Maintaining Terabytes

Selena Deckelmann
Emma, Inc - http://myemma.com
PostgreSQL Global Development Group

PgCon - May 18-20, 2011

http://tech.myemma.com
@emmaemailtech

PgCon - May 18-20, 2011

Environment at Emma

- 1.6 TB, 1 cluster, Version 8.2 (RAID10)
- 1.1 TB, 2 clusters, Version 8.3 (RAID10)
- 8.4, 9.0 Dev
- Putting 9.0 into production (June 2011)
- pgpool, Redis, RabbitMQ, NFS

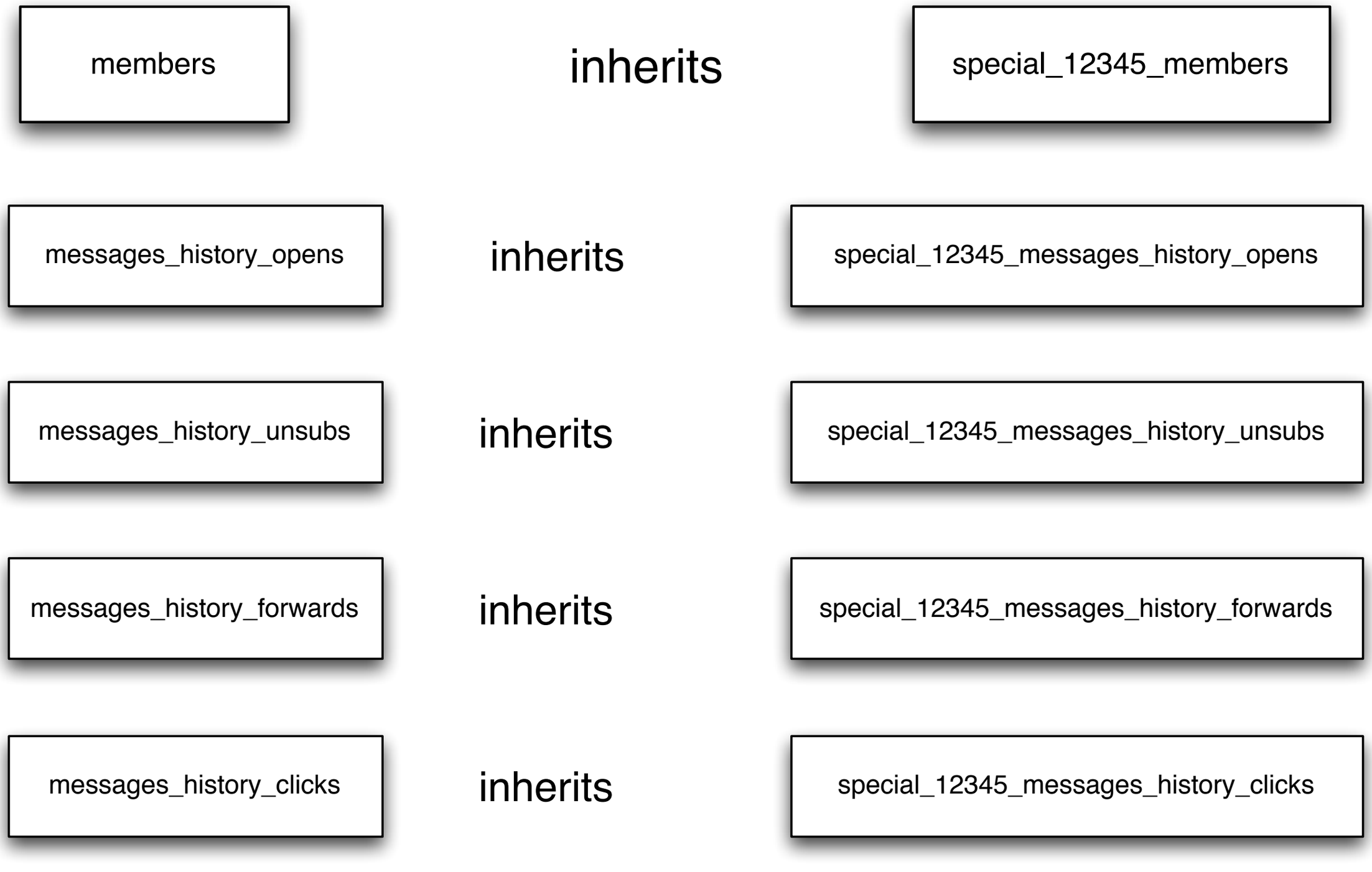
Other stats

- peaks: ~3000 commits per second
- average writes: 4 MBps
- average reads: 8 MBps
- From benchmarks we've done, load is pushing the limits of our hardware.

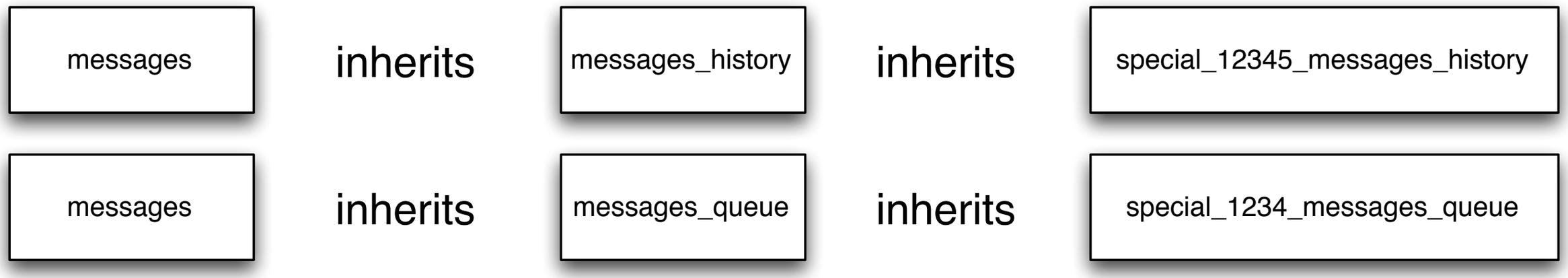
I say all of this with love.

Our schema.

For each customer....



For each customer....



Huge catalogs

- 409,994 tables (and counting)
- Minor mistake in parent table definitions
  - Parent table updates take 30+ minutes

not null default nextval('important_sequence'::regclass)

vs

not null default nextval('important_sequence'::text)

Huge catalogs

- Bloat in the catalog
- User-provoked ALTER TABLE
- VACUUM FULL of catalog takes 2+ hrs

Huge catalogs suck

- 9,019,868 total data points for table stats
- 4,550,770 total data points for index stats
- Stats collection is slow

Disk Management

- $PGDATA:
  - pg_tblspc (TABLESPACES)
  - pg_xlog - separate disks!
  - global/pg_stats - tmpfs!
  - wal for warm standby - another volume!

Problems we worked through with big schemas

- Bloat
- Backups
- System resource exhaustion
- Minor upgrades
- Major upgrades
- Transaction wraparound

Bloat Causes

- Frequent UPDATE patterns using tables as queues for email
- Frequent DELETEs without VACUUM
  - a terabyte of dead tuples

BLOAT QUERY

```
SELECT
  schemaname, tablename, reltuples::bigint, relpages::bigint, otta,
  ROUND(CASE WHEN otta=0 THEN 0.0 ELSE sml.relpages/otta::numeric END,1) AS tbloat,
  CASE WHEN relpages < otta THEN 0 ELSE relpages::bigint - otta END AS wastedpages,
  CASE WHEN relpages < otta THEN 0 ELSE bs*(sml.relpages-otta)::bigint END AS wastedbytes,
  CASE WHEN relpages < otta THEN '0 bytes'::text ELSE (bs*(relpages-otta))::bigint || ' bytes' END AS wastedsize,
  iname, ituples::bigint, ipages::bigint, iotta,
  ROUND(CASE WHEN iotta=0 OR ipages=0 THEN 0.0 ELSE ipages/iotta::numeric END,1) AS ibloat,
  CASE WHEN ipages < iotta THEN 0 ELSE ipages::bigint - iotta END AS wastedipages,
  CASE WHEN ipages < iotta THEN 0 ELSE bs*(ipages-iotta) END AS wastedibytes,
  CASE WHEN ipages < iotta THEN '0 bytes' ELSE (bs*(ipages-iotta))::bigint || ' bytes' END AS wastedisize
FROM (
  SELECT
    schemaname, tablename, cc.reltuples, cc.relpages, bs,
    CEIL((cc.reltuples*((datahdr+ma-
      (CASE WHEN datahdr%ma=0 THEN ma ELSE datahdr%ma END))+nullhdr2+4))/(bs-20::float)) AS otta,
    COALESCE(c2.relname,'?') AS iname, COALESCE(c2.reltuples,0) AS ituples, COALESCE(c2.relpages,0) AS ipages,
    COALESCE(CEIL((c2.reltuples*(datahdr-12))/(bs-20::float)),0) AS iotta
  FROM (
    SELECT
      ma,bs,schemaname,tablename,
      (datawidth+(hdr+ma-(case when hdr%ma=0 THEN ma ELSE hdr%ma END)))::numeric AS datahdr,
      (maxfracsum*(nullhdr+ma-(case when nullhdr%ma=0 THEN ma ELSE nullhdr%ma END))) AS nullhdr2
    FROM (
      SELECT
        schemaname, tablename, hdr, ma, bs,
        SUM((1-null_frac)*avg_width) AS datawidth,
        MAX(null_frac) AS maxfracsum,
        hdr+(
          SELECT 1+count(*)/8
          FROM pg_stats s2
          WHERE null_frac<>0 AND s2.schemaname = s.schemaname AND s2.tablename = s.tablename
        ) AS nullhdr
      FROM pg_stats s, (
        SELECT
          (SELECT current_setting('block_size')::numeric) AS bs,
          CASE WHEN substring(v,12,3) IN ('8.0','8.1','8.2') THEN 27 ELSE 23 END AS hdr,
          CASE WHEN v ~ 'mingw32' THEN 8 ELSE 4 END AS ma
        FROM (SELECT version() AS v) AS foo
      ) AS constants
      GROUP BY 1,2,3,4,5
    ) AS foo
  ) AS rs
  JOIN pg_class cc ON cc.relname = rs.tablename
  JOIN pg_namespace nn ON cc.relnamespace = nn.oid AND nn.nspname = rs.schemaname AND nn.nspname <> 'information_schema'
  LEFT JOIN pg_index i ON indrelid = cc.oid
  LEFT JOIN pg_class c2 ON c2.oid = i.indexrelid
) AS sml
WHERE tablename = 'addr'
ORDER BY wastedbytes DESC LIMIT 1;
```

Use check_postgres.pl

https://github.com/bucardo/check_postgres/

PgCon - May 18-20, 2011

Fixing bloat

- Wrote scripts to clean things up
  - VACUUM (for small amounts)
  - CLUSTER
  - TRUNCATE (data loss!)
  - Or most extreme: DROP/CREATE
- And then ran the scripts.

Backups

- pg_dump takes longer and longer

```
   backup    |        duration
-------------+------------------------
 2009-11-22 | 02:44:36.821475
 2009-11-23 | 02:46:20.003507
 2009-11-24 | 02:47:06.260705
 2009-12-06 | 07:13:04.174964
 2009-12-13 | 05:00:01.082676
 2009-12-20 | 06:24:49.433043
 2009-12-27 | 05:35:20.551477
 2010-01-03 | 07:36:49.651492
 2010-01-10 | 05:55:02.396163
 2010-01-17 | 07:32:33.277559
 2010-01-24 | 06:22:46.522319
 2010-01-31 | 10:48:13.060888
 2010-02-07 | 21:21:47.77618
 2010-02-14 | 14:32:04.638267
 2010-02-21 | 11:34:42.353244
 2010-02-28 | 11:13:02.102345
```

Backups

- pg_dump fails
- patching pg_dump for SELECT ... LIMIT
- Crank down shared_buffers
- or...

http://seeifixedit.com/view/there-i-fixed-it/45

Install 32-bit Postgres and libraries on a 64-bit system.

Install 64-bit Postgres/libs of the same version.

Copy “hot backup” from 32-bit sys over to 64-bit sys.

Run pg_dump from 64-bit version on 32-bit Postgres.

PSA

- Warm standby is not a backup
- Hot backup instances
- “You don’t have valid backups, you have valid restores.” (thanks @sarahnovotny)
- Necessity is the mother of invention...

Ship WAL from Solaris x86 -> Linux
It did work!

Running out of inodes

- UFS on Solaris
  "The only way to add more inodes to a UFS filesystem is:
  1. destroy the filesystem and create a new filesystem with a higher inode density
  2. enlarge the filesystem - growfs man page"
- Solution 0: Delete files.
- Solution 1: Sharding and bigger FS on Linux
- Solution 2: ext4 (soon!)

Running out of available file descriptors

- Too many open files by the database
- Solution 1: Pooling - pgpool-II or pgbouncer?
- Solution 3: Shard

Minor upgrades

- Stop/start database
- CHECKPOINT() before shutdown

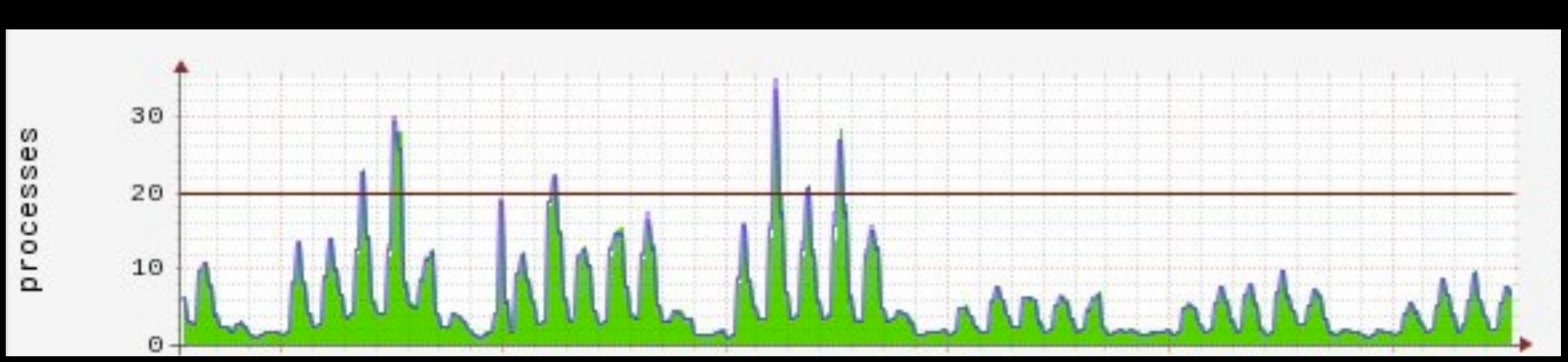
Major Version upgrades

- Too much downtime to dump/restore
- Write tools to migrate data
- Trigger-based replication
- `pg_upgrade`

Transaction wraparound avoidance

- autovacuum triggers too frequently
  - 200,000 transactions (2 days)
  - Watch age(datfrozenxid)
  - Increase autovacuum_freeze_max_age


Transactions
1.2 G
1.0 G
8.8 G
8.6 G
8.4 G
8.2 G



Thanks!

- We're hiring! - selena@myemma.com
- Emma's Tech Blog: http://tech.myemma.com
- My blog: http://chesnok.com
- http://twitter.com/selenamarie