Infrastructure at your Service.

PostgreSQL upgrade best practices

dbi services

About me

Daniel Westermann

Senior Consultant

Open Infrastructure Technology Leader

+41 79 927 24 46

daniel.westermann@dbi-services.com

Who we are
dbi services

Experts At Your Service
- Over 50 specialists in IT infrastructure
- Certified, experienced, passionate

Based In Switzerland
- 100% self-financed Swiss company
- Over CHF 8.4 mio. turnover

Leading In Infrastructure Services
- More than 150 customers in CH, D, & F
- Over 50 SLAs dbi FlexService contracted

Certified Management System SQS ISO/IEC 20000-1

Best Workplace in Switzerland 2017 Small Companies 20-49 employees, Rank 7

GREAT PLACE TO WORK® Best Companies to Work for 2017 Switzerland

dbi services is hiring (career@dbi-services.com)

What is this about

Agenda

Introduction

Upgrade preparations

How to upgrade

Demo

Introduction

Introduction
Never touch/change a running system?

Who agrees?

Introduction
Never touch/change a running system?

When you never touch a running a system ...

- Are you sure the instance will come up again when restarted?
- Are you sure you are not affected by security issues?
- Silent data corruptions?
- Can you restore and recover? Really?
- What is the status of your operating system then? Solaris 8? Linux 2.x?
  - You'll definitely have security issues there at least
- Can you still get disks in case you need them?
- Is there anybody who knows the system then?
- Who is able to support that?
- When the system really is not used, then shut it down
- There will be a point in time where you'll have to touch it

Introduction
Never touch a running system?

Things are changing, keep yourself updated

```
psql (8.4.22)
Type "help" for help.

postgres=# create extension hstore;
ERROR:  syntax error at or near "extension"
LINE 1: create extension hstore;

postgres=# alter system set shared_buffers=128M;
ERROR:  syntax error at or near "system"
LINE 1: alter system set shared_buffers=128M;

postgres=# show wal_compression;
ERROR:  unrecognized configuration parameter "wal_compression"
```

You will miss a lot of cool features otherwise

Introduction

When you have something like this …

```
# select version();
|                                    version
+-----------------------------------------------------------------------------------------
| PostgreSQL 8.4.22 on x86_64-unknown-linux-gnu, compiled by GCC gcc (GCC) 4.8.5 20150623
(Red Hat 4.8.5-11), 64-bit
+-----------------------------------------------------------------------------------------
(1 row)
```

… or even this

```
# select version();
|                                    version
+-----------------------------------------------------------------------------------------
| PostgreSQL 9.2.21 on x86_64-unknown-linux-gnu, compiled by GCC gcc (GCC) 4.8.5 20150623
(Red Hat 4.8.5-11), 64-bit
+-----------------------------------------------------------------------------------------
(1 row)
```

Introduction
… then it is time to upgrade

Introduction

Ok, ok, got it … but where to start

Version	Current minor	Supported	Released	EOL
9.6	9.6.3	Yes	SEP-2016	SEP-2021
9.5	9.5.7	Yes	JAN-2016	JAN-2021
9.4	9.4.12	Yes	DEC-2014	DEC-2019
9.3	9.3.17	Yes	SEP-2013	SEP-2018
9.2	9.2.21	Yes	SEP-2012	SEP-2017
9.1	9.1.24	Yes	SEP-2011	SEP-2016
9.0	9.0.23	No	SEP-2010	SEP-2015
8.4	8.4.22	No	JUL-2009	JUL-2014
8.3	8.3.23	No	FEB-2008	FEB-2013
8.2	8.2.23	No	DEC-2006	DEC-2011
8.1	8.1.23	No	NOV-2005	NOV-2010
8.0	8.0.26	No	JAN-2005	OCT-2010
…	…	…	…	…
6.3	6.3.2	No	MAR-1998	MAR-2003

Introduction
Ok, ok, got it … but where to start

Release schedules (well, at least for the minor versions)

- https://www.postgresql.org/developer/roadmap/

PostgreSQL

The world's most advanced open source database.

Home | About | Download | Documentation | Community | Developers | Support | Your account

- Developers
- Core Team
- Roadmap
- Coding
- Testing
  - Beta Information
- Mailing Lists
- Developer FAQ

Roadmap

PostgreSQL is a non-commercial, all volunteer, free software project, and as such there is no formal list of feature requirements required for development. We really do follow the mantra of letting developers scratch their own itches.

Upcoming minor releases

The PostgreSQL project aims to make *at least* one minor release every quarter, on a predefined schedule. If it becomes necessary due to an important bugfix or security issue, more releases will be made between these dates, so this list should be considered a minimum. At each of these dates, a new minor release will be made for each supported version.

The target date for these releases are, unless otherwise stated, the second Thursday of February, May, August, and November. The current schedule for upcoming releases is:

- August 10th, 2017
- November 9th, 2017
- February 8th, 2018

Next major release

The next major release of PostgreSQL is planned to be the 10 release. A tentative schedule for this version has a release in the third quarter of 2017.

Introduction
Where to find security related information

There is a dedicated website for security issues on www.postgresql.org

- https://www.postgresql.org/support/security/

Reference	Affected versions	Fixed in	Component	Class	Description
CVE-2017-7484	9.2-9.6	9.6.3, 9.5.7, 9.4.12, 9.3.17, 9.2.21	core server	C	selectivity estimators bypass SELECT privilege checks
CVE-2017-7485	9.3-9.6	9.6.3, 9.5.7, 9.4.12, 9.3.17	client	A	libpq ignores PGREQUIRESSL environment variable
CVE-2017-7486	9.2-9.6	9.6.3, 9.5.7, 9.4.12, 9.3.17, 9.2.21	core server	C	pg_user_mappings view discloses foreign server passwords
CVE-2016-7048	9.1-9.5	9.5.5, 9.4.10, 9.3.15, 9.2.19, 9.1.24	packaging	A	Interactive installer downloads software over plain HTTP, then executes it

Introduction
You have to, yes, you really, really have to

Read the release notes

- https://www.postgresql.org/docs/current/static/release.html

Home → **Documentation** → **Manuals** → **PostgreSQL 9.6**

This page in other versions: 9.2 / 9.3 / 9.4 / 9.5 / current (9.6) | Development versions: devel / 10 | Unsupported versions: 7.1 / 7.2 / 7.3 / 7.4 / 8.0 / 8.1 / 8.2 / 8.3 / 8.4 / 9.0 / 9.1

Prev Up **PostgreSQL 9.6.3 Documentation**

Appendix E. Release Notes

Table of Contents

Introduction
Release notes

When you do not take your time to do that

- 9.6.3
  - Indexes on columns containing such large values should be reindexed, since they may be corrupt.
- 9.6.2
  - However, if your installation has been affected by the bug described in the first changelog entry below, then after updating you may need to take action to repair corrupted indexes.
- 9.6.1
  - ... then after updating you may need to take action to repair corrupted free space maps and/or visibility maps

Introduction
Release notes

When you do not take your time to do that

- 9.5.6
  - ... then after updating you may need to take action to repair corrupted indexes
- 9.5.5
  - ... then after updating you may need to take action to repair corrupted free space maps
- 9.5.2
  - ... you may need to REINDEX some indexes after applying the update
- 9.5.2
  - In pg_upgrade, skip creating a deletion script when the new data directory is inside the old data directory
  - Blind application of the script in such cases would result in loss of the new data directory

Introduction
Release notes

!!! https://www.postgresql.org/docs/current/static/release.html !!!

Introduction
What are PostgreSQL minor and major versions?

Currently the third digit of the version number defines the minor release

- 9.5.1, 9.5.2, 9.5.3
- 9.4.4, 9.4.3, 9.4.2

Currently the first and second digit of the version number define the major release

- 9.5.1, 9.5.2, 9.5.3
- 9.4.4, 9.4.3, 9.4.2

Introduction
What are PostgreSQL minor and major versions?

This will change starting with PostgreSQL 10

- The first digit defines the major version
  - 10, 11, 12, ...
- The second digit defines the minor version
  - 10.1, 10.2, 10.3, ...

The third digit will be history

Introduction
PostgreSQL 10 will break things

Some changes

- pg_xlog => pg_wal
- pg_switch_xlog() => pg_switch_wal()
- pg_receivexlog => pg_receivewal
- --xlogdir => --waldir
- pg_clog => pg_xact
- pg_log => log
- WAL-related functions and views use lsn instead of location
- pg_dump/pg_dumpall do not anymore support versions prior to PostgreSQL 8.0

Introduction
PostgreSQL 10 will bring cool features

Some PostgreSQL 10 features (probably)

- Quorum commit for synchronous replicas
- Parallel query V2
- Logical replication
- Wait events for latches
- Partitioning syntax
- Client side connection failover
- WAL logged hash indexes
- ...

Introduction
Getting support

When you run into issues or have questions make use of the mailing lists

- https://www.postgresql.org/list/
- Usually the pgsql-general list is the list to start with
  - https://www.postgresql.org/list/pgsql-general
- You will be surprised how fast you get answers

But read this before

- https://wiki.postgresql.org/wiki/Guide_to_reporting_problems
- Especially the section: "**Things not to do**"

Introduction
Getting support

Search, before posting

Search for: upgrade 8.4 [Search]

List: pgsql-general

Post date: within last year

Sort by: Rank

Results 1-20 of 465.

Result pages: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 Next

1. Re: Index seems "lost" after consecutive deletes [0.10]
From Edson Richter <edsonrichter@hotmail.com> on 2016-06-15T19:19:41.
Em 14/06/2016 12:02, Edson Richter escreveu: > Em 14/06/2016 10:32, David G. Johnston escreveu
https://www.postgresql.org/message-id/BLU436-SMTP239604466AA89EF6FE89D2ECF550@phx.gbl

2. Re: Postgres 9.5.2 upgrade to 9.6 [1.30]
From "David G. Johnston" <david.g.johnston@gmail.com> on 2016-06-22T18:42:22.
On Wed, Jun 22, 2016 at 2:36 PM, Michelle Schwan wrote: > I have
https://www.postgresql.org/message-id/CAKFQuwYEC=Q6x=K5JeDAzWRhUaqgH3XQVXwLiiMOeS2PGuXiHA@mail.gmail.com

3. Re: ERROR: missing chunk number 0 for toast value while using logical decoder.\ [0.20]

Introduction
Getting support

When you do not use the PostgreSQL community version, e.g.

- EnterpriseDB
- 2ndQuadrant
- Greenplum
- Citus
- ...
- https://wiki.postgresql.org/wiki/PostgreSQL_derived_databases

Use the support of the vendor, not the PostgreSQL mailing lists

Upgrade preparations

Upgrade preparations
Where does your PostgreSQL installation come from?

How many choices do you have to get PostgreSQL onto your systems?

- Compiled from source code
- Packages provided by your operating system distribution
- apt and yum based PostgreSQL repositories
  - https://wiki.postgresql.org/wiki/Apt
  - https://yum.postgresql.org/
- The installer provided by EnterpriseDB
  - https://www.enterprisedb.com/downloads/postgres-postgresql-downloads#linux

Upgrade preparations
Where does your PostgreSQL installation come from?

What exactly is installed (RedHat based)?

```
$ yum search postgres
postgresql.i686 : PostgreSQL client programs
postgresql.x86_64 : PostgreSQL client programs
postgresql-contrib.x86_64 : Extension modules distributed with PostgreSQL
postgresql-devel.i686 : PostgreSQL development header files and libraries
postgresql-devel.x86_64 : PostgreSQL development header files and libraries
postgresql-docs.x86_64 : Extra documentation for PostgreSQL
postgresql-jdbc.noarch : JDBC driver for PostgreSQL
postgresql-jdbc-javadoc.noarch : API docs for postgresql-jdbc
postgresql-libs.i686 : The shared libraries required for any PostgreSQL clients
postgresql-libs.x86_64 : The shared libraries required for any PostgreSQL clients
postgresql-odbc.x86_64 : PostgreSQL ODBC driver
postgresql-plperl.x86_64 : The Perl procedural language for PostgreSQL
postgresql-plpython.x86_64 : The Python2 procedural language for PostgreSQL
…
```

Upgrade preparations
Where does your PostgreSQL installation come from?

What exactly is installed (Debian based)?

```
$ apt search postgres
postgresql/stable 9.4+165+deb8u2 all
  object-relational SQL database (supported version)
postgresql-client/stable 9.4+165+deb8u2 all
  front-end programs for PostgreSQL (supported version)
postgresql-client-common/stable 165+deb8u2 all
  manager for multiple PostgreSQL client versions
postgresql-common/stable 165+deb8u2 all
  PostgreSQL database-cluster manager
postgresql-doc/stable 9.4+165+deb8u2 all
  documentation for the PostgreSQL database management system
postgresql-plperl-9.1/stable 9.1.22-0+deb8u1 amd64
  PL/Perl procedural language for PostgreSQL 9.1
postgresql-server-dev-all/stable 165+deb8u2 all
  extension build tool for multiple PostgreSQL versions
…
```

Upgrade preparations
Where does your PostgreSQL installation come from?

What exactly is installed (SUSE based)?

```
$ zypper search postgres
  | postgresql-devel      | PostgreSQL development header files and libraries
  | postgresql-init       | Init script and other infrastructure for PostgreSQL
  | postgresql-init       | Init script and other infrastructure for PostgreSQL
  | postgresql-jdbc       | Official JDBC Driver for PostgreSQL
  | postgresql-jdbc       | Official JDBC Driver for PostgreSQL
  | postgresql94          | Basic Clients and Utilities for PostgreSQL
  | postgresql94          | Basic Clients and Utilities for PostgreSQL
  | postgresql94-contrib  | Contributed Extensions and Additions to PostgreSQL
  | postgresql94-devel    | PostgreSQL development header files and libraries
  | postgresql94-docs     | HTML Documentation for PostgreSQL
  | postgresql94-libs     | Basic Clients and Utilities for PostgreSQL
  | postgresql94-server   | The Programs Needed to Create and Run a PostgreSQL Server
```

Upgrade preparations
Where does your PostgreSQL installation come from?

What exactly is installed (FreeBSD)?

```
$ pkg search postgres
postgresql-jdbc-9.2.1004        The Java JDBC implementation for PostgreSQL
postgresql-libpgeasy-3.0.4_1    Easy-to-use C interface to PostgreSQL
postgresql-libpqxx-4.0.1_1      New C++ interface for PostgreSQL
postgresql-libpqxx3-3.1.1_1     New C++ interface for PostgreSQL
postgresql-odbc-09.06.0100      PostgreSQL ODBC driver
postgresql-plproxy-2.7          PL/Proxy - database partitioning system
postgresql-relay-1.3.2_1        Multiplex multiple PostgreSQL databases to one relay
postgresql-repmgr-3.3           PostgreSQL replication manager
postgresql-repmgr2-2.0.3_1      PostgreSQL replication manager
postgresql96-client-9.6.2       PostgreSQL database (client)
postgresql96-contrib-9.6.2      The contrib utilities from the PostgreSQL distribution
postgresql96-docs-9.6.2         The PostgreSQL documentation set
postgresql96-plperl-9.6.2       Write SQL functions for PostgreSQL using Perl5
postgresql96-plpython-9.6.2     Module for using Python to write SQL functions
```

Upgrade preparations
Where does your PostgreSQL installation come from?

Most of the distributions provide separate packages for

- PostgreSQL server
- PostgreSQL clients
- PostgreSQL extensions / contrib
- PostgreSQL development libraries
- PostgreSQL documentation
- ...

Make sure you install the same set of packages for your target release

Upgrade preparations
Where does your PostgreSQL installation come from?

When you installed from source

```
postgres@pgday1:/home/postgres/ [I9221] pg_config
BINDIR = /u01/app/postgres/product/92/db_21/bin
DOCDIR = /u01/app/postgres/product/92/db_21/share/doc
HTMLDIR = /u01/app/postgres/product/92/db_21/share/doc
INCLUDEDIR = /u01/app/postgres/product/92/db_21/include
PKGINCLUDEDIR = /u01/app/postgres/product/92/db_21/include
INCLUDEDIR-SERVER = /u01/app/postgres/product/92/db_21/include/server
LIBDIR = /u01/app/postgres/product/92/db_21/lib
PKGLIBDIR = /u01/app/postgres/product/92/db_21/lib
LOCALEDIR = /u01/app/postgres/product/92/db_21/share/locale
MANDIR = /u01/app/postgres/product/92/db_21/share/man
SHAREDIR = /u01/app/postgres/product/92/db_21/share
SYSCONFDIR = /u01/app/postgres/product/92/db_21/etc
PGXS = /u01/app/postgres/product/92/db_21/lib/pgxs/src/makefiles/pgxs.mk
…
```

Upgrade preparations
Where does your PostgreSQL installation come from?

When you installed from source - continued

```
postgres@pgday1:/home/postgres/ [I9221] pg_config

CONFIGURE = '--prefix=/u01/app/postgres/product/92/db_21' '--exec-
prefix=/u01/app/postgres/product/92/db_21' '--
bindir=/u01/app/postgres/product/92/db_21/bin' '--
libdir=/u01/app/postgres/product/92/db_21/lib' '--
sysconfdir=/u01/app/postgres/product/92/db_21/etc' '--
includedir=/u01/app/postgres/product/92/db_21/include' '--
datarootdir=/u01/app/postgres/product/92/db_21/share' '--
datadir=/u01/app/postgres/product/92/db_21/share' '--with-pgport=5432' '--with-perl' '--
with-python' '--with-openssl' '--with-pam' '--with-ldap' '--with-libxml' '--with-libxslt'
'--with-segsize=2' '--with-blocksize=8' '--with-wal-segsize=64'
```

Upgrade preparations
Where does your PostgreSQL installation come from?

When you installed from source - continued

```
postgres@pgday1:/home/postgres/ [I9221] pg_config
CC = gcc
CPPFLAGS = -D_GNU_SOURCE -I/usr/include/libxml2
CFLAGS = -O2 -Wall -Wmissing-prototypes -Wpointer-arith -Wdeclaration-after-statement -
Wendif-labels -Wmissing-format-attribute -Wformat-security -fno-strict-aliasing -fwrapv -
fexcess-precision=standard
CFLAGS_SL = -fpic
LDFLAGS = -Wl,--as-needed -Wl,-rpath,'/u01/app/postgres/product/92/db_21/lib',--enable-
new-dtags
LDFLAGS_EX =
LDFLAGS_SL =
LIBS = -lpgport -lxslt -lxml2 -lpam -lssl -lcrypto -lz -lreadline -lcrypt -ldl -lm
VERSION = PostgreSQL 9.2.21
```

Upgrade preparations
Where does your PostgreSQL installation come from?

When you installed from source - continued

- Make sure you configure/compile your target version with the same settings as the source

```
PGHOME=/u01/app/postgres/product/95/db_1/
SEGSIZE=2
BLOCKSIZE=8
WALSEGSIZE=64
./configure --prefix=${PGHOME} \
            --with-perl \
            --with-python \
            --with-openssl \
            --with-pam \
            --with-ldap \
            --with-libxml \
            --with-segsize=${SEGSIZE} \
            --with-blocksize=${BLOCKSIZE} \
            --with-wal-segsize=${WALSEGSIZE}
```

Upgrade preparations
Where does your PostgreSQL installation come from?

When you don't use the same options you will run into issues like this

```
2017-05-15 15:01:04.527 CEST - 2 - 21860 -  - @ DETAIL:  The database cluster was
initialized with RELSEG_SIZE 131072, but the server was compiled with RELSEG_SIZE 262144.
2017-05-15 15:01:04.527 CEST - 3 - 21860 -  - @ HINT:  It looks like you need to
recompile or initdb.
```

Upgrade preparations
Do you use any extensions?

Which extensions are used on the source?

```
postgres=# \dx
                                    List of installed extensions
  Name   | Version |                              Description
---------+---------+------------+-------------------------------------------------------
 hstore  | 1.1     | data type for storing sets of (key, value) pairs
 pg_trgm | 1.0     | text similarity measurement and index searching based on trigrams
 plperl  | 1.0     | pg_catalog | PL/Perl procedural language
 plpgsql | 1.0     | pg_catalog | PL/pgSQL procedural language
(4 rows)
```

When you have any non-default extensions you'll need to install them on the target before upgrading (e.g. cstore_fdw)

Upgrade preparations
Do you use custom statistic targets?

Did you set any custom statistics targets on the source?

```
with tabs as
( select tablename
    from pg_tables
   where schemaname not in ('information_schema','pg_catalog')
)
select attrelid::regclass, attname, attstattarget
  from pg_attribute a
     , tabs b
 where attrelid::regclass::varchar = b.tablename
   and attstattarget > 0
 order by 1,2,3;
```

attrelid	attname	attstattarget
pgbench_accounts	abalance	**1234**

Upgrade preparations
Do you use custom statistic targets?

Statistics are not transferred to the target, no matter which method you use for upgrading (they are stored in the catalog)

- Generate a script that sets the statistics target for you

```
with tabs as
( select tablename
       , schemaname
    from pg_tables
   where schemaname not in ('information_schema','pg_catalog')
)
select 'alter table '||b.schemaname||'.'||b.tablename||' alter column '||a.attname||' set
statistics '||a.attstattarget||';'
  from pg_attribute a
     , tabs b
 where attrelid::regclass::varchar = b.tablename
   and attstattarget > 0;
```

Upgrade preparations
Do you use custom statistic targets?

Statistics are not transferred to the target, no matter which method you use for upgrading (they are stored in the catalog)

- Generate a script that sets the statistics target for you

```
┌──────────────────────────────────────────────────────────────────────────────┐
│                                   ?column?                                   │
├──────────────────────────────────────────────────────────────────────────────┤
| alter table public.pgbench_accounts alter column bid set statistics 2345;     |
| alter table public.pgbench_accounts alter column filler set statistics 3456;  |
| alter table public.pgbench_history alter column aid set statistics 4567;      |
| alter table public.pgbench_history alter column delta set statistics 5678;    |
| alter table public.pgbench_accounts alter column abalance set statistics 1234; |
└──────────────────────────────────────────────────────────────────────────────┘
```

Upgrade preparations
You do use version specific directories, do you?

When you install PostgreSQL make sure that you install into a version specific directory, e.g.

```
$ ls -la /opt/postgres/
total 0
drwxr-xr-x. 8 postgres postgres 78 Jun  2 16:02 .
drwxr-xr-x. 3 root     root     21 Jun  2 16:01 ..
drwx------. 2 postgres postgres  6 Jun  2 16:02 9.5.5
drwx------. 2 postgres postgres  6 Jun  2 16:02 9.5.6
drwx------. 2 postgres postgres  6 Jun  2 16:02 9.5.7
drwx------. 2 postgres postgres  6 Jun  2 16:02 9.6.1
drwx------. 2 postgres postgres  6 Jun  2 16:02 9.6.2
drwx------. 2 postgres postgres  6 Jun  2 16:02 9.6.3
```

This way you will always have the old binaries available

Upgrade preparations

You do use version specific directories, do you?

When you initdb your cluster, make $PGDATA version specific as well. e.g. (more on the reasons later)

```
$ tree
.
├── 9.5.5
├── 9.5.6
├── 9.5.7
├── 9.6.1
├── 9.6.2
├── 9.6.3
└── data
    ├── 9.5
    |   └── MY_INST1
    ├── 9.5
    |   └── MY_INST2
    └── 9.6
        └── MY_INST1
```

Upgrade preparations
You do use version specific directories, do you?

When you are using tablespaces avoid version specific locations

```
postgres@pgday1:/home/postgres/ [pg9221] ls -la /u90/pgdata/PG1/9.2/tablespaces/
total 0
drwx------. 4 postgres postgres 52 Jun 29 13:32 .
drwxr-xr-x. 3 postgres postgres 24 Jun 28 07:27 ..
drwx------. 4 postgres postgres 34 Jun 28 07:31 PG_9.2_201204301
drwx------. 3 postgres postgres 18 Jun 29 13:32 PG_9.6_201608131
```

The version is in the directory name anyway by default

Upgrade preparations

Create a test instance where you can test your upgrade

- Exactly the same operating system
- Exactly the same PostgreSQL version
  - When you are on PostgreSQL 9.1+
    - pg_basebackup (--xlog)
  - Below 9.1
    - pg_dump / pg_dumpall
- Check all parameters
  - Some maybe changed?
  - Some are new?

How to upgrade

How to upgrade
Minor version upgrades

For minor version upgrades the procedure is simple

- Install the new binaries into a new location
- Shutdown the instance
- Switch the environment to the new instance
- Start the instance with the new binaries
- Done

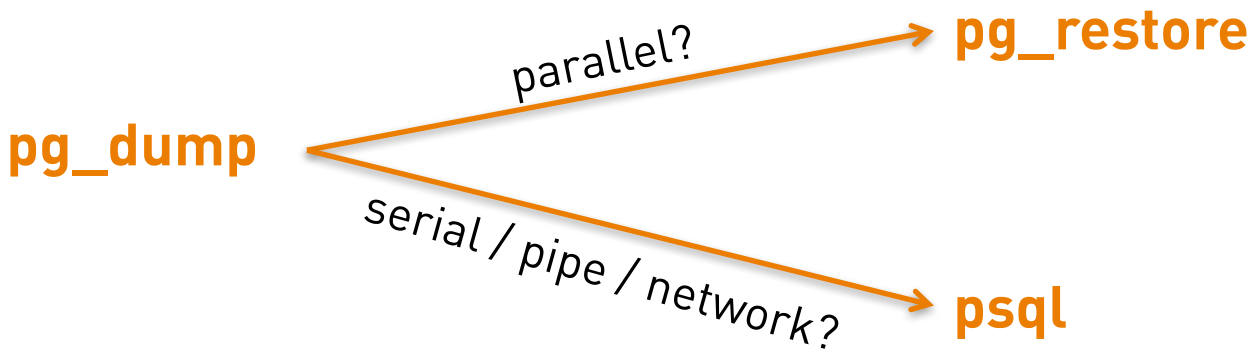
You did read the release notes before, didn't you?

How to upgrade
Major version upgrades

For major version upgrades you have more options

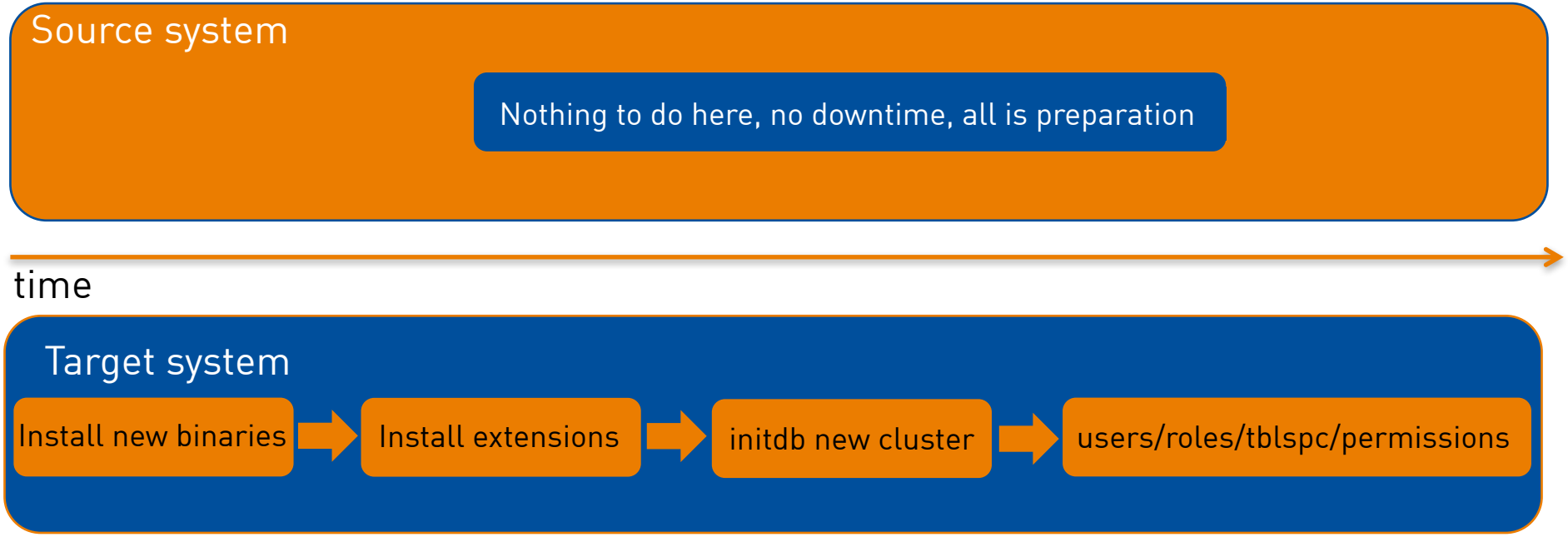
- Install the new binaries into a new location
  - pg_dump
  - pg_dumpall
  - pg_dumpall & pg_dump
  - pg_upgrade
  - (Starting with PostgreSQL 10: Logical replication)

How to upgrade
Major version upgrades



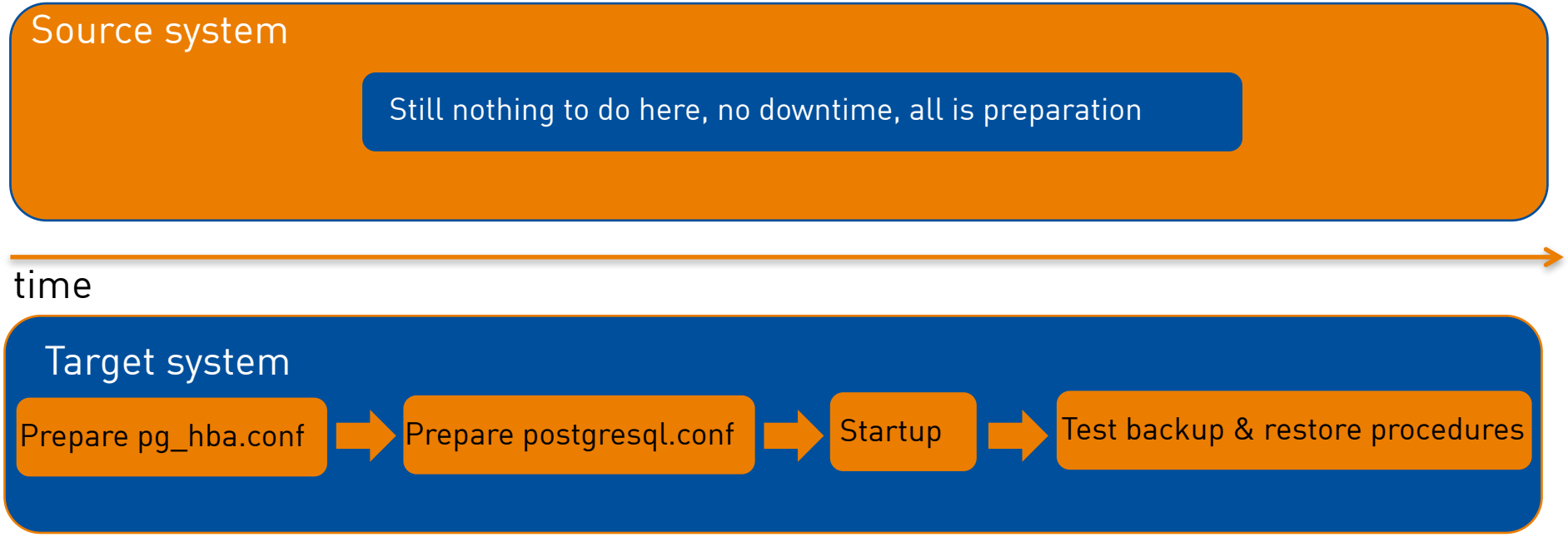
How to upgrade
Major version upgrades – pg_dump/pg_restore

How to start, where to start and what is next?



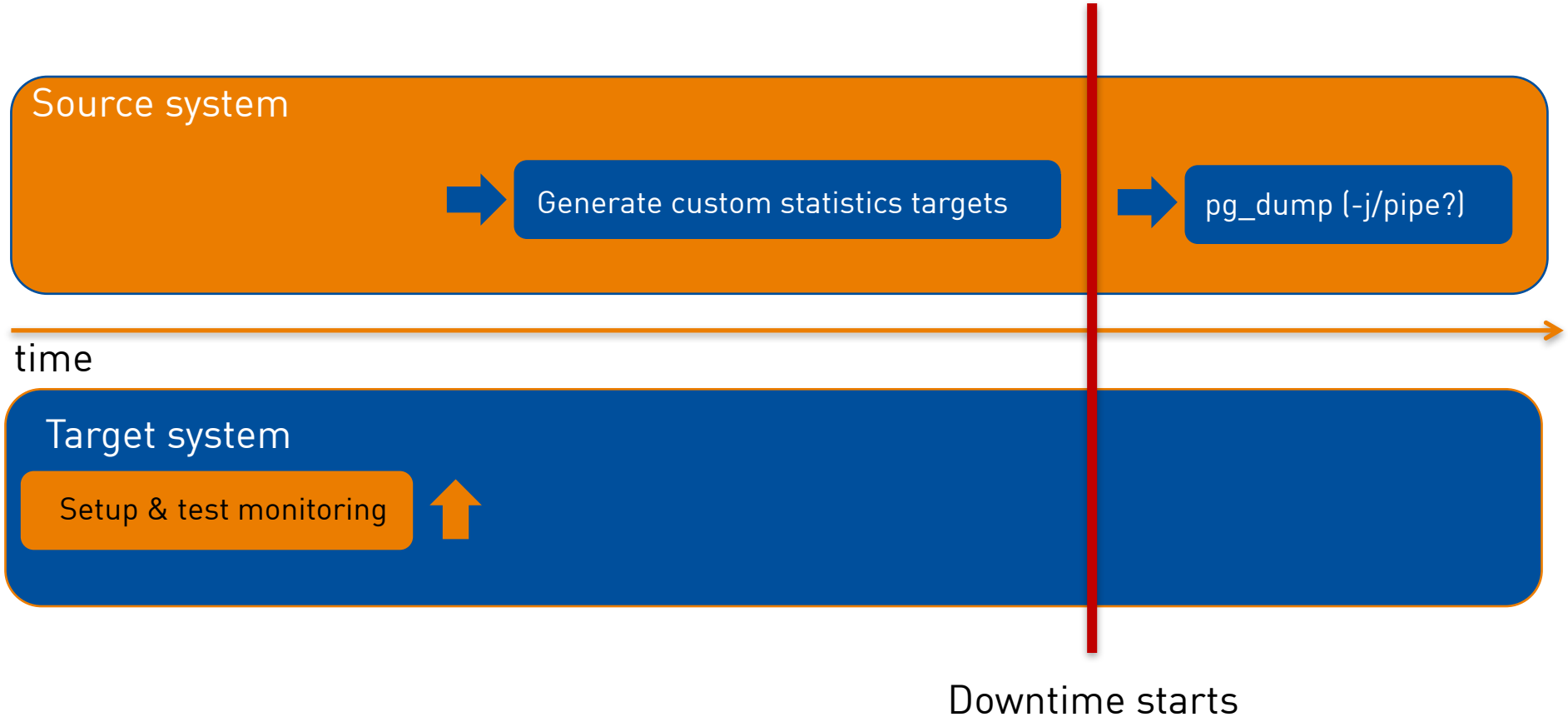
How to upgrade
Major version upgrades – pg_dump/pg_restore

How to start, where to start and what is next?



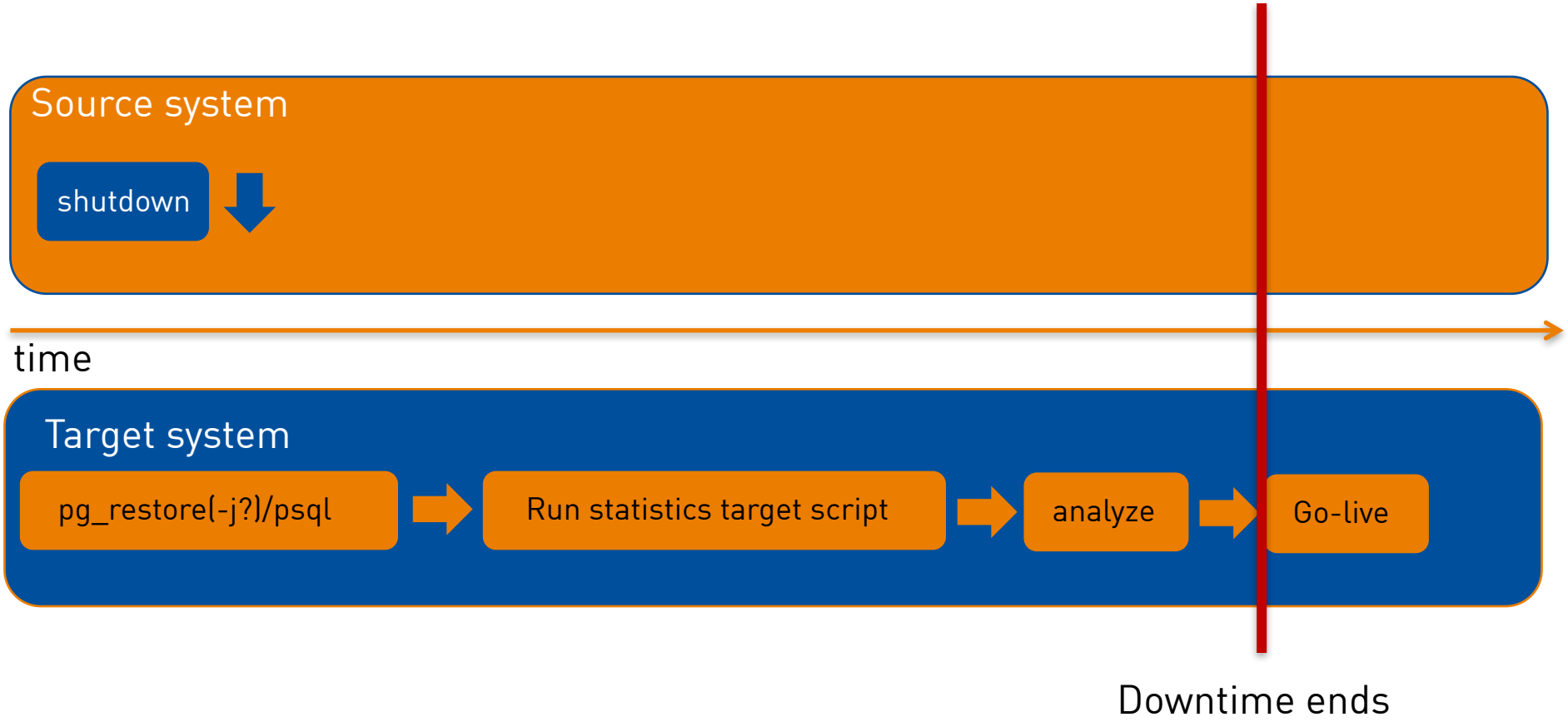
How to upgrade
Major version upgrades – pg_dump/pg_restore

How to start, where to start and what is next?



How to upgrade
Major version upgrades – pg_dump/pg_restore

How to start, where to start and what is next?



How to upgrade

Major version upgrades – pg_dump/pg_restore

You tested all of that before, didn't you?

You did read the release notes, didn't you?

You will closely monitor your new instance for the next hours, won't you

How to upgrade
Major version upgrades – pg_dump/pg_restore

Why did you forgot your replicas then?

- Either prepare the replica the same way as you prepared the master
  - Setup streaming replication before you restore
  - Restore and let the replica catch up, but take care of
    - min_wal_size => PostgreSQL 9.5
    - max_wal_size >= PostgreSQL 9.5
    - wal_keep_segments <= PostgreSQL 9.5
    - or use physical replication slots
- or rebuild the replica when the master is fine

How to upgrade
Major version upgrades – pg_dump/pg_restore

pg_dump --help

- Yes, review the parameters
- Since PostgreSQL 9.3 you can dump and restore in parallel

```
$ pg_dump --help | grep "\-j"
  -j, --jobs=NUM               use this many parallel jobs to dump
```

  - Does not work intra-table
  - When you only have one large table it might not help you much
  - You need to use the directory output format (-F d)
  - What is the value of your max_connections parameter?
  - Can not be used when you want to pipe to psql

How to upgrade
Major version upgrades – pg_dump/pg_restore

pg_dump --help

- Yes, review the parameters
- Only dump the schema and restore it to the target

```
$ pg_dump --help | grep "\-\-schema-only"
 -s, --schema-only            dump only the schema, no data
```

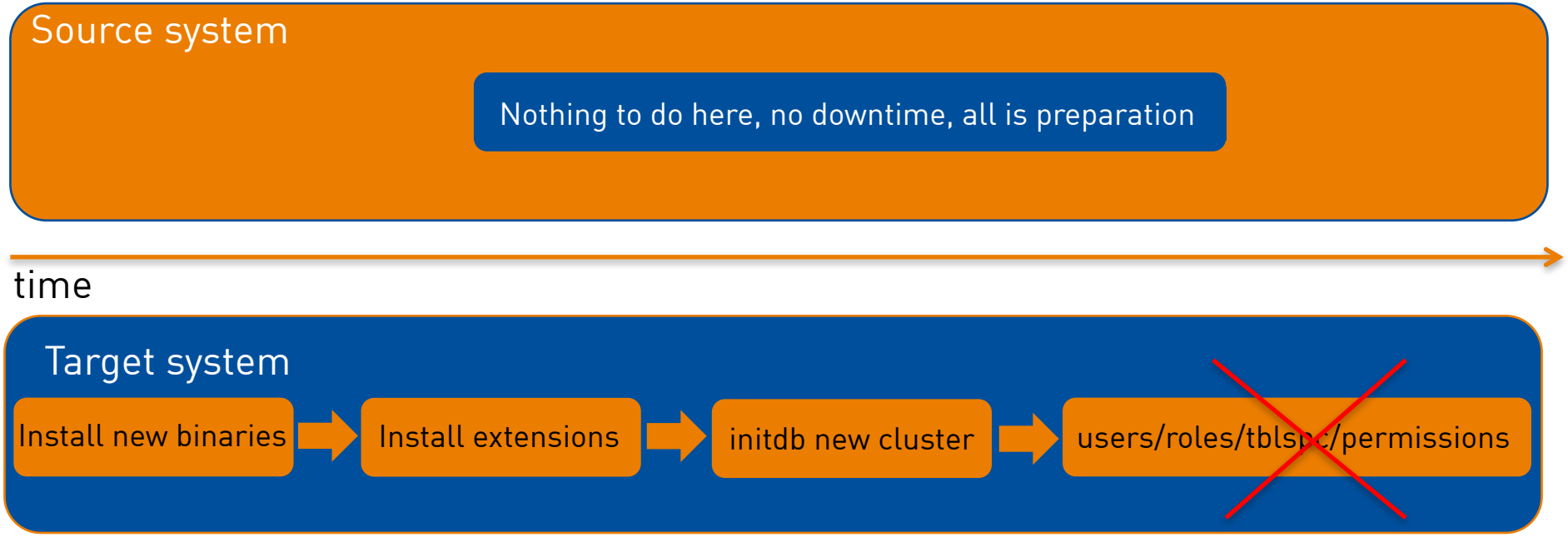
- Then dump and restore the data only

```
$ pg_dump --help | grep "\-\-data-only"
 -a, --data-only              dump only the data, not the schema
```

How to upgrade
Major version upgrades – pg_dumpall

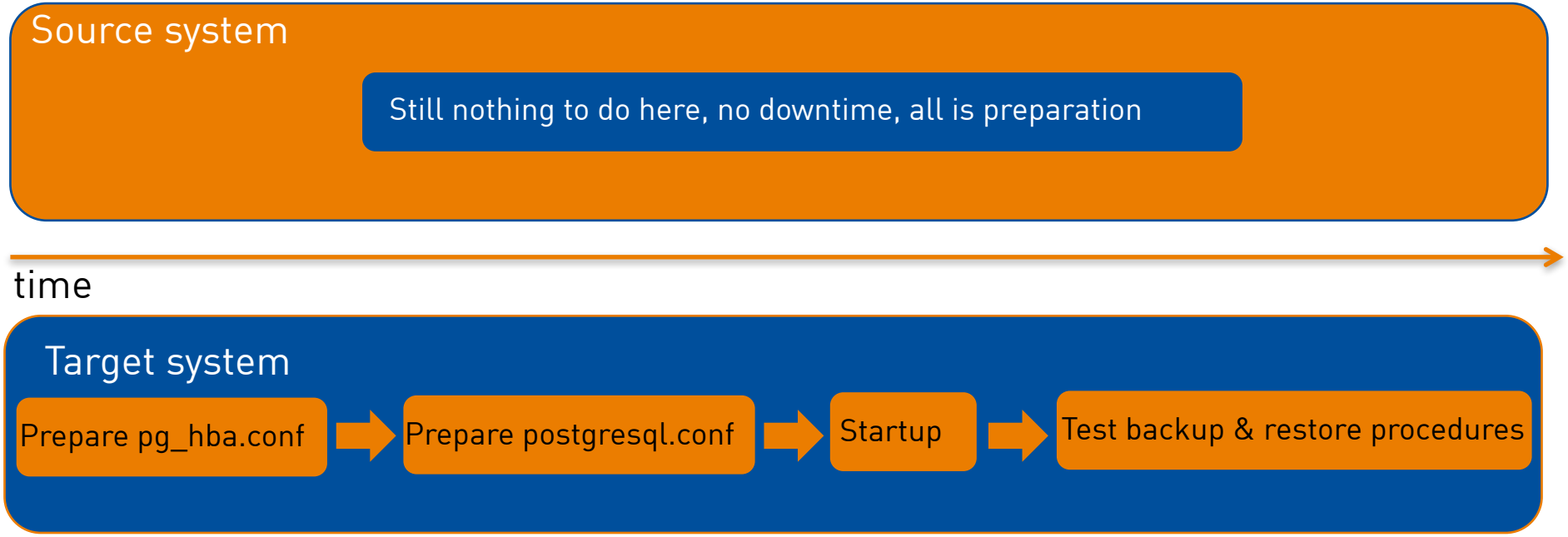
How to upgrade
Major version upgrades – pg_dumpall

How to start, where to start and what is next?



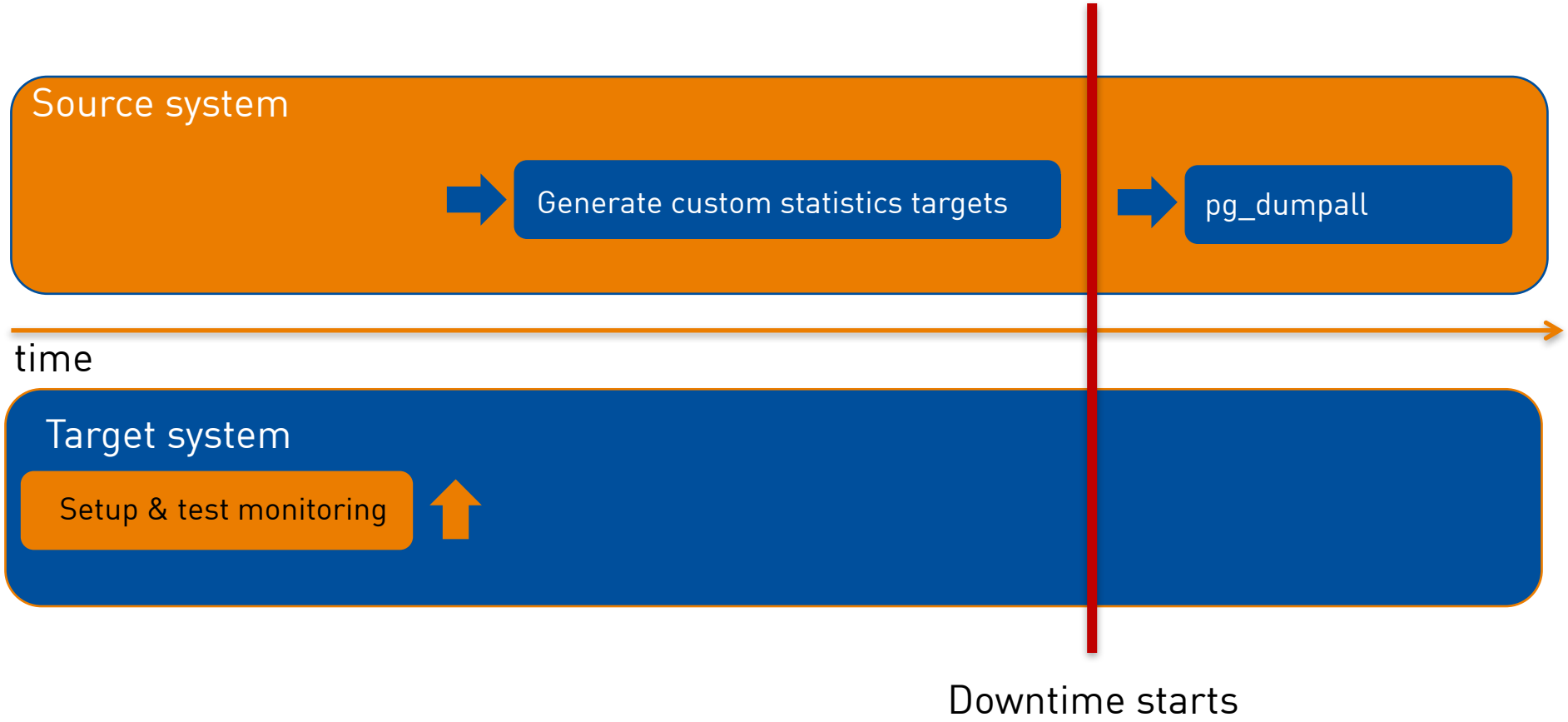
How to upgrade
Major version upgrades – pg_dumpall

How to start, where to start and what is next?



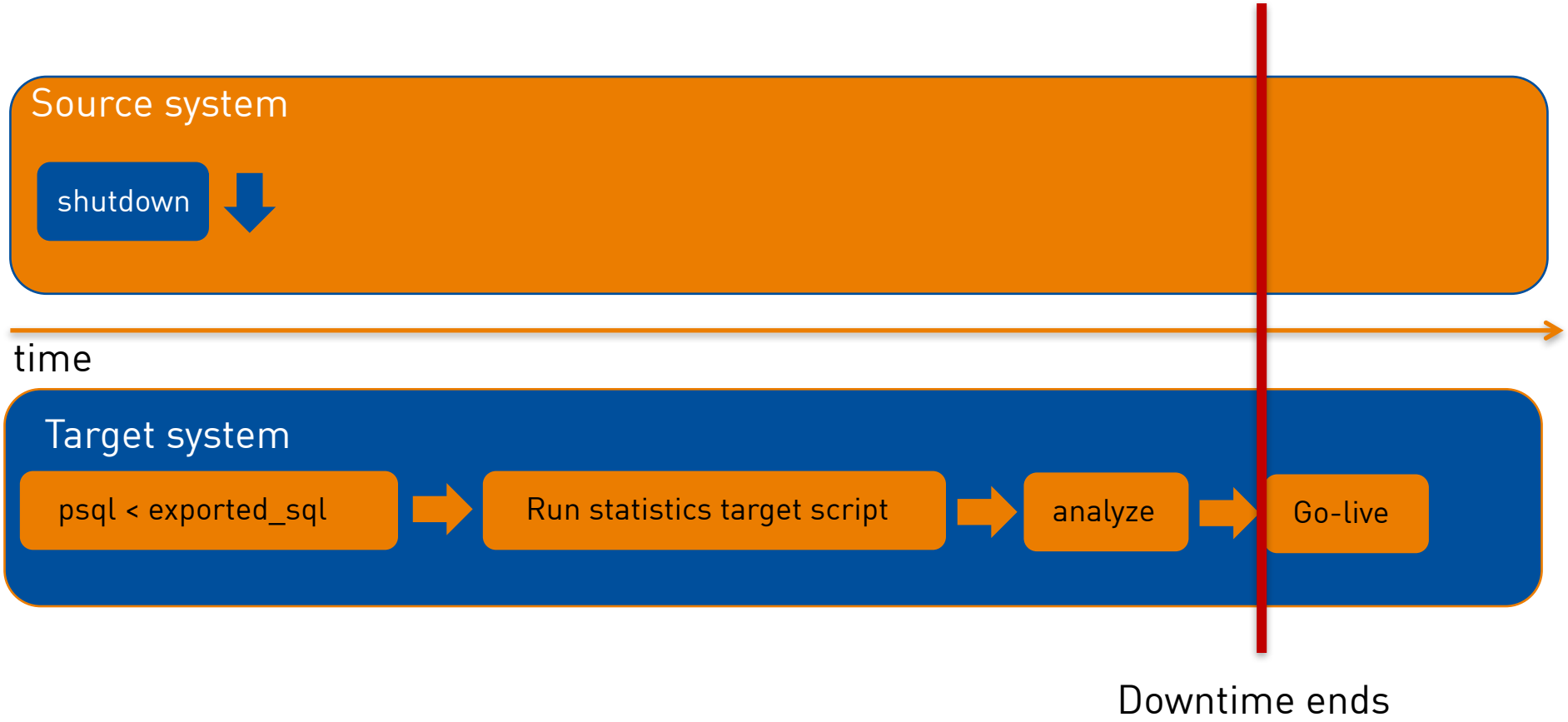
How to upgrade
Major version upgrades – pg_dumpall

How to start, where to start and what is next?



How to upgrade
Major version upgrades – pg_dumpall

How to start, where to start and what is next?



How to upgrade

Major version upgrades – pg_dumpall

You tested all of that before, didn't you?

You did read the release notes, didn't you?

You will closely monitor your new instance for the next hours, won't you

How to upgrade
Major version upgrades – pg_dumpall

pg_dumpall --help

- Yes, review the parameters
- Only dump the schema(s) and restore it/them to the target

```
$ pg_dumpall --help | grep "\-\-schema-only"
 -s, --schema-only            dump only the schema, no data
```

- Then dump and restore the data only

```
$ pg_dump --help | grep "\-\-data-only"
 -a, --data-only              dump only the data, not the schema
```

How to upgrade
Major version upgrades – pg_dumpall

pg_dumpall --help

- Yes, review the parameters
- Dump only the global objects and restore to the target

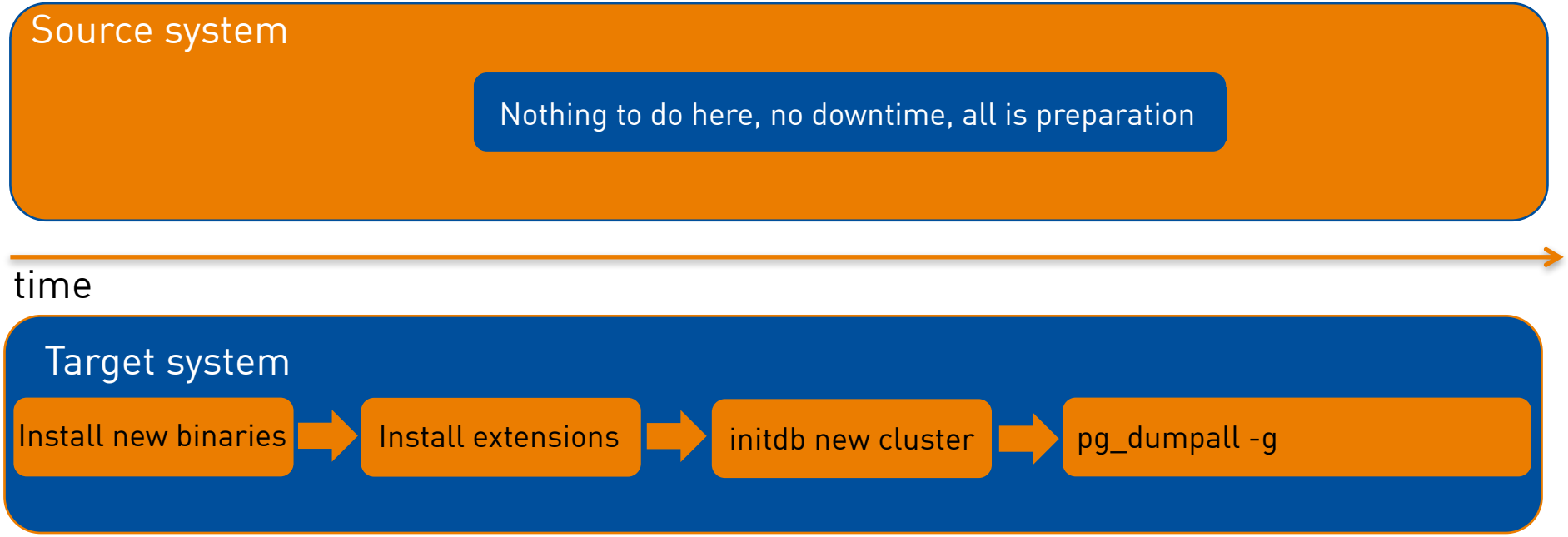
```
$ pg_dumpall --help | grep global
  -g, --globals-only           dump only global objects, no databases
```

  - Users / Roles
  - Global permissions
  - Tablespaces

- When you have this you can use pg_dump / pg_restore in parallel (-j)

How to upgrade
Major version upgrades – pg_dumpall

How to start, where to start and what is next?



How to upgrade
Major version upgrades – pg_upgrade

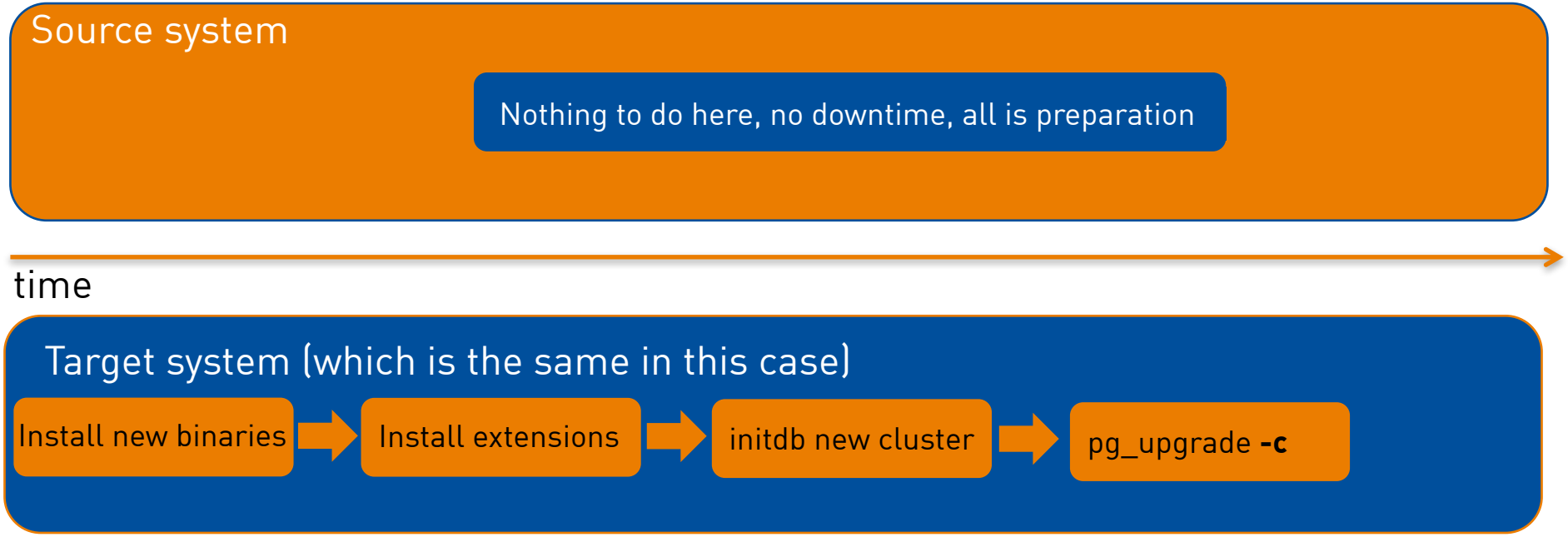
pg_upgrade

There is only one disadvantage, which is?

Source and target must be on the same server!

How to upgrade
Major version upgrades – pg_upgrade

How to start, where to start and what is next?



How to upgrade
Major version upgrades – pg_upgrade

Always run pg_upgrade in check mode first

```
$ export PGDATAOLD=/u02/pgdata/PG1/9.2/
$ export PGDATANEW=/u02/pgdata/PG1/9.6/
$ export PGBINOLD=/u01/app/postgres/product/92/db_21/bin/
$ export PGBINNEW=/u01/app/postgres/product/96/db_3/bin/
$ $PGBINNEW/pg_upgrade -c
```

- This will not touch your old cluster
- Runs compatibility checks and will tell you when something is wrong

How to upgrade
Major version upgrades – pg_upgrade

Always run pg_upgrade in check mode first

```
postgres@pgday1:/home/postgres/ [PG1] $PGBINNEW/pg_upgrade -c
*failure*
Consult the last few lines of "pg_upgrade_server.log" for
the probable cause of the failure.
Performing Consistency Checks on Old Live Server
------------------------------------------------
Checking cluster versions                                   ok
Checking database user is the install user                  ok
Checking database connection settings                       ok
…
Checking for roles starting with 'pg_'                      ok
Checking for invalid "line" user columns                    ok
Checking for presence of required libraries                 ok
Checking database user is the install user                  ok
Checking for prepared transactions                          ok
*Clusters are compatible*
```

How to upgrade
Major version upgrades – pg_upgrade

Always run pg_upgrade in check mode first

```
$ ls -la *upgrade*.log
-rw-------. 1 postgres postgres 1962 Jun 29 09:18 pg_upgrade_internal.log
-rw-------. 1 postgres postgres  358 Jun 29 09:17 pg_upgrade_restore.log
-rw-------. 1 postgres postgres 2076 Jun 29 09:18 pg_upgrade_server.log
-rw-------. 1 postgres postgres  537 Jun 29 09:18 pg_upgrade_utility.log
```

- pg_upgrade will try to start your old cluster
- pg_upgrade will try to start your new cluster

```
$ cat pg_upgrade_server.log
…
command: "/u01/app/postgres/product/92/db_21/bin/pg_ctl" -w -l "pg_upgrade_server.log" -D
"/u02/pgdata/PG1/9.2/" -o "-p 50432 -c autovacuum=off -c
autovacuum_freeze_max_age=2000000000  -c listen_addresses='' -c
unix_socket_permissions=0700" start >> "pg_upgrade_server.log" 2>&1
…
```

How to upgrade
Major version upgrades – pg_upgrade

When you old cluster is down you will not get the *failure*

> ... but do really want to shutdown in the preparation phase?

```
$ pg_ctl -D /u02/pgdata/PG1/9.2/ stop -m fast
$ PGBINNEW/pg_upgrade -c
Performing Consistency Checks
-----------------------------
Checking cluster versions                                   ok
Checking database user is the install user                  ok
Checking database connection settings                       ok
Checking for prepared transactions                          ok
…
Checking for invalid "line" user columns                    ok
Checking for presence of required libraries                 ok
Checking database user is the install user                  ok
Checking for prepared transactions                          ok
*Clusters are compatible*
```

How to upgrade
Major version upgrades – pg_upgrade

How to start, where to start and what is next?

Source system

Still nothing to do here, no downtime, all is preparation

time

Target system (which is the same in this case)

Prepare pg_hba.conf → Prepare postgresql.conf → Startup → Test backup & restore procedures

How to upgrade
Major version upgrades – pg_upgrade

How to start, where to start and what is next?

Source system

Shutdown

time

Target system (which is the same in this case)

pg_upgrade

Copy mode?

Link mode?

Downtime starts

How to upgrade
Major version upgrades – pg_upgrade

pg_upgrade can operate in two modes

- When you go with the defaults your whole cluster will be **copied**
- Remember the version specific $PGDATA recommendation?
- When you have this

```
$ echo $PGDATA
/var/lib/postgres
```

- Where do you want to get the new cluster created?
- Better include your PostgreSQL major version

```
$ echo $PGDATA
/var/lib/postgres/9.2
```

- In copy mode the downtime is dependent on the size of your cluster

How to upgrade
Major version upgrades – pg_upgrade

pg_upgrade can operate in two modes

- You can use the link mode
- This will create hard links in the new cluster which point to the same files as the old cluster

```
$ $PGBINNEW/pg_upgrade --help | grep link
  -k, --link                    link instead of copying files to new cluster
```

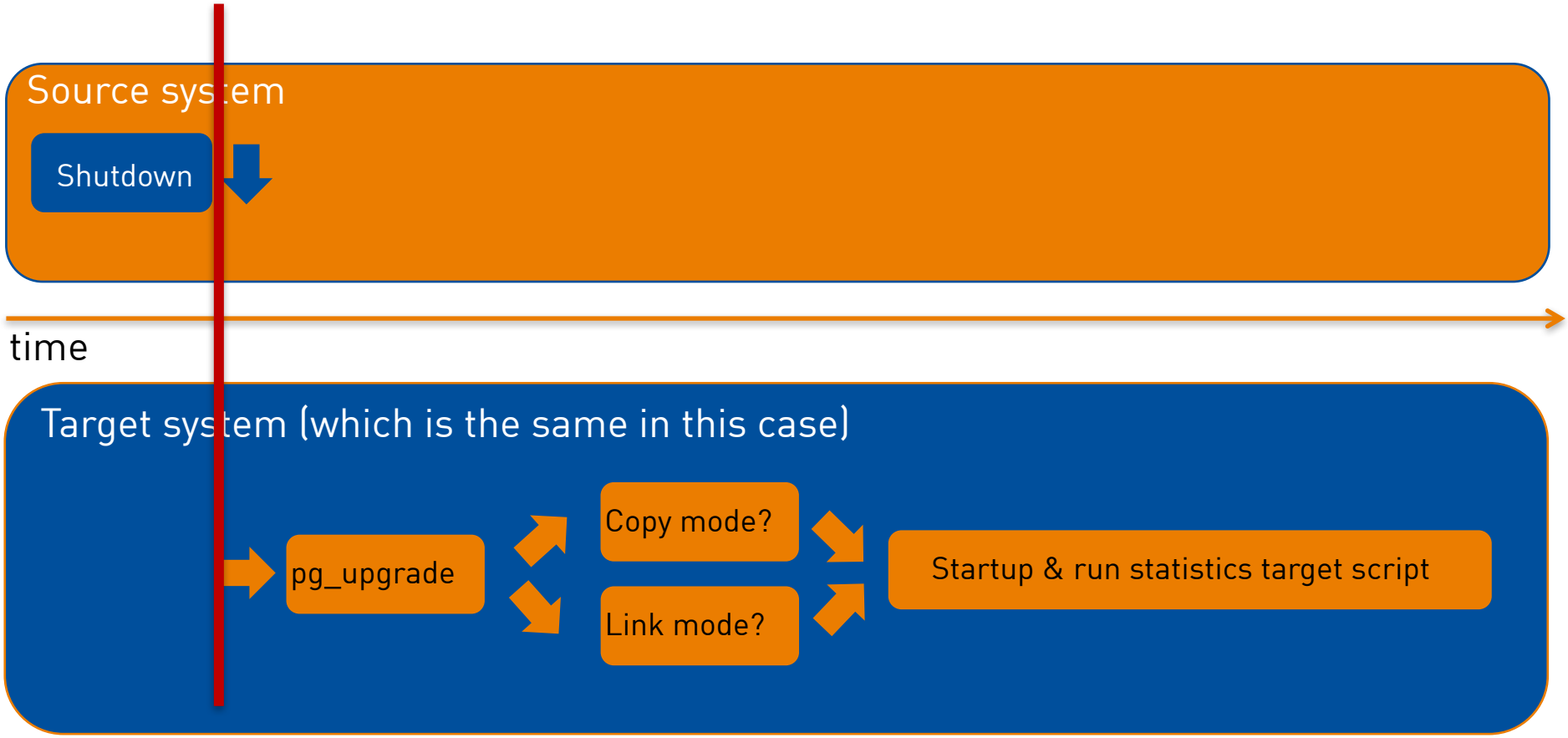
- This is very fast and almost independent of the size of your cluster
- **But:** When you go for link mode you can not switch back to the old cluster as soon as you started the new cluster **!!!**
- Can be used to quickly upgrade a replica (rsync of the hard links)

```
$ rsync --archive --delete --hard-links --size-only data data95 [HOST2]:/u01/pg/
```

- You also need to rsync all your tablespaces and maybe pg_xlog

How to upgrade
Major version upgrades – pg_upgrade

How to start, where to start and what is next?



Downtime starts

How to upgrade
Major version upgrades – pg_upgrade

How to start, where to start and what is next?

Source system

time

Target system (which is the same in this case)

./analyze_new_cluster.sh → ./delete_old_cluster.sh ???

How to upgrade
Major version upgrades – pg_upgrade

./analyze_new_cluster.sh

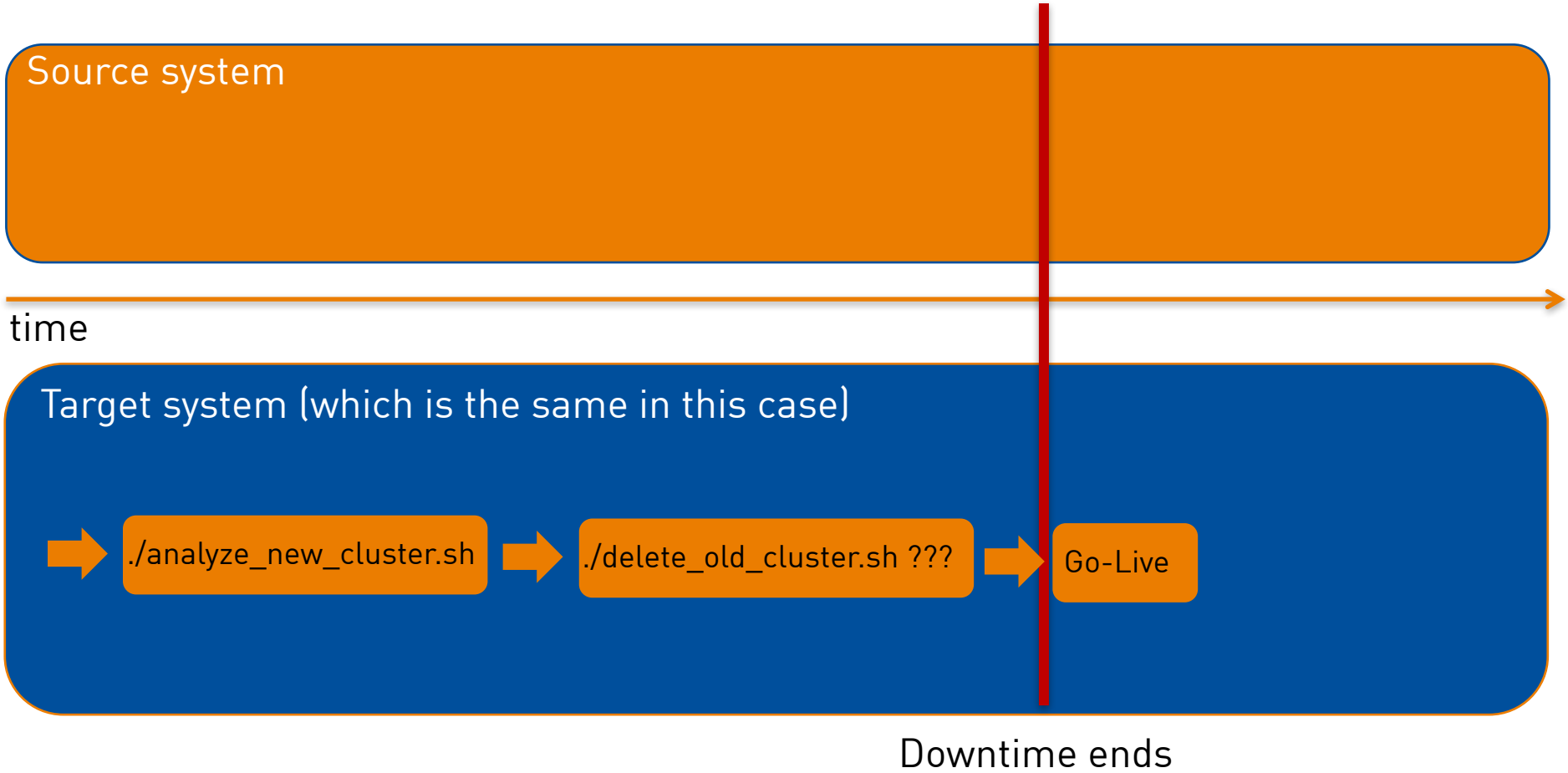
```
This script will generate minimal optimizer statistics rapidly so your system is usable,
and then gather statistics twice more with increasing accuracy.  When it is done, your
system will have the default level of optimizer statistics. If you have used ALTER TABLE
to modify the statistics target for any tables, you might want to remove them and restore
them after running this script because they will delay fast statistics generation.

If you would like default statistics as quickly as possible, cancel

this script and run:

    "/u00/app/pg/product/9.5/bin/vacuumdb" --all --analyze-only
```

./delete_old_cluster.sh – be careful with this one

```
rm -rf $OLDPGDATA
```

How to upgrade
Major version upgrades – pg_upgrade

How to start, where to start and what is next?



How to upgrade
Major version upgrades – pg_upgrade

pg_upgrade --help

- Yes, review the parameters
- You can copy/link in parallel as well

```
$ $PGBINNEW/pg_upgrade --help | grep "\-j"
 -j, --jobs                    number of simultaneous processes or threads to use§
```

- Retaining the SQL and Log files even after a successful upgrade makes sense

```
$ $PGBINNEW/pg_upgrade --help | grep "retain"
 -r, --retain                  retain SQL and log files after success
```

  - This proves success and can be added to the documentation

You did document what you did, didn't you?

How to upgrade

Major version upgrades – pg_upgrade

You tested all of that before, didn't you?

You did read the release notes, didn't you?

You will closely monitor your new instance for the next hours, won't you

How to upgrade
Major version upgrades – Extensions

No matter which method you used, check your extensions after the upgrade

```
# select * from pg_available_extensions;

┌────────────────────┬────────────────┬────────────────────
│        name        │ default_version │ installed_version
├────────────────────┼────────────────┼────────────────────
│ plpgsql            │ 1.0            │ 1.0
│ plperl             │ 1.0            │ 1.0
│ plperlu            │ 1.0            │ NULL
│ plpython2u         │ 1.0            │ NULL
│ plpythonu          │ 1.0            │ NULL
│ earthdistance      │ 1.1            │ NULL
│ file_fdw           │ 1.0            │ NULL
│ fuzzystrmatch      │ 1.1            │ NULL
│ hstore             │ 1.4            │ 1.1
```

How to upgrade
Major version upgrades – Extensions

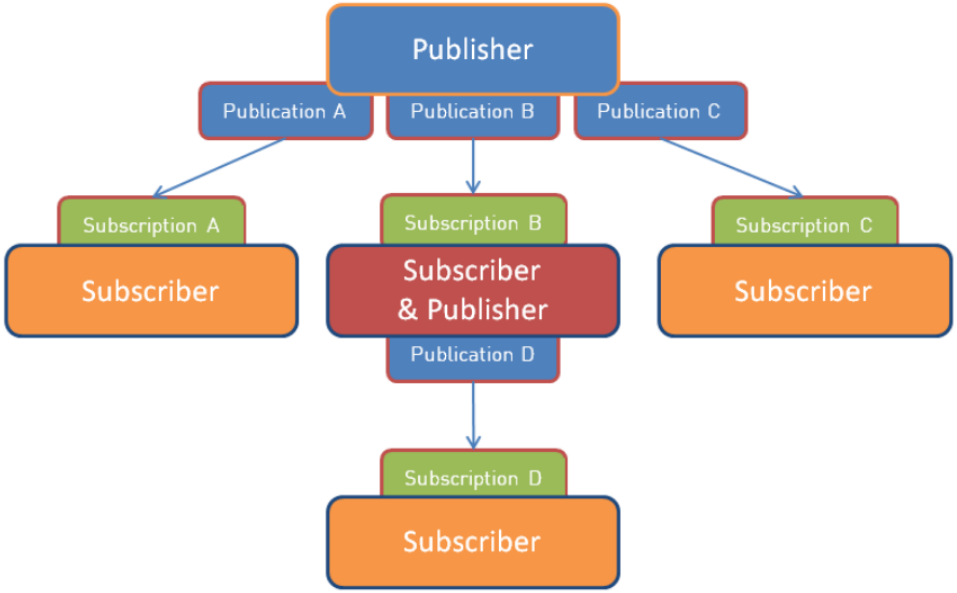
Extensions may need an update as well

```
# alter extension hstore update;
ALTER EXTENSION
# select * from pg_available_extensions where name = 'hstore';
┌────────┬────────────────┬───────────────────
│  name  │ default_version │ installed_version
├────────┼────────────────┼───────────────────
│ hstore │ 1.4            │ 1.4
└────────┴────────────────┴───────────────────
(1 row)
```

How to upgrade
Major version upgrades – logical replication

Starting with PostgreSQL 10 there (probably) will be build in logical replication

- Can be used to offload to reporting instances
- Can be used to consolidate data into another instance
- Can also be used for near zero downtime upgrades



How to upgrade
Major version upgrades – logical replication

On the source you need to create a publication

```
postgres=# create publication my_first_publication for all tables;
CREATE PUBLICATION
```

On the target you create the subscription

```
postgres=# create subscription my_first_subscription connection 'host=localhost port=6666
dbname=postgres user=postgres' publication my_first_publication;
CREATE SUBSCRIPTION
```

The initial copy of the data happens automatically by default

Requires **wal_level = logical**

Demo
Upgrade from PostgreSQL 9.2.21 to 9.6.3

Conclusion

PostgreSQL upgrade best practices

Make sure you read the release notes

Minor upgrades usually are simple: Install the new binaries and switch your cluster over, done

For major upgrades the recommended method is pg_upgrade when you can stay on the same host

- Otherwise combine pg_dumpall, pg_dump and pg_restore

Please, please stay on a supported version and test, test, test your upgrade procedure

Any questions? Please do ask

Daniel Westermann
Senior Consultant
Open Infrastructure Technology Leader

+41 79 927 24 46
daniel.westermann@dbi-services.com

We look forward to working with you!