WAL Reduction

Rahila Syed

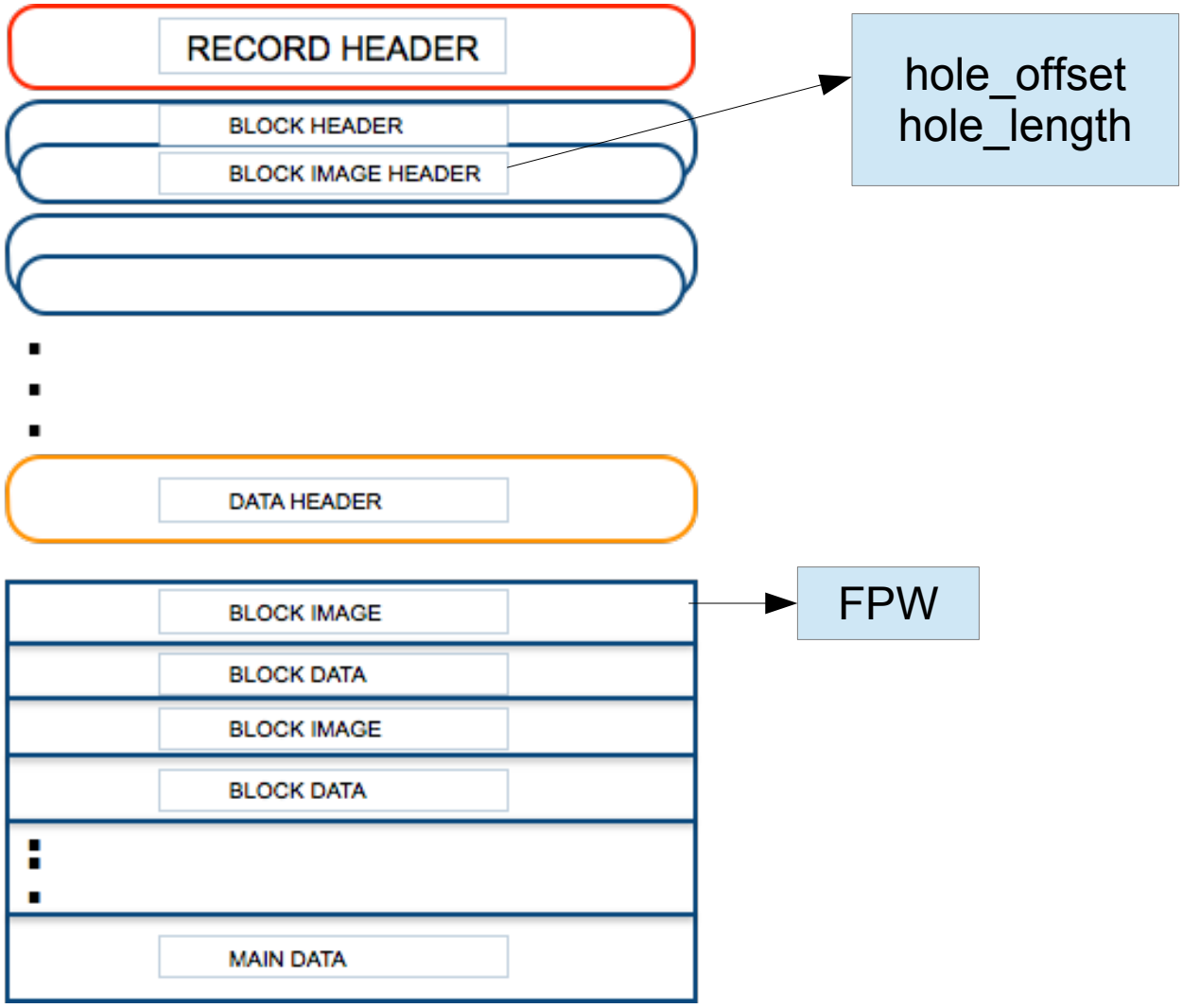
Agenda

- Write Ahead Log(WAL)
  - Structure Of WAL Record
  - FPW(Full page writes)
- Application Of WAL
  - Archiving
  - Streaming Replication
  - Recovery
- Need for WAL Reduction
- Ways to Reduce WAL size
  - WAL compression during streaming replication
  - Compressing archive segments
  - Logging page without hole(FPW)
  - Compression of WAL record, FPW compression
  - WAL reduction for Update Operation

Write Ahead Log(WAL)

- Provides data integrity
- Sequence of WAL Records
- One record of each change
- WAL is written in unit of 8k pages
- WAL files are 16MB segments(configurable)

Structure Of WAL Record



Full Page Writes(FPW)

- Avoids unrecoverable data corruption
- Block images backed up in WAL
- On first modification of the page after checkpoint
- Controlled by GUC named full_page_writes
- Occupies large portion of WAL.

Application Of WAL

- WAL Archiving
  - Back-up of WAL segments
  - WAL segments can be recycled once archived
- Streaming replication
  - Continuous shipping of WAL records from master to standby
  - Standby server is kept upto date
- Recovery
  - Replaying the WAL files to restore the database

We Need to Reduce WAL Size Because..

- WAL files occupy large area on disk especially in large OLTP databases
- There is an increase in disk writes due to regular fsync()s required
- Introduces delays in streaming replication
- Slows down transactions during synchronous replication

Ways to Reduce WAL Size

- Compressing archive segments
- Compression of WAL during streaming replication
  - SSH
  - SSL
- Logging full page image without hole
- Compression of WAL record
  - Record compression
  - FPW compression
- Reduction in WAL for update operation

WAL Compression during Streaming Replication-SSL

- WAL records can be compressed while streaming from master to standby
- SSL needs to be enabled while configuring PostgreSQL using - with-openssl
- SSL connection can be established between master and standby by setting the sslmode='require' in the primary_conninfo parameter in recovery.conf file on standby.
- Edit the replication entry in pg_hba.conf. It should use "hostssl" instead of "host"

WAL Compression during Streaming Replication-SSH

- SSH tunnel can be established between master and standby
- Run ssh -L -C 3333:localhost:5432 rahila@172.26.126.156 on standby
- This will establish a ssh tunnel starting at port 3333 on the standby and ending at port 5432 on master (ip 172.26.126.156)
- To establish streaming replication using SSH tunnel use localhost for replication in pg_hba.conf on master

```
# TYPE  DATABASE        USER            CIDR-ADDRESS          METHOD □
# Replication connections
□host    replication     replica         127.0.0.1/32          trust
```

SSH compression continued

- Following connection string is to be used in recovery.conf on standby primary_conninfo = 'host=127.0.0.1 port=3333 user=replica

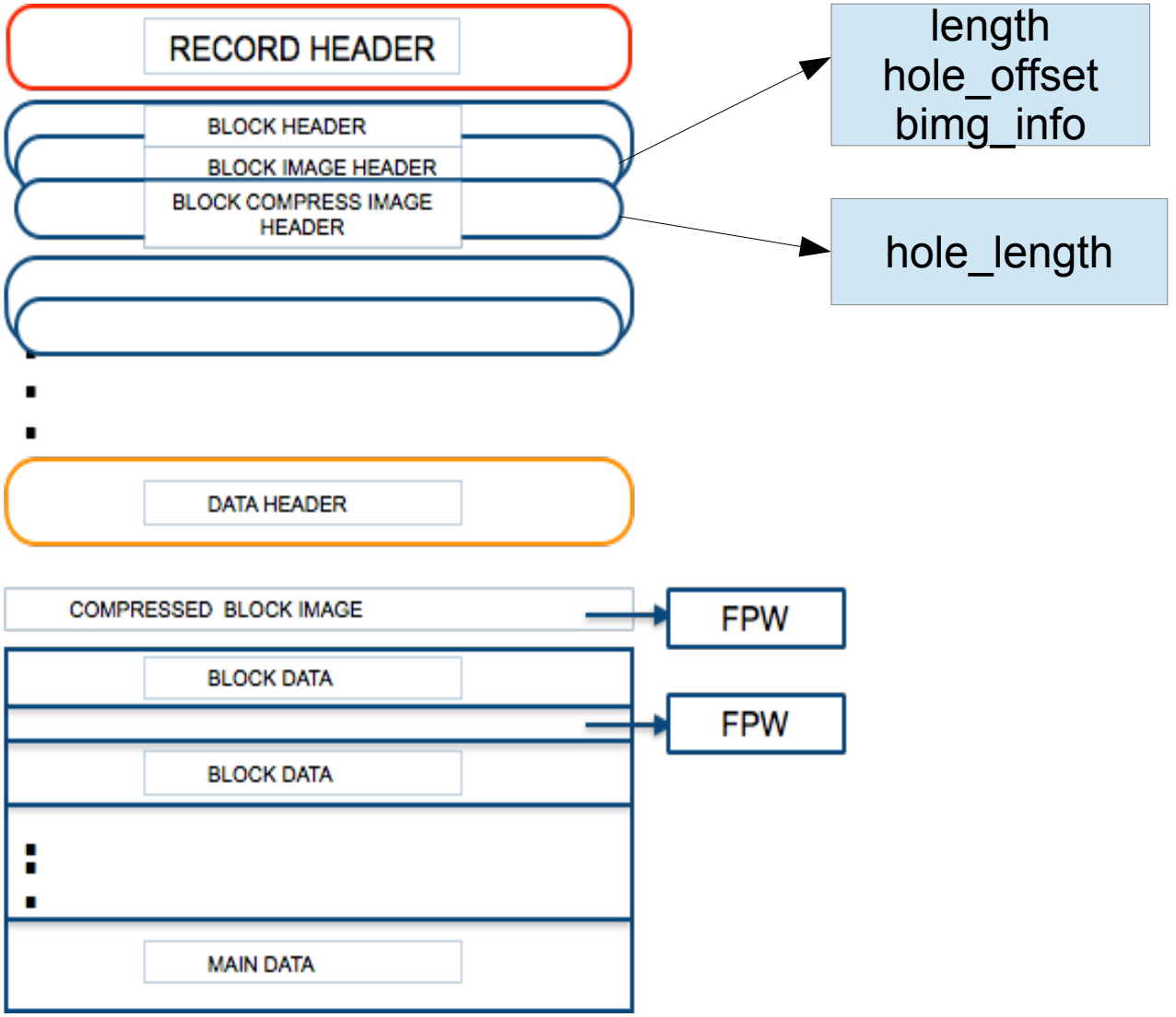
Comparison SSH vs SSL

	Asynchronous replication	SSL	SSH
Time:	1 hr 17 mins 58 secs	36 mins 02 secs	36 mins 25 secs
Kbytes Transferred:	1258403.744 KB	591255 KB **Compression**: 53 %	482537 KB **Compression**: 61.6 %
Avg CPU utilization(%user)	Standby: 0.067 Master: 0.031	Standby: 0.231 Master: 0.284	Standby: 0.119 Master: 0.209
Avg CPU utilization(%sys)	Standby: 0.114 Master: 0.059	Standby: 0.120 Master: 0.060	Standby: 0.129 Master:0.075
Avg CPU utilization(%iowait)	Standby: 0.099 Master:0.009	Standby: 0.152 Master: 0.009	Standby: 0.222 Master: 0.009

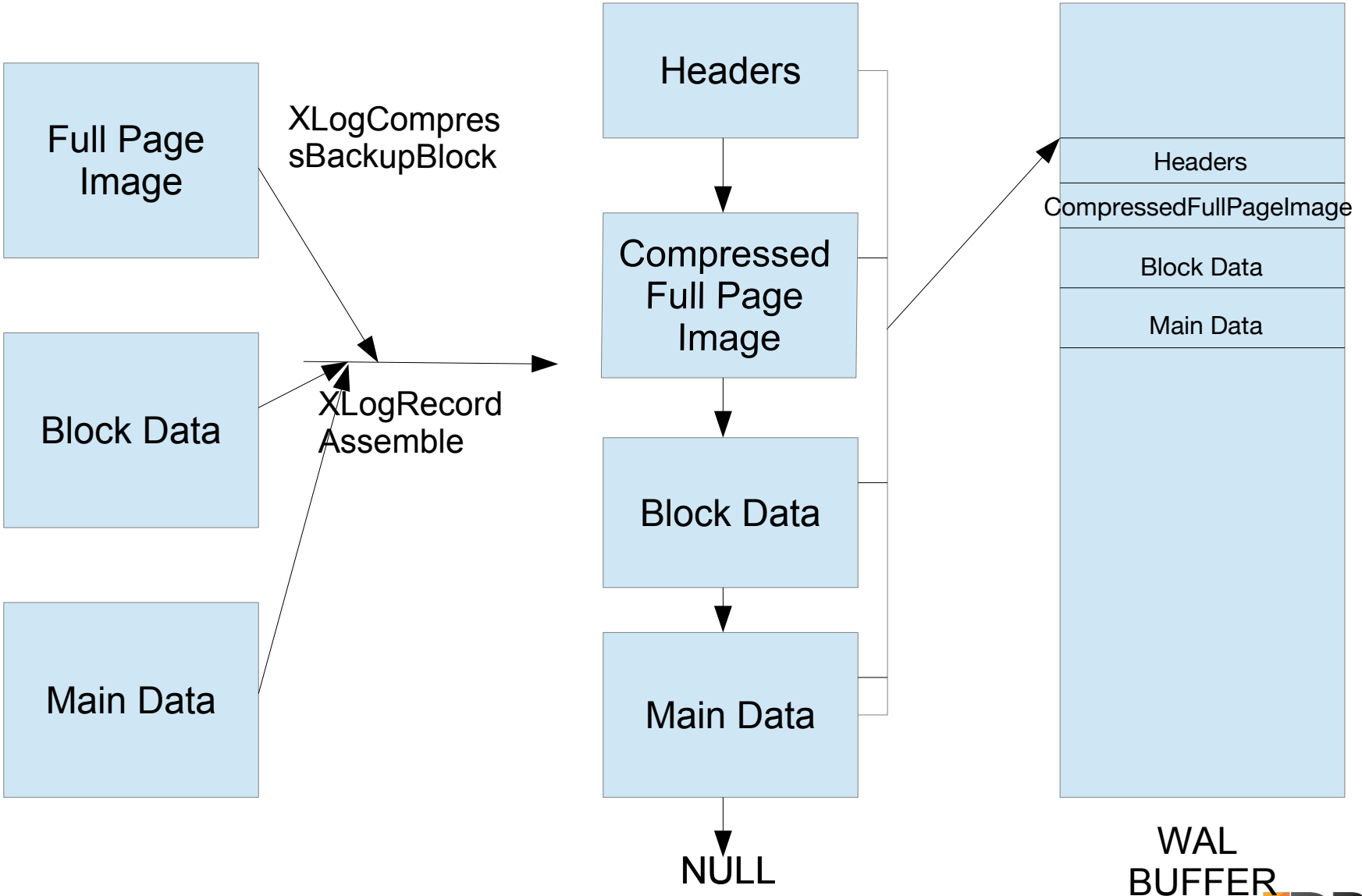
Compression of FPW(9.5)

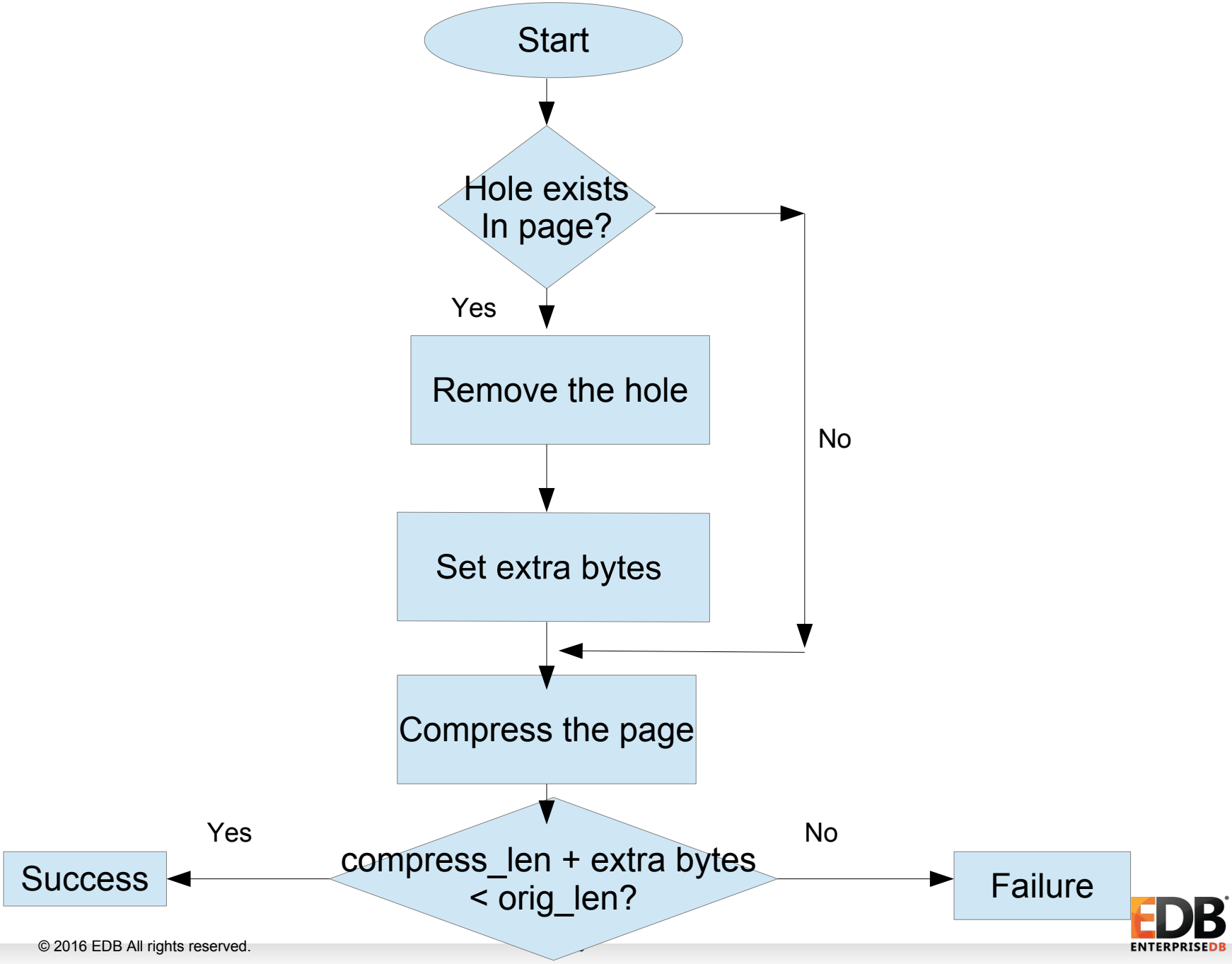
- When GUC wal_compression = 'on' FPWs in WAL are compressed
- FPWs are compressed at the time of constructing a WAL record
- Uses built-in compression pglz
- Compresses FPWs during basebackup
- Compressed image is decompressed during WAL-replay
- Reduces WAL volume, at the cost of some extra CPU cycles

Changes in WAL Record



FPW compression internals





Performance of FPW with write-only transaction

- Server config - Intel® Xeon ® Processor E5-2650 (2 GHz, 8C/16T, 20 MB) * 2 nos .
- RAM: 32GB Disk : HDD 450GB

Compression	Amount of WAL	TPS	CPU usage
On	1.845 GB	427.83	350.36 ms
Off	2.043 GB	428.51	342.06 ms
	Reduction in WAL % 9. 69	TPS remains almost same	Impact on CPU was very less

Performance of FPW with Synchronous Replication

- Server config - Intel® Xeon ® Processor E5-2650 (2 GHz, 8C/16T, 20 MB) * 2 nos .
- RAM: 32GB Disk : HDD 450GB

Compression	Amount of WAL	TPS	CPU usage
On	45.50 GB	362.82	847.47ms
Off	70.50 GB	318.13	548.36 ms
	WAL reduction % 35.48	Increase in TPS %14.05	Increase in CPU usage % 54.4

Recent Synchronous Replication performance numbers

- Server config - Intel x86, 8 sockets, 64 cores, 128 hardware threads, 500GB RAM
- Shared buffers: 8GB pgbench scale: 100

Compression	Amount of WAL	TPS	CPU usage
On	764.34 MB	320.64	165.12ms
Off	942.70 MB	311.38	163.14 ms
	18.9% WAL Reduction		

Recovery Speed

Compression :on	Compression :off	Conclusion
52.25 secs	62.94 secs	Reduction in recovery time 17%

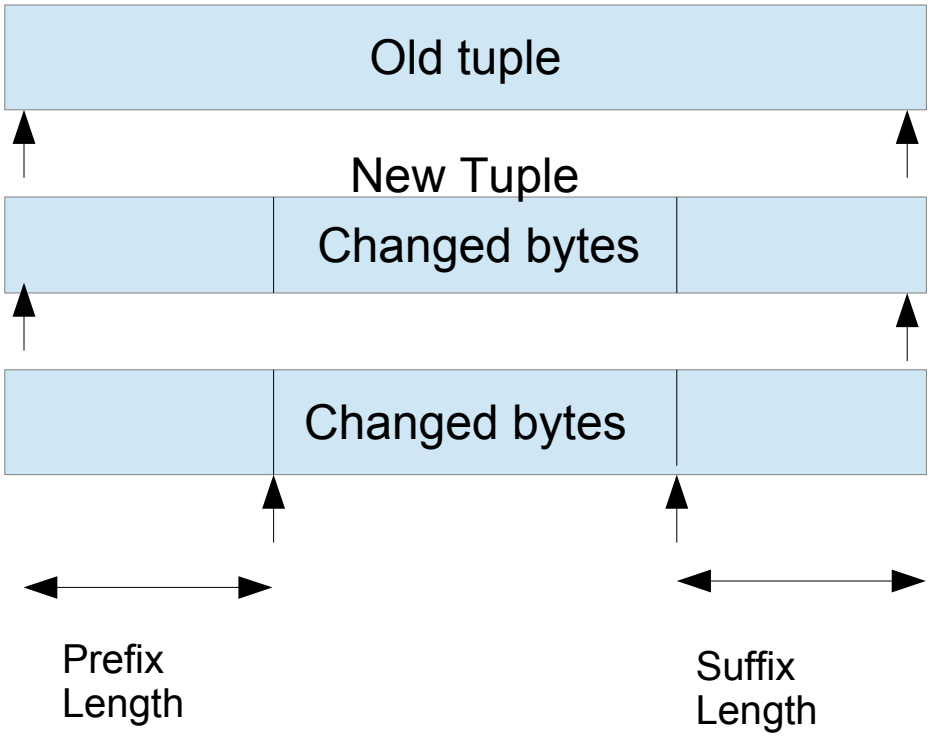
Security Vulnerability

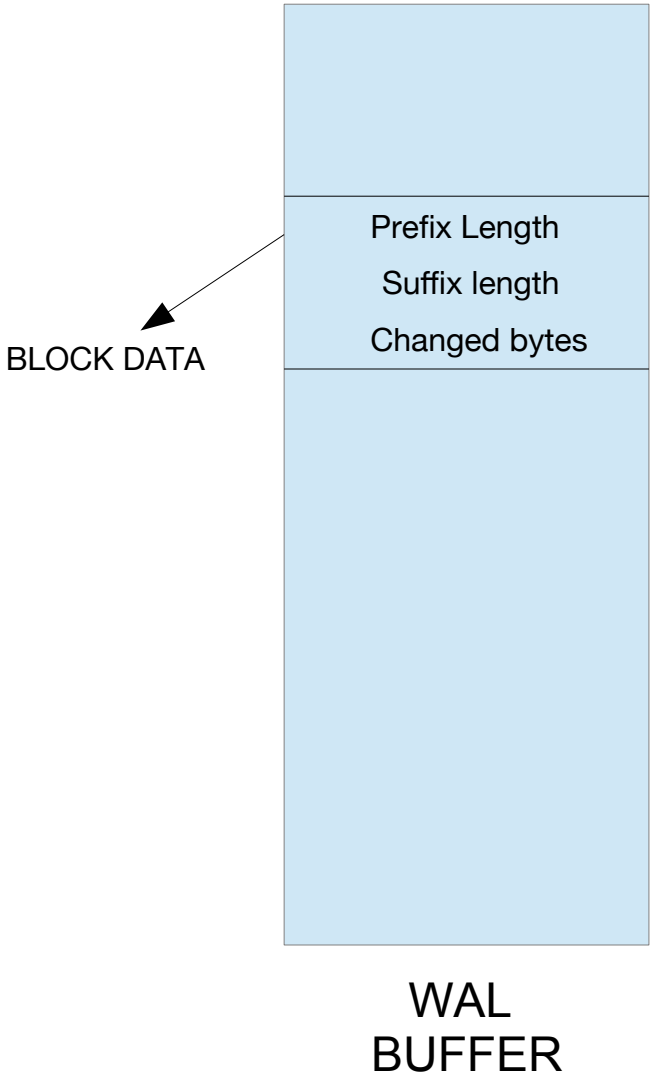
- The compressibility of a full page image can give hints about the content of the page
- CRIME and BREACH attacks on SSL
- Prohibit non-superusers from tweaking WAL compression
- wal_compression is made PGC_SUSET

WAL Reduction for Update Operation(9.4)

- Only log the changed portion of new tuple
- Old and new tuple should be on same page
- Useful when most fields are not modified

Construction of Reduced WAL Record





Replay of the reduced tuple

- At the time of WAL replay, old buffer is fetched from disk
- While applying new tuple changes, old tuple data of prefix length from old buffer is copied to the new tuple
- This is followed by the changed bytes of new tuple from the WAL log.
- Changed bytes of new tuple are followed by data equal to suffix length from old tuple

Performance measurements for WAL reduction with update

Unpatched

Test Name	WAL generated	Duration
two short fields, no change	619488208	21.44
two short fields, one changed	624834688	21.53
two short fields, both changed	624858696	22.32
one short and one long field, no change	519313936	9.93
ten tiny fields, all changed	697886872	26.65
hundred tiny fields, all changed	319147488	14.90
hundred tiny fields, half changed	319062960	14.64
hundred tiny fields, half nulled	286996768	13.48
9 short and 1 long, short changed	345818336	8.86

Performance measurements for WAL reduction with update

With WAL reduction for update patch

Test Name	WAL generated	Duration
two short fields, no change	585103464	23.34
two short fields, one changed	624786760	22.71
two short fields, both changed	624795128	22.89
one short and one long field, no change	323185264	8.72
ten tiny fields, all changed	700236720	26.91
hundred tiny fields, all changed	319187640	14.91
hundred tiny fields, half changed	319076384	14.77
hundred tiny fields, half nulled	240637008	13.48
9 short and 1 long, short changed	257548416	8.84

THANK YOU