pgagroal

High-performance connection pool for PostgreSQL

Agenda

- pgagroal
  - p-g-a-gro-al
  - C17
  - 3-clause BSD license
- Architecture
- Features
- Deployment
- Performance
- Roadmap
- Closing thoughts

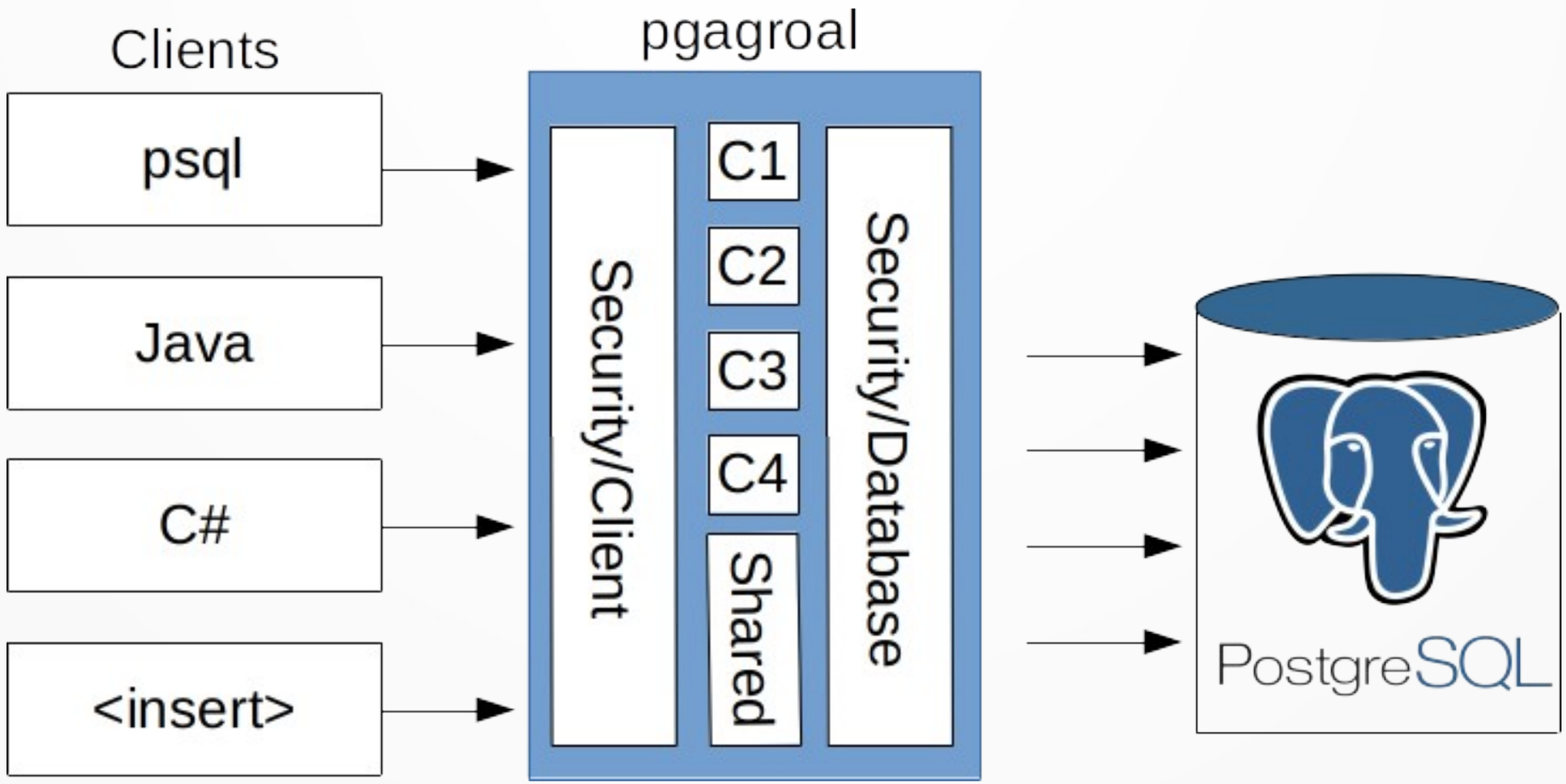
Connection pool

- Provide database connections to clients
- Provide a central access point to a database cluster

Architecture

- Process model
- Shared memory model across processes
- libev for fast network interactions
- Atomic operations are used to keep track of state
- The PostgreSQL native protocol v3 for its communication
- Dependencies
  - libev
  - OpenSSL

Architecture



Shared memory

- Configuration and state shared among all processes
  - mmap()
    - PROT_READ | PROT_WRITE
    - MAP_ANONYMOUS | MAP_SHARED
  - Configuration settings
  - State (atomic schar)
  - Servers
  - Limits
  - Access control
  - Users
  - Connections
- pgagroal.h, shmem.h, configuration.h, shmem.c and configuration.c

Security

- Client modules towards PostgreSQL
- Server modules for clients
- Types
  - Trust
  - Reject
  - Password
  - MD5
  - SCRAM-SHA-256
  - All (server)
- Mix’n’match

Security

- Password management
  - Master key
  - AES-256 based store
- Transport Layer Security v1.2+
  - Client pgagroal
- OpenSSL
- security.h, security.c

Pool

- Pool of connections
  - Maintain state of each connection
- API
  - pgagroal_get_connection
  - pgagroal_return_connection
  - pgagroal_kill_connection
  - Management operations
- Flexible member array
  - struct connection connections[]
- pgagroal.h, pool.h, pool.c

Limit

- Define connection limit
  - Database
  - User
  - Both
- Best matching rule
- Through atomic_ushort active_connections
- pgagroal.h, pool.c

Messages

- Protocol-native message format
  - PostgreSQL protocol version 3.x
- All message formats needed for “startup”
  - Request / response
  - Security
- Each process has a fixed memory buffer for communication
  - static struct message* message
  - static void* data
- Raw communication
  - Socket
  - SSL
- message.h, message.c, memory.h, memory.c

Client

- Process
- Authentication
  - Obtains a connection
- Setup pipeline
- Run
- Return or kill the connection
- exit(code)
- worker.h, worker.c

Pipeline

- Defines the behavior of interaction
  - Client pgagroal
  - pgagroal PostgreSQL
- Pipelines
  - Performance
  - Session
- Callback from libev
  - Event based, f.ex. select or epoll
- pipeline.h, pipeline_perf.c, pipeline_session.c

Management

- Management of pool
- Unix Domain Socket
- Transfer socket descriptor from client process to main process
- Operations
  - Flush
  - Enable / disable database
  - Status
  - Graceful shutdown
- management.h, management.c

Features

- Connection pool
  - Trust, Password, MD5, SCRAM-SHA-256
    - Auth, Prefill, Pooling
- Limit connections for a database / user
  - Database / User
  - All / User
  - Database / All
  - All / All
- Transport Layer Security v1.2+

Features

- Prefill support
  - Create connections upon startup for a database / user pair
  - Requires a user vault
- Remove idle connections
  - After specified number of seconds
  - Off
- Perform connection validation
  - Foreground
  - Background
  - Off

Features

- Allow database access
  - Enable
  - Disable
- Shutdown
  - Gracefully
    - Cancel shutdown
  - Fast

Features

- Daemon mode
- User vault
  - Master key
  - AES-256
- Run-time administration tool
- Administration tool

Deployment

```
pgagroal 0.6.0
  High-performance connection pool for PostgreSQL

Usage:
  pgagroal [ -c CONFIG_FILE ] [ -a HBA_CONFIG_FILE ] [ -d ]

Options:
  -c, --config CONFIG_FILE      Set the path to the pgagroal.conf file
  -a, --hba HBA_CONFIG_FILE     Set the path to the pgagroal_hba.conf file
  -l, --limit LIMIT_CONFIG_FILE Set the path to the pgagroal_databases.conf file
  -u, --users USERS_FILE        Set the path to the pgagroal_users.conf file
  -d, --daemon                  Run as a daemon
  -V, --version                 Display version information
  -?, --help                    Display help
```

pgagroal.conf

```
[pgagroal]
host = *
port = 2345

log_type = file
log_level = info
log_path = /tmp/pgagroal.log

max_connections = 100
idle_timeout = 600
validation = off
unix_socket_dir = /tmp/.s.pgagroal

[primary]
host = localhost
port = 5432
```

pgagroal_hba.conf

```
#
# TYPE  DATABASE USER  ADDRESS      METHOD
#
host    alice    alice all          scram-sha-256
host    all      bob   10.0.0.0/16  md5
host    all      all   all          all
```

pgagroal_databases.conf

```
#
# DATABASE USER   MAX_SIZE INITIAL_SIZE MIN_SIZE
#
alice      alice  10       5            0
bob        bob    10       5            0
all        all    all
```

pgagroal_users.conf

```
# Create master key for the user vault
pgagroal-admin master-key

# At least 8 characters long
# Use at least 1 upper case letter (A, B, C, ...)
# Use at least 1 lower case letter (a, b, c, ...)
# Use at least 1 number (1, 2, 3, ...)
# Use at least 1 special character (!, @, #, ...)

# Add alice
pgagroal-admin -u pgagroal_users.conf add-user
User: alice
Password: alice

# Add bob too...
```

Lets go !

```
# Run in foreground

pgagroal -c pgagroal.conf -a pgagroal_hba.conf -l pgagroal_databases.conf -u pgagroal_users.conf


# Log file

05-28 13:00:00.000 32497 32497 I pgagroal.main pgagroal: started on localhost:2345


#  Connect

psql -h localhost -p 2345 -U alice
```

pgagroal-cli

```
Usage:
  pgagroal-cli [ -c CONFIG_FILE ] [ COMMAND ]

Options:
  -c, --config CONFIG_FILE Set the path to the pgagroal.conf file
  -V, --version            Display version information
  -?, --help               Display help

Commands:
  flush-idle               Flush idle connections
  flush-gracefully         Flush all connections gracefully
  flush-all                Flush all connections. USE WITH CAUTION !
  is-alive                 Is pgagroal alive
  enable                   Enable a database
  disable                  Disable a database
  gracefully               Stop pgagroal gracefully
  stop                     Stop pgagroal
  cancel-shutdown          Cancel the graceful shutdown
  status                   Status of pgagroal
  details                  Detailed status of pgagroal
```

pgagroal-admin

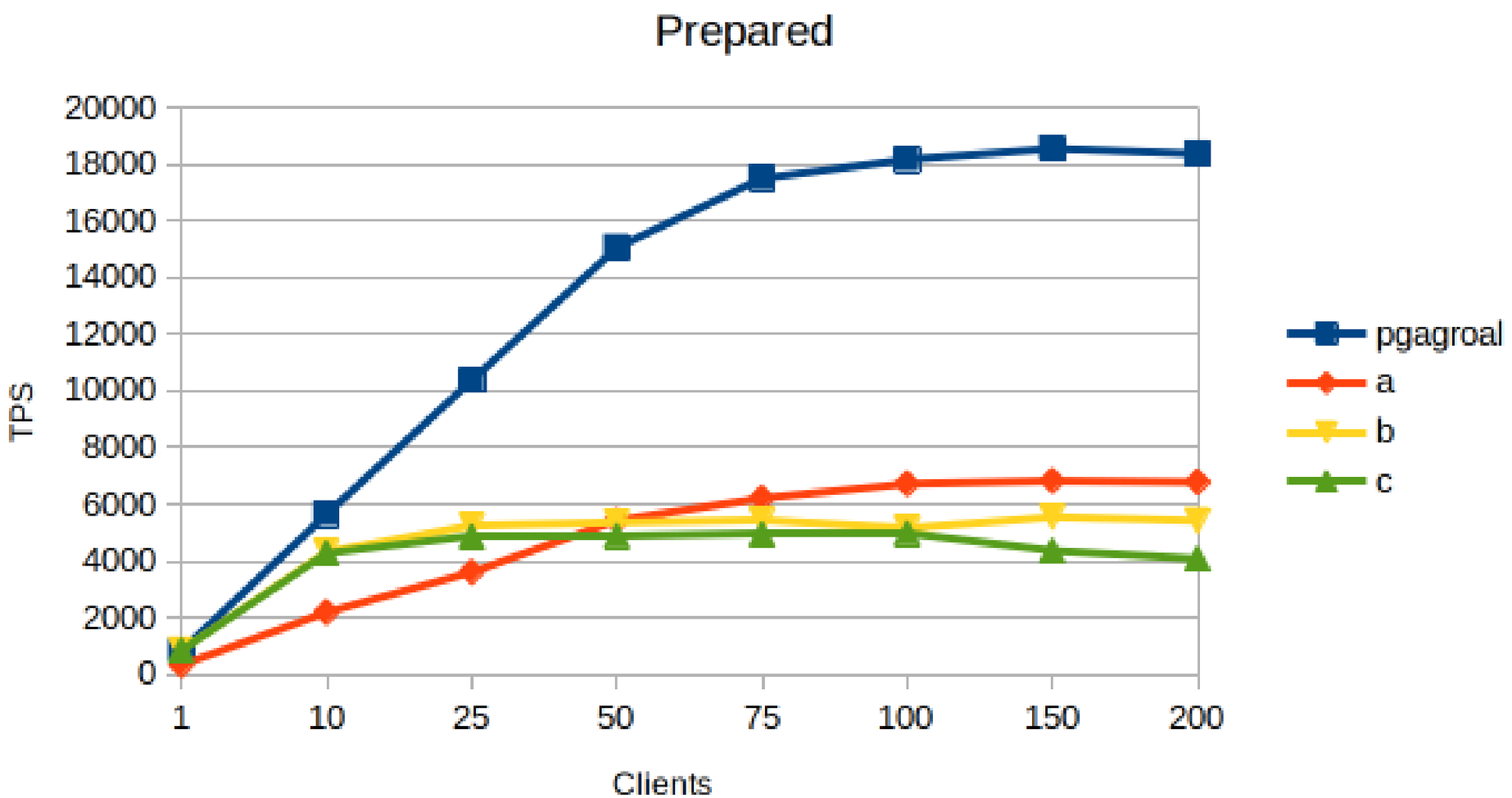
```
pgagroal-admin 0.6.0
  Administration utility for pgagroal

Usage:
  pgagroal-admin [ -u USERS_FILE ] [ COMMAND ]

Options:
  -u, --users USERS_FILE Set the path to the pgagroal_users.conf file
  -V, --version          Display version information
  -?, --help             Display help

Commands:
  master-key             Create or update the master key
  add-user               Add a user
  update-user            Update a user
  remove-user            Remove a user
  list-users             List all users
```

Performance run

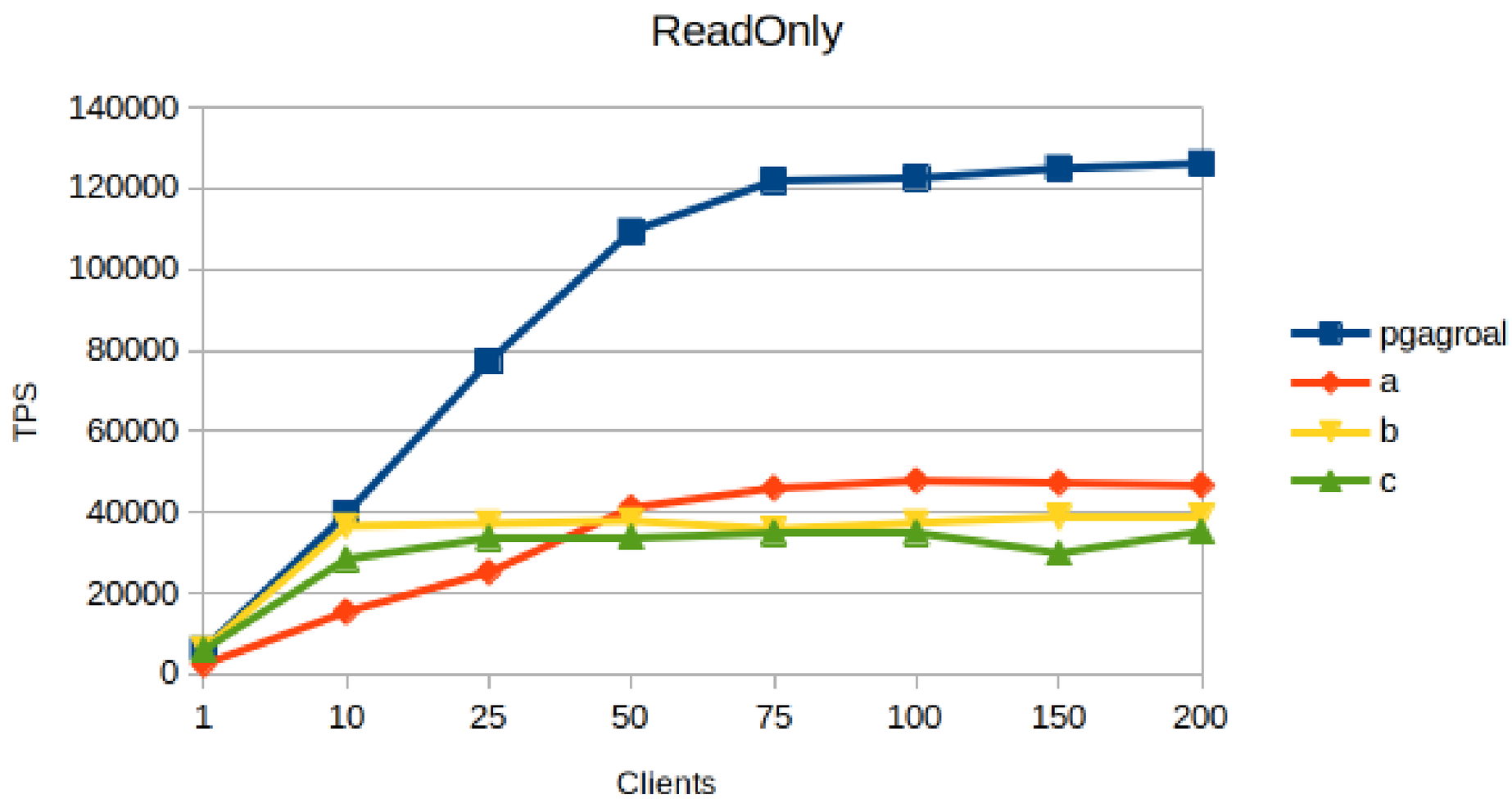
- Red Hat Enterprise Linux 7.7
  - 10G network
- pgagroal vs 3 other connection pools
  - We will call them ‘a’, ‘b’ and ‘c’
- Latest versions as of January 14, 2020
- All pools optimized for performance
- We will use pgbench for the tests

PLEASE
Run your own benchmarks !

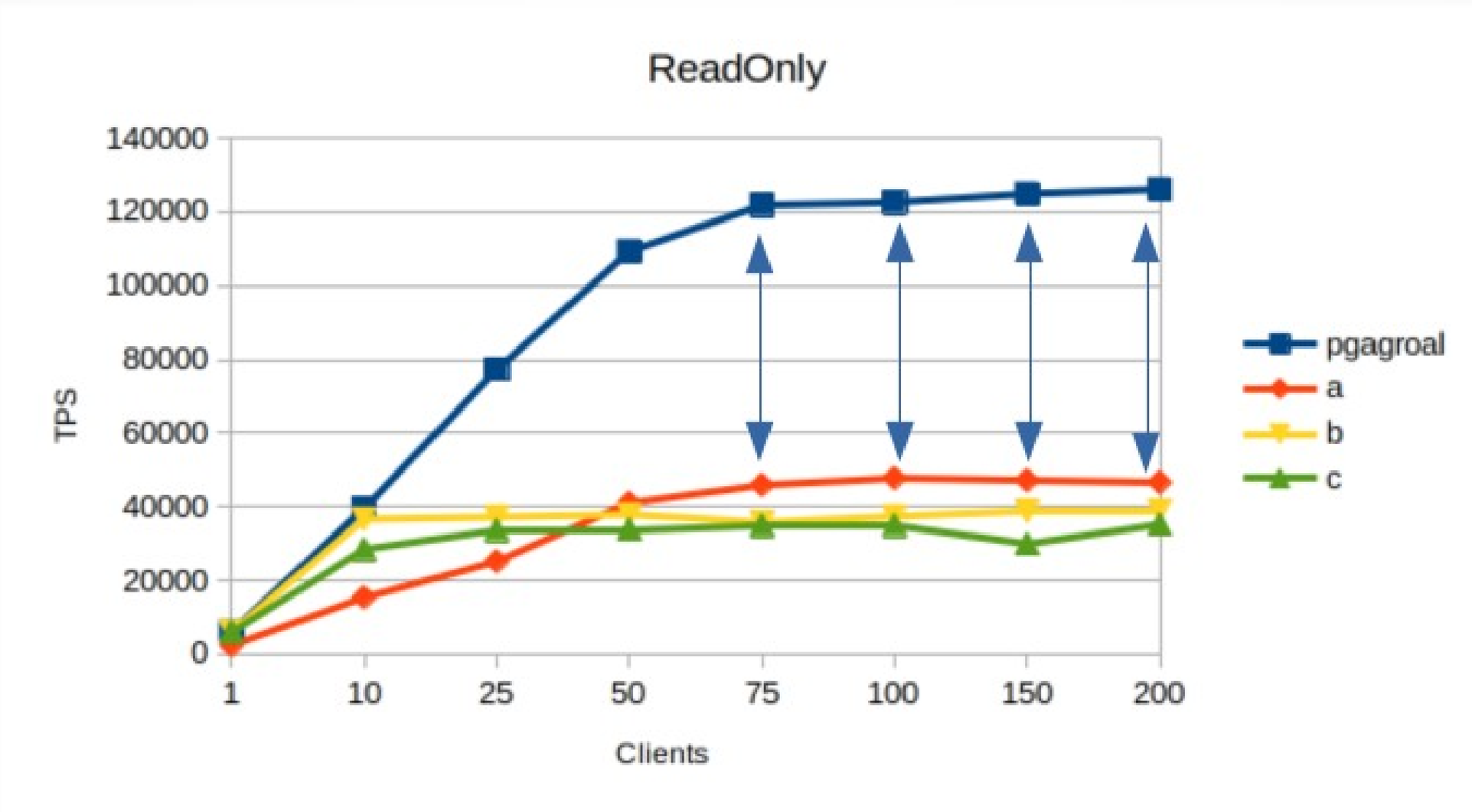
pgbench -M prepared



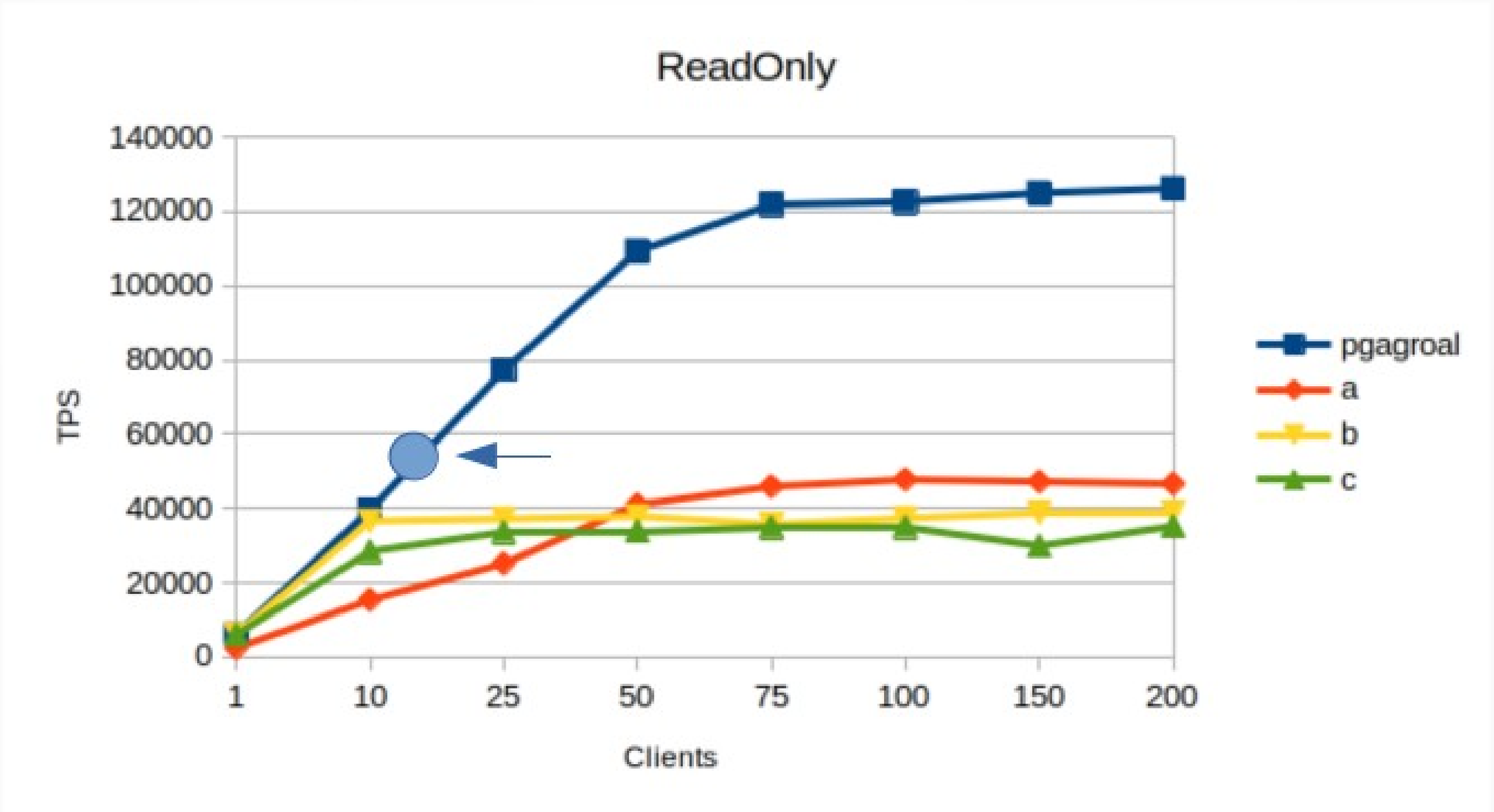
pgbench -S -M prepared



Important


ReadOnly
140000
120000
100000
80000
60000
40000
20000
0
TPS
1
10
25
50
75
100
150
200
Clients
pgagroal
a
b
c

Important


ReadOnly
140000
120000
100000
80000
60000
40000
20000
0
TPS
1
10
25
50
75
100
150
200
Clients
pgagroal
a
b
c

Performance

- RSS ~5Mb
- Connection overhead ~67kb
- “Zero” allocations at run-time
- Cloud friendly
- Future
  - libev/io_uring
  - Linux 5.6+

pgagroal 0.7.x

- Prometheus support
- Remote management

Roadmap

- Fail-over support
- High availability (HA)
- SELECTs on replicas
- Transaction pipeline
- Query cache

Closing thoughts

- Try out pgagroal on your own setup
  - Do your own benchmarks
- Star on GitHub
- Follow on Twitter
- Vote for features
- Contribute !

Thank you for your time !

https://agroal.github.io/pgagroal/

@pgagroal