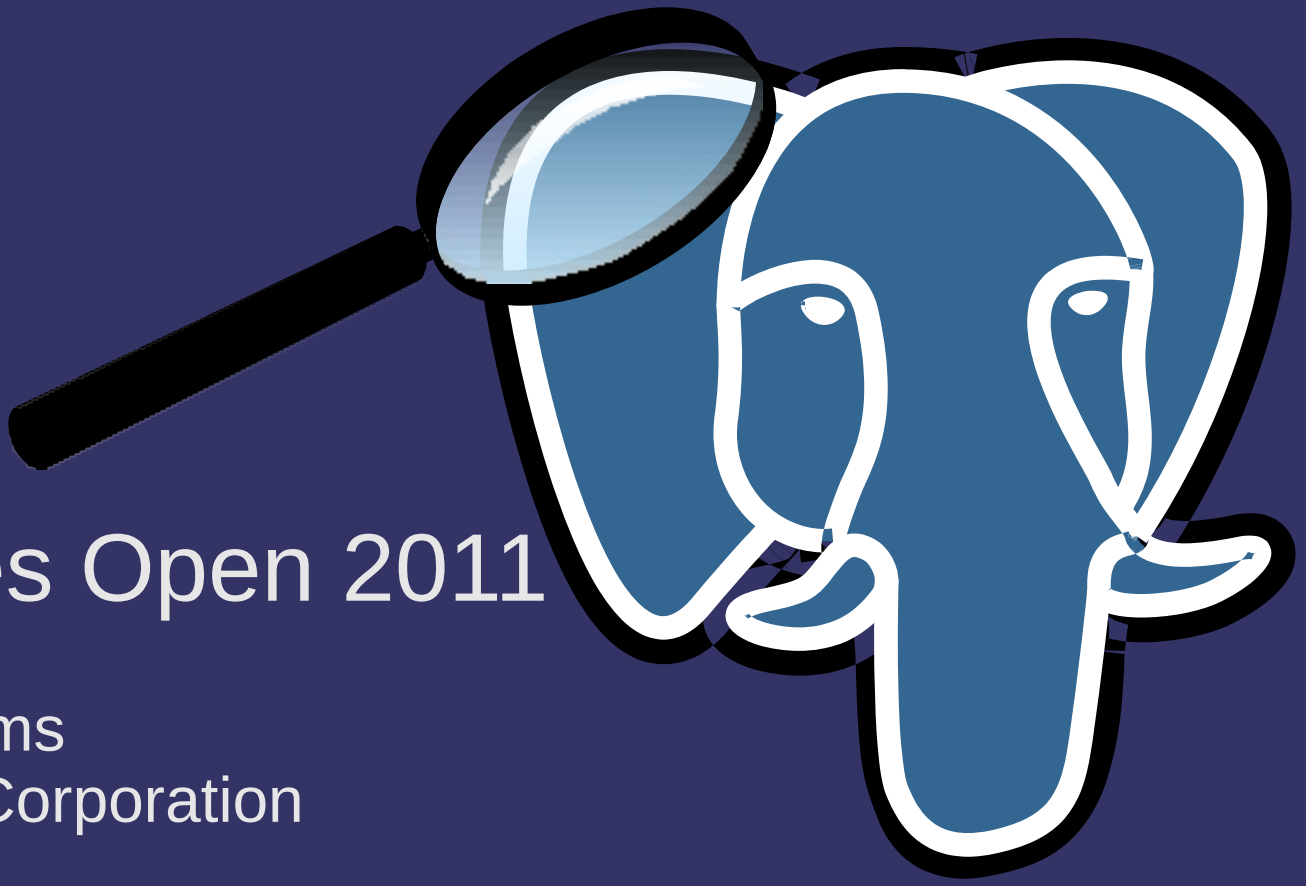
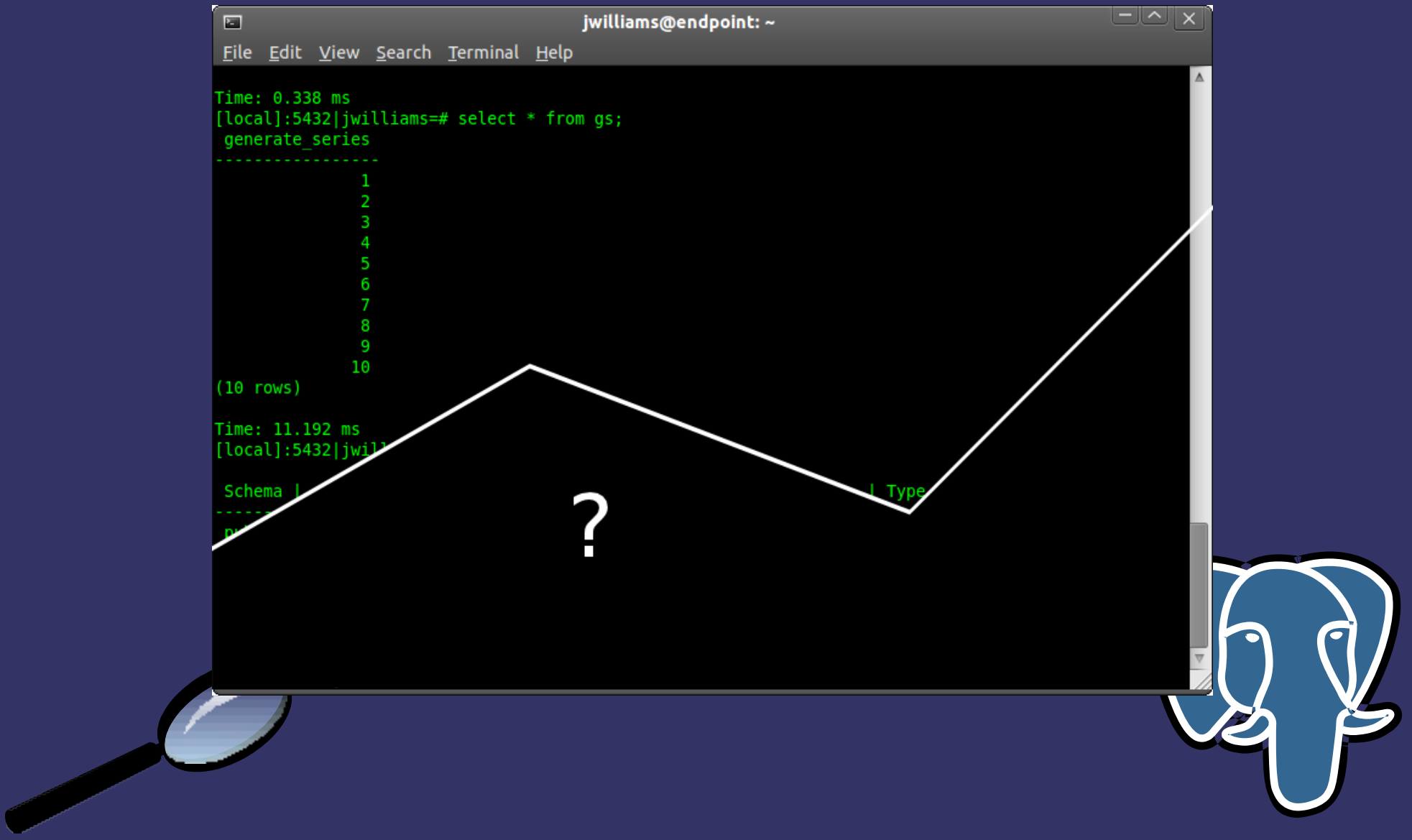
Monitor the Heck Out Of Your Database



Postgres Open 2011

Josh Williams
End Point Corporation

Data in...
Data out...



What are we looking for?
Why do we care?

- Performance of the system
- Application throughput
- Is it dead … about to die?

Different people care about different things

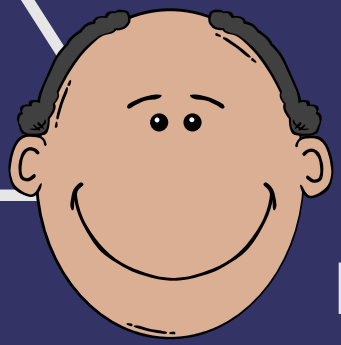
Devs

System Performance

Application Throughput

Ops

Is it about to die?



PHB's

Monitoring Postgres

- Log Monitoring for errors
- Log Monitoring for query performance
- Control files / External commands
- Statistics from the database itself

Log Monitoring for Error Conditions

- ERROR: division by zero
- FATAL: password authentication failed for user “postgres”
- PANIC: could not write to file “pg_xlog/...”: No space left on device
- **tail_n_mail**

tail_n_mail

- http://bucardo.org/wiki/tail_n_mail
- Written in Perl, requires Net::SMTP::SSL
- … or *sendmail* binary
- Slight misnomer
- Sample Config: tail tail_n_mail

tail_n_mail

- Create a file: tail_n_mail.config:

```
EMAIL: postgres_OMG@endpoint.com
FROM: postgres@server.local
MAILSUBJECT: PG ERROR REPORT
FILE: /var/log/pgsql-%Y-%m-%d.log
INCLUDE: FATAL:
INCLUDE: PANIC:
INCLUDE: ERROR:
```

tail_n_mail

- Create a file: ~/.tailnmailrc:

```
log_line_prefix: '%t [%p]: [%l-1] %u@d '
```

… or …

```
pgmode: syslog
```

- ➲ (Check your postgresql.conf)

tail_n_mail

- Test: perl tail_n_mail tail_n_mail.config
- Schedule it to run every minute:

```
* * * * * perl tail_n_mail
  --quiet tail_n_mail.config
```

```
[1] (between lines 45 and 57, occurs 5 times)
cp: cannot stat `/data/wal_archive/00000001000003AA000000B9': No such file or directory

[2] (from line 44)
2011-09-08 21:25:30 UTC @ [31774]
FATAL: replication terminated by primary server

[3] (from line 48)
2011-09-08 21:25:30 UTC @ [19657]
FATAL: could not connect to the primary server: FATAL: the database system is shutting down FATAL: the database system is
shutting down

[4] (from line 53)
2011-09-08 21:25:56 UTC @ [19666]
FATAL: could not connect to the primary server: could not connect to server: Connection timed out Is the server running on
```

tail_n_mail

- Copy, edit tail_n_mail.config:

```
EMAIL: postgres_OMG@endpoint.com
FROM: postgres@server.local
MAILSUBJECT: PG ERROR REPORT
FILE: /var/log/pgsql-%Y-%m-%d.log
INCLUDE: FATAL:
INCLUDE: PANIC:
INCLUDE: ERROR:
EXCLUDE: ERROR:  duplicate key
^ value violates unique constraint
```

Log Monitoring for Query Performance

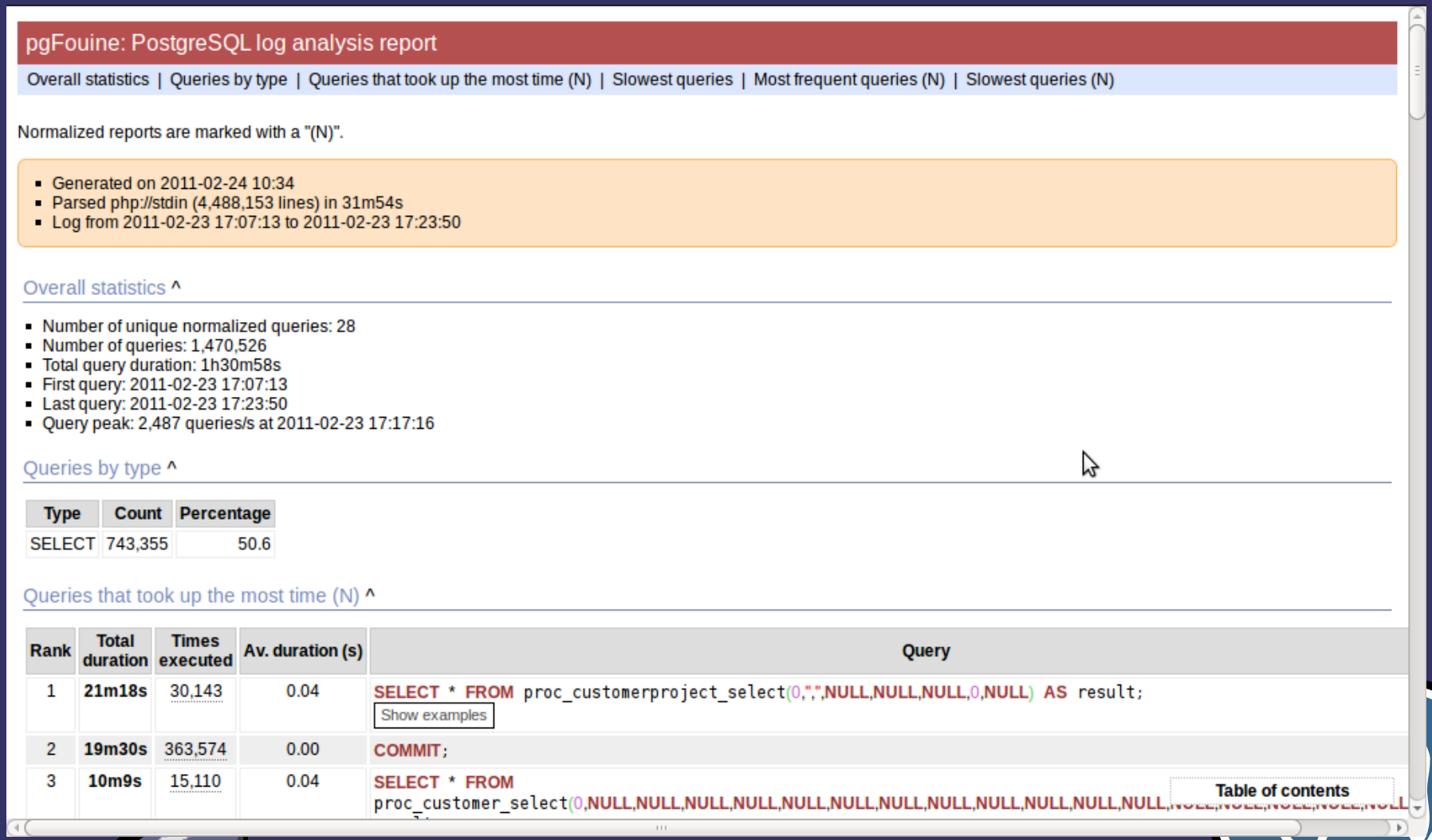
- By default successful SQL isn't logged
- Set in postgresql.conf:
- log_statement = 'ddl'
- log_min_duration_statement = 200
- log_line_prefix = '%t [%p]: [%l-1] %u@%d '
- **pgFouine**
- **pgsi**

pgFouine

- http://pgfouine.projects.postgresql.org/
- Written in PHP (???)
- A little quirky, esp. regarding line prefix
- For scheduled task, write a wrapper script

```
cat $logdir/postgres-$yesterday.log \
| bin/pgfouine -logtype stderr -memorylimit 512
> pgfouine-$yesterday.html
```

pgFouine



pgFouine: PostgreSQL log analysis report

Overall statistics | Queries by type | Queries that took up the most time (N) | Slowest queries | Most frequent queries (N) | Slowest queries (N)

Normalized reports are marked with a "(N)".

- Generated on 2011-02-24 10:34
- Parsed php://stdin (4,488,153 lines) in 31m54s
- Log from 2011-02-23 17:07:13 to 2011-02-23 17:23:50

Overall statistics ^

- Number of unique normalized queries: 28
- Number of queries: 1,470,526
- Total query duration: 1h30m58s
- First query: 2011-02-23 17:07:13
- Last query: 2011-02-23 17:23:50
- Query peak: 2,487 queries/s at 2011-02-23 17:17:16

Queries by type ^

Type	Count	Percentage
SELECT	743,355	50.6

Queries that took up the most time (N) ^

Rank	Total duration	Times executed	Av. duration (s)	Query
1	**21m18s**	30,143	0.04	`SELECT * FROM proc_customerproject_select(0,',',NULL,NULL,NULL,0,NULL) AS result;` Show examples
2	**19m30s**	363,574	0.00	`COMMIT;`
3	**10m9s**	15,110	0.04	`SELECT * FROM proc_customer_select(0,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,...`

Table of contents

pgsi – the Postgres System Impact report

- http://bucardo.org/wiki/pgsi
- Written in Perl
- Little more tolerant of log_line_prefix
- Invocation similar, write a wrapper script

```
bin/pgsi.pl --quiet \
 --file=$logdir/postgres-$yesterday.log \
> pgsi-$yesterday.html
```

pgsi – the Postgres System Impact report

- SELECT (23)
- SET (1)
- BEGIN (1)
- COMMIT (1)

Query System Impact : SELECT

Log activity from 2011-02-23 17:07:13 to 2011-02-23 17:23:50

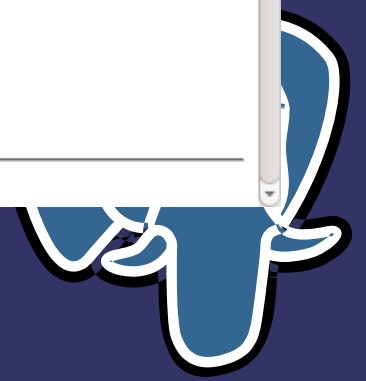
System Impact:	**128.28**
Mean Duration:	42.43 ms
Median Duration:	41.61 ms
Total Count:	30143
Mean Interval:	33.08 ms
Std. Deviation:	7.46 ms

```
SELECT *
 FROM proc_customerproject_select(?, ?, ?, ?, ?, ?, ?, ?) AS result
```

System Impact:	**61.11**
Mean Duration:	40.32 ms
Median Duration:	39.36 ms
Total Count:	15110
Mean Interval:	65.98 ms
Std. Deviation:	6.23 ms

```
SELECT *
 FROM proc_customer_select(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?) AS result
```

System Impact:	**46.88**



External Commands

- But that's OLD stuff. I want to see NOW.
- And besides, who relies on email anymore?
- **check_postgres**

check_postgres

- Intended to plug into Nagios or similar
- Collection of several monitoring actions
- http://bucardo.org/wiki/check_postgres
- Written in Perl (seeing a pattern?)
- Requirements ~~ vary depending on action
- check_postgres.pl –action=foo
- --include=specific-object --exclude=objects
- --warning=X --critical=Y

check_postgres – Important Actions

- --action=backends
- Connections, and % of max_connections
- Thresholds really flexible: %, remaining
- Count active connections with --noidle
- --action=bloat --db=bar
- Heuristics to figure out wasted space
- --include specific tables or indexes

check_postgres – --action=bloat

Analyze only when necessary.

- Take a look at the query, search for:
- ## This was fun to write

check_postgres – Important Actions

- --action=query_time
- Long-running queries
- Complex thresholds: “2 for 10 minutes”
- --action=txn_idle
- Long-running idle transactions
- Mostly the same code as query_time
- --action=txn_wraparound
- Transactions since datfrozenxid

check_postgres – Additional Actions

- Useful for warm (or hot) standby:
- --action=checkpoint
- How long since the last checkpoint
- Uses pg_controldata, no PG connection
- --action=archive_ready
- Number of unarchived WAL files
- --action=hot_standby_delay
- Must be able to connect to both servers

check_postgres – Additional Actions

- Sanity Checks:
- --action=database_size
- No default thresholds
- --action=relation_size
- --action=wal_files
- --action=last_autovacuum / autoanalyze
- May be a little tricky

check_postgres – Additional Actions

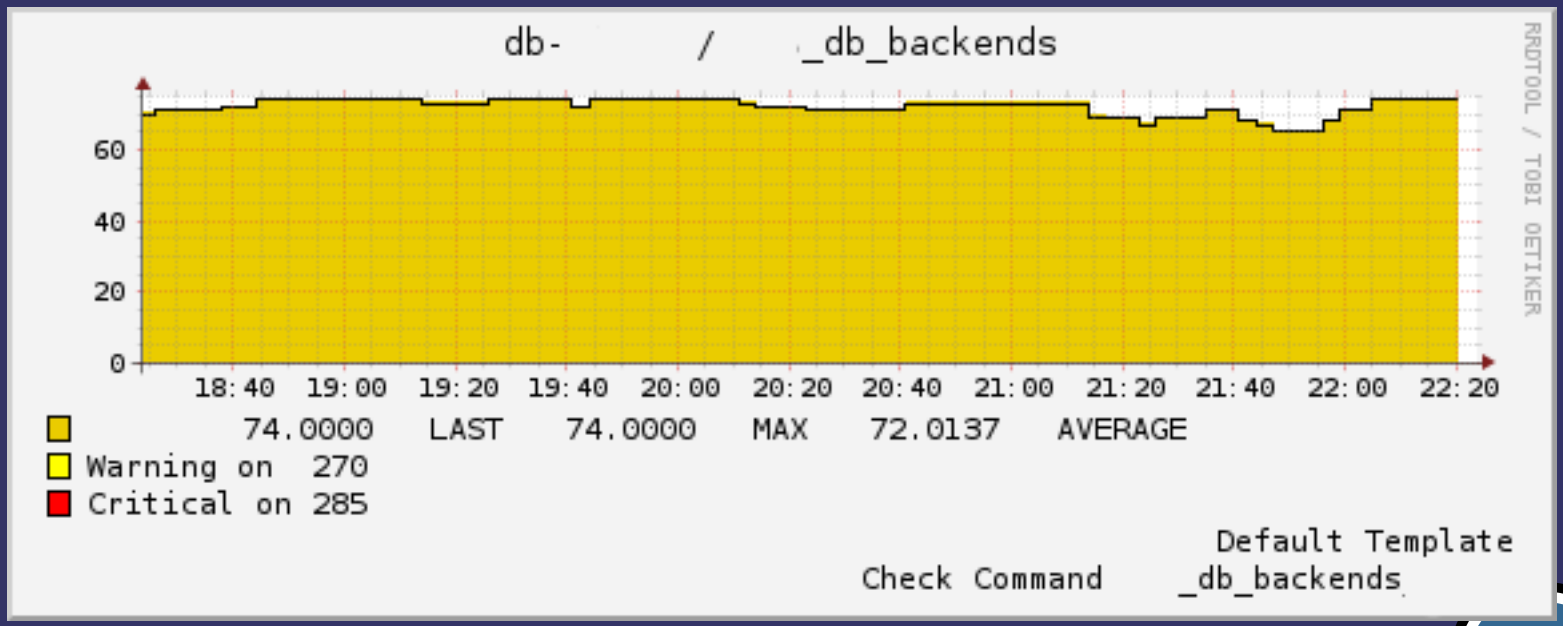
- --action=prepared_txns
- Age of prepared transactions
- --action=new_version_pg
- Something else? --action=custom_query
- Please let us know!
- --action=saneversion?

Live Metrics and Statistics

- check_postgres does metrics, too!
- --output={nagios,mrtg,cacti,simple}
- Metrics collection: Graphite / Cacti
- Other metrics sources:
- Getting Statistics from Postgres

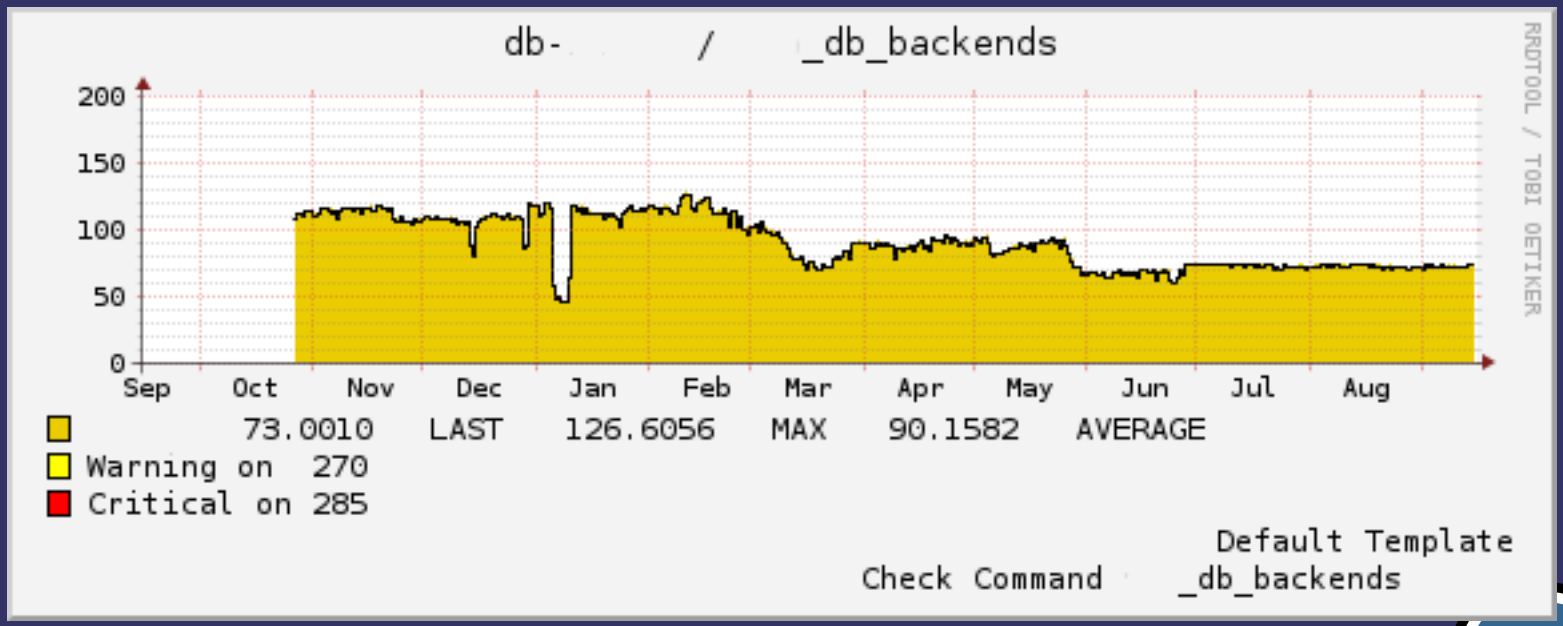
check_postgres – Metrics Actions

- --action=backends



check_postgres – Metrics Actions

- --action=backends



check_postgres – Metrics Actions

- --action=locks

```
POSTGRES_LOCKS OK: DB "postgres" total locks: 53 |
time=0.05s production.total=52
production.waiting=0 postgres.total=1
postgres.waiting=0
```

check_postgres – Metrics Actions

- --action=database_size
- Or more specifically: relation_size
- --action=dbstats
- Cacti-friendly pg_stat_database view
- --action=hitratio
- Keep track of buffer cache hit rate

Getting Statistics from Postgres … Directly

- If you don't have Cacti, Nagios, Graphite...
- Take a look at pg_stat_database:

```
[local]:5434|postgres=# select * from pg_stat_database;
 datid |  datname  | numbackends | xact_commit | xact_rollback |
-------+-----------+-------------+-------------+---------------+
     1 | template1 |           0 |           0 |             0 |
 11939 | template0 |           0 |           0 |             0 |
 11947 | postgres  |           2 |       18006 |             0 |
 16393 | foo       |           3 |          76 |            15 |
(4 rows)
```

Getting Statistics from Postgres … Directly

- If you don't have Cacti, Nagios, Graphite...
- Take a look at pg_stat_user_tables:

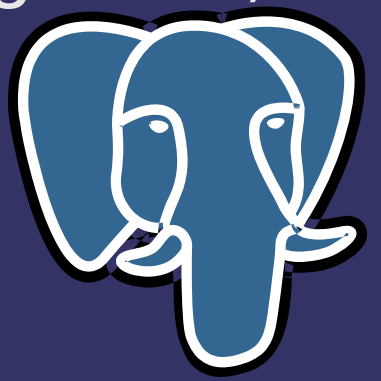
```
[local]:5432|postgres=# select * from pg_stat_user_tables ;
 relid | scheman | relname | seq_scan | seq_tup_read | idx_scan |
-------+---------+---------+----------+--------------+----------+
 36659 | public  | gs2     |        5 |     30000033 |          |
 19736 | public  | baz     |        5 |            0 |          |
 19739 | public  | gs      |        6 |     60000006 |          |
(3 rows)
```

Getting Statistics from Postgres … Directly

- If you don't have Cacti, Nagios, Graphite...
- Take a look at pg_stat_bgwriter:

```
[local]:5432|psql81=# select * from pg_stat_bgwriter;
 checkpoints_timed | checkpoints_req | buffers_checkpoint | bu...
-------------------+-----------------+--------------------+------
              1871 |              26 |              52956 |
(1 row)
```

- INSERT INTO bgwriter_history
- SELECT now(), * FROM pg_stat_bgwriter;
- Poor man's metrics collection!



Other Things to Monitor

- The OS environment, of course
- Hardware – Drive / RAID status!
- Load balancer / Connection pool
- Replication
- The application!

Other Things: OS Environment

- Nagios (etc) will have built-in system checks
- Never a bad idea!
- Also see: sysstat
- Periodic snapshots of CPU, network, etc

- Hardware: See what your vendor provides

Other Things: Application

- Load Balancer or Connection Pooler:
- Should provide their own metrics
- pgBouncer? check_postgres!

- Application:
- Give Graphite a look – correlation FTW!

Monitoring Postgres

- Log Monitoring for errors
- Log Monitoring for query performance
- Control files / External commands
- Statistics from the database itself

Questions?

Slides:

http://joshwilliams.name/talks/monitoring/