure App Container Images

Pods

Aggregate images in one shared environment

Containers


front1
php
nginx
monitoring
nerve
synapse
pgsql-main1
pgsql
monitoring
nerve
nerve
zookeeper
synapse
Service Discovery
Service Codebase
create
dgr
build
Container Registry
store
ggn
run
"Distributed init system"
fleet
etcd
Hardware
CoreOS
host
rkt PODs
fleet cluster
bare-metal servers
1 type of hardware
3 disk profiles

Service Discovery

Why ?

1

Get rid of DNS internally
Adapt to change

Service Discovery

Why ?

Zookeeper

1 —— 2 ——

Key-Value store
Reliable, Fast, Scalable

Service Discovery

Why ?

Zookeeper

Report

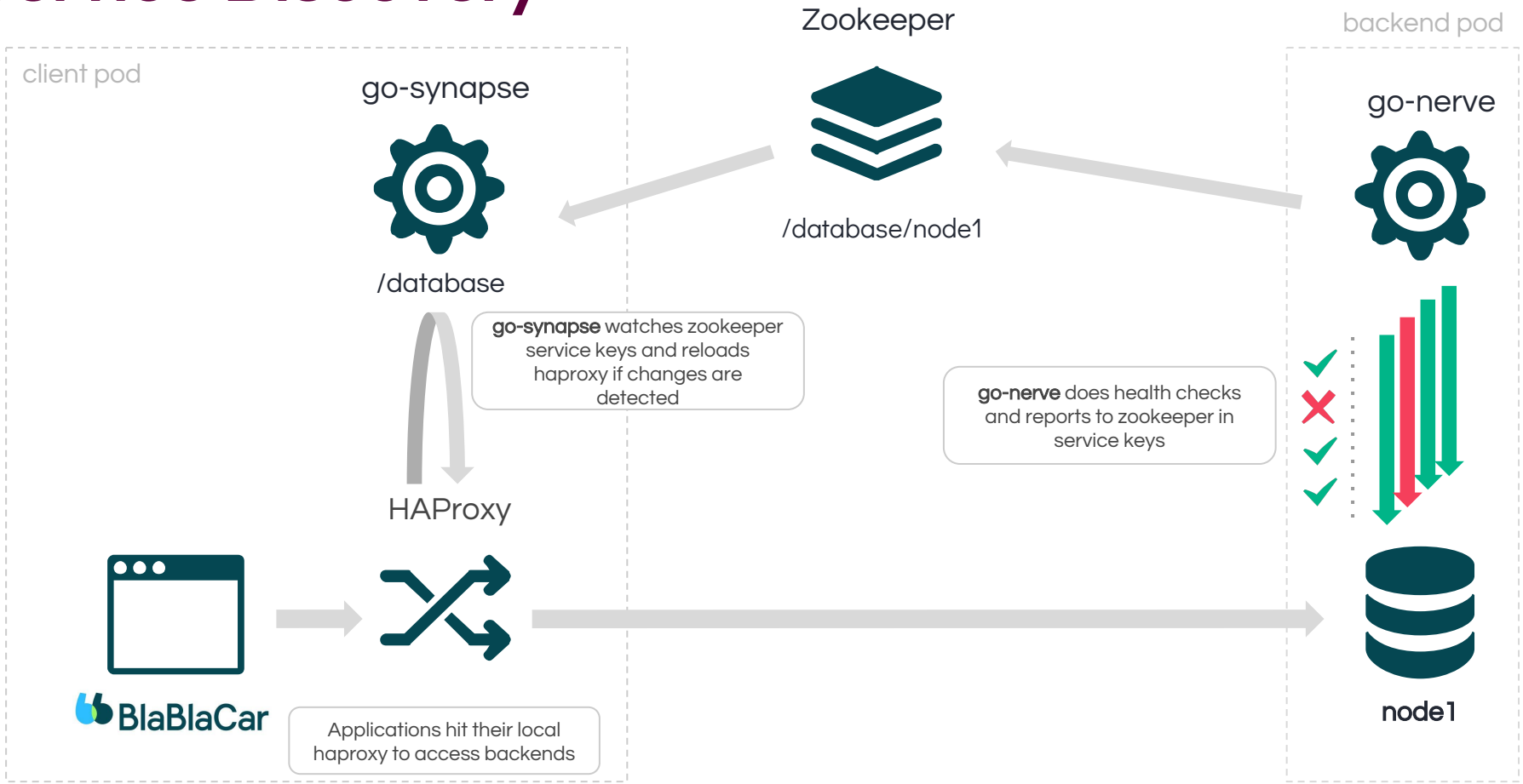
1 — 2 — 3 —

Go-Nerve
Health Checks
Ephemeral keys
Present on each pod

Service Discovery

Go-Synapse
Watch Zookeeper
Update HAProxy configuration

Service Discovery



02

PostgreSQL usage at BlaBlaCar

Usage

Third-party applications

Prerequisite

Home Made tools

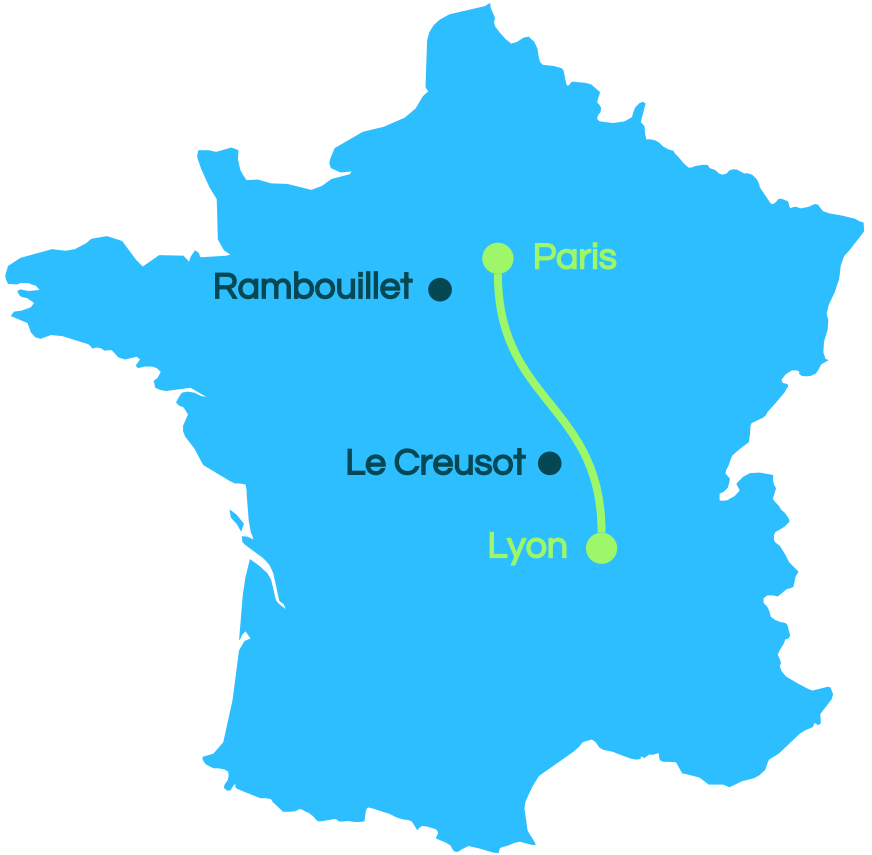
Confidence

Spatial

PostGIS

PostGIS

- Travel company
- Corridoring
- Point to Point



3 685

Rides passed by Amiens last month

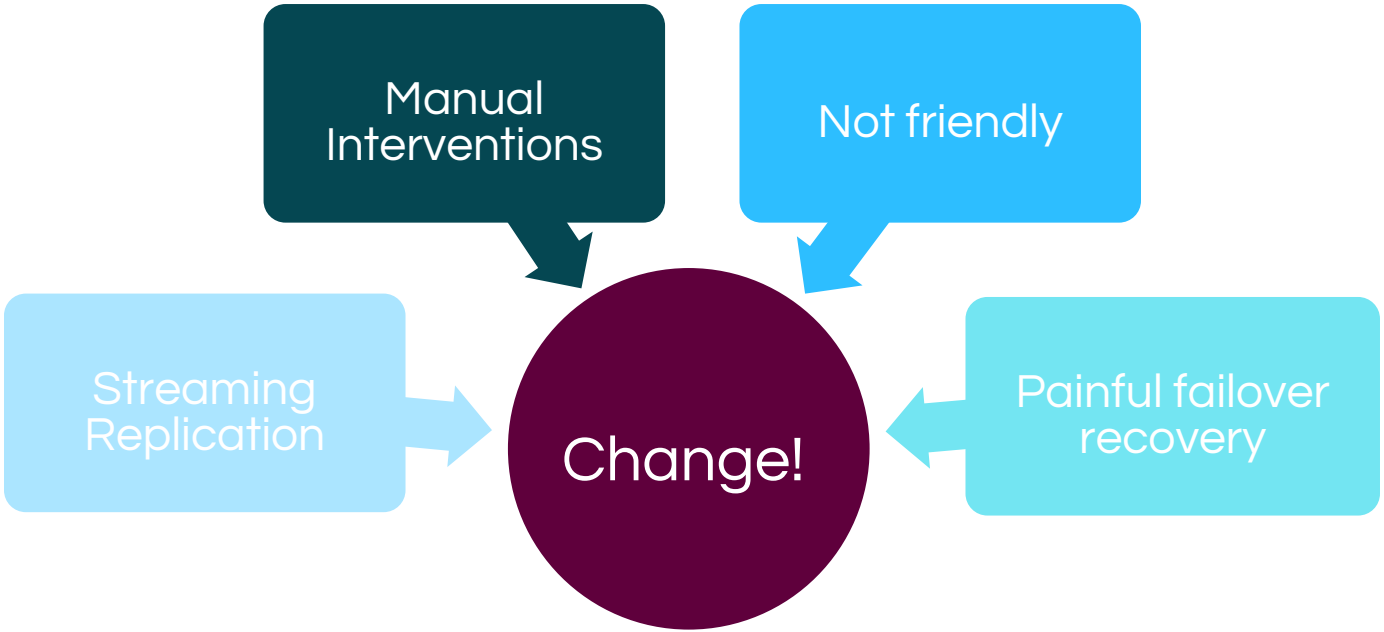
1M

Number of meeting points

50k

Rows reads per minutes

Operate


Manual Interventions
Not friendly
Streaming Replication
Change!
Painful failover recovery

Target

- Scale writes
- Ease deployments
- Maximum availability
- Expandable resources

- Slaves
- Failovers

Possibilities

Postgres-XC (x2)

Postgres-XL

PgLogical

Bucardo

Slony

Londiste

Switching to a new implementation

BDR

- Bi-Directional Replication
- OpenSource project by 2ndQuadrant
- Multi Master Asynchronous Replication
- 2 to 48 nodes
- Optimal for Geo Distributed databases

BDR : The Confirmation

- All nodes support reads and writes
- No failovers
- No other process / nodes needed
- Partition tolerant

BDR : Caveats

- ⊗ Modified version of PostgreSQL 9.4
  BDR 2.0 with PostgreSQL 9.6 for 2ndQuadrant support customers
- ⊗ Replication lag
- ⊗ Conflicts
- ⊗ DDL lock
- ⊗ Statement not replicated
- ⊗ Some statement not supported yet

Implementation

```
[~/build-tools/aci/aci-postgresql-bdr] $ tree
.
├── Jenkinsfile
├── aci-manifest.yml
├── attributes
│   ├── base.yml
│   └── postgresql.yml
├── files
│   └── tmp
│       └── postgresql
│           ├── environment
│           ├── pg_ctl.conf
│           ├── pg_ident.conf
│           └── start.conf
├── runlevels
│   ├── build
│   │   └── 00.install.sh
│   └── build-late
│       └── 00.clean.sh
└── templates
    └── dgr
        └── runlevels
            └── prestart-late
                ├── 00.init-instance.sh.tmpl
                └── 01.init-database.sh.tmpl
```

Run

Check

Check if node have entries in the bdr_nodes table, if yes : skip init

Init

Implementation (init)

1. If no “donor” attributes : Init as new group

When the node have “donor” attributes :

1. Retrieve user definition on donor (`pg_dumpall -g`)
2. Join BDR group
3. Create minimum objects if not present

New fresh node

1. Part local node on donor
2. Delete entries on donor
   (bdr_nodes and bdr_connections)

Node already referenced but changed host or have lost his data

Monitoring and Alerting

Monitoring

Key principles:

- Usage
- Saturation

BDR exporter specifics

Template values for BDR specifics

Extend metrics to all PostgreSQL needs

```
$ cat aci-prometheus-postgresql-exporter/templates/queries.tmpl.yaml

{{ if .use_bdr }}
pg_replication_bdr_count:
  query: "select (select count(*) from bdr.bdr_nodes) as bdr_nodes, (select count(*) from
bdr.bdr_connections) as bdr_connections;"
  metrics:
    - bdr_nodes:
        usage: "GAUGE"
        description: "Number of rows in the bdr_nodes table"
    - bdr_connections:
        usage: "GAUGE"
        description: "Number of rows in the bdr_connections table"
{{ end }}

pg_replication_count:
  query: "select (select count(*) from pg_stat_replication) as stat_repli, (select count(*) from
pg_replication_slots where active=true) as rep_slots;"
  metrics:
    - stat_repli:
        usage: "GAUGE"
        description: "Number of rows in the pg_stat_replication table"
    - rep_slots:
        usage: "GAUGE"
        description: "Number of rows in the pg_replication_slots table with the active status"

[...]
```

Backup and Recovery

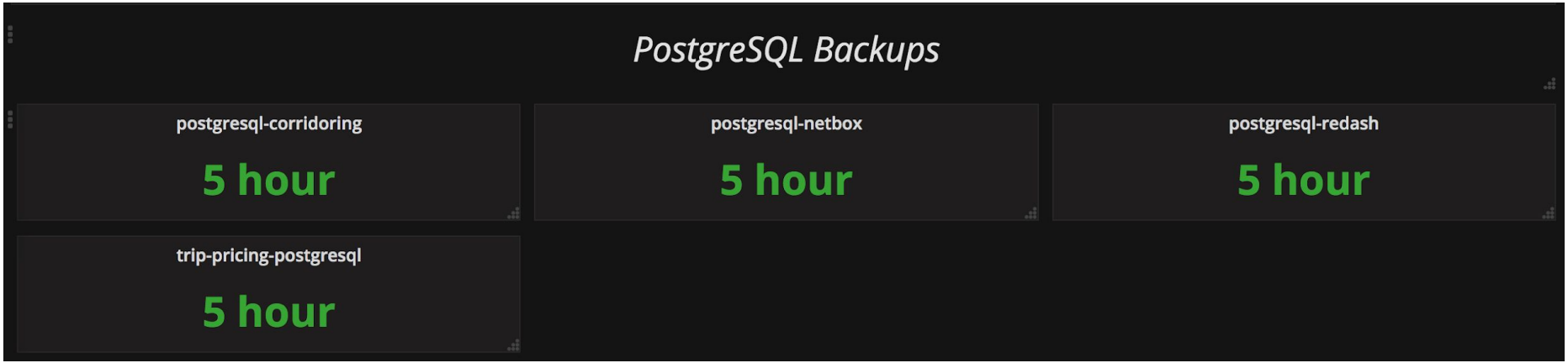
pg_dump

1. Retrieve dumps
2. Alter structure dump
3. Load structure and data dump

Backup and Recovery

```
$ cat pod-mysql-backup/aci-backup/templates/opt/backup-main.tmpl.sh

function startbackup {
  begin_unixtime=$(date +%s)
  cat <<EOF | curl --data-binary @-
http://prometheus-gw:9091/metrics/job/backup_{{.env}}/target/$node/service/$service/type/{{.backup.type}}
  # HELP backup_begin_unixtime
  # TYPE backup_begin_unixtime counter
  backup_begin_unixtime $begin_unixtime
EOF
}
```



Alerting

PromQL to find out unhealthy services

Labeling for routing to Slack & Pager Duty

Annotations with templating to have clear descriptions, URL to dashboards and ops runbooks

```
$ cat prometheus-rules/alert.postgresql.rules

# Alert: There is less replication active than bdr nodes
ALERT BackupsTooOld

IF time() - backup_end_unixtime{exported_service=~".*postgresql.*"} ) > ( 3600 * 24 )


LABELS {
  severity="warning",
  stack="backups",
  team="data_infrastructure"
}


ANNOTATIONS {
   summary="Backup {{ $labels.type }} on {{ $labels.exported_service }}.{{ $labels.target }} is too
old.",
   dashboard="https://grafana.blabla.car/dashboard/db/db-backups ",
}
```

Feedback

- Clearly satisfied with availability
- Reactive community
- Know what your needs are
- Sanity checks
- BDR 3.0 coming soon!

What’s next?

“Thanks!

Questions