Pulp Tasking System 2025

Updates on the Pulp Project’s kitchen

~ by Pedro Brochado

Foreword

- Thanks to the tasking working group!
  - Dennis
  - Brian
  - Decko
  - Matthias

Foreword

- Thanks to the tasking working group!
  - Dennis
  - Brian
  - Decko
  - Matthias
- Special thanks for Matthias for mentoring me in the tasking system and concurrency world

Overview

Overview

1. Background

Overview

1. Background
2. Cooking in Pulp’s kitchen

Overview

1. Background
2. Cooking in Pulp’s kitchen
3. New menu

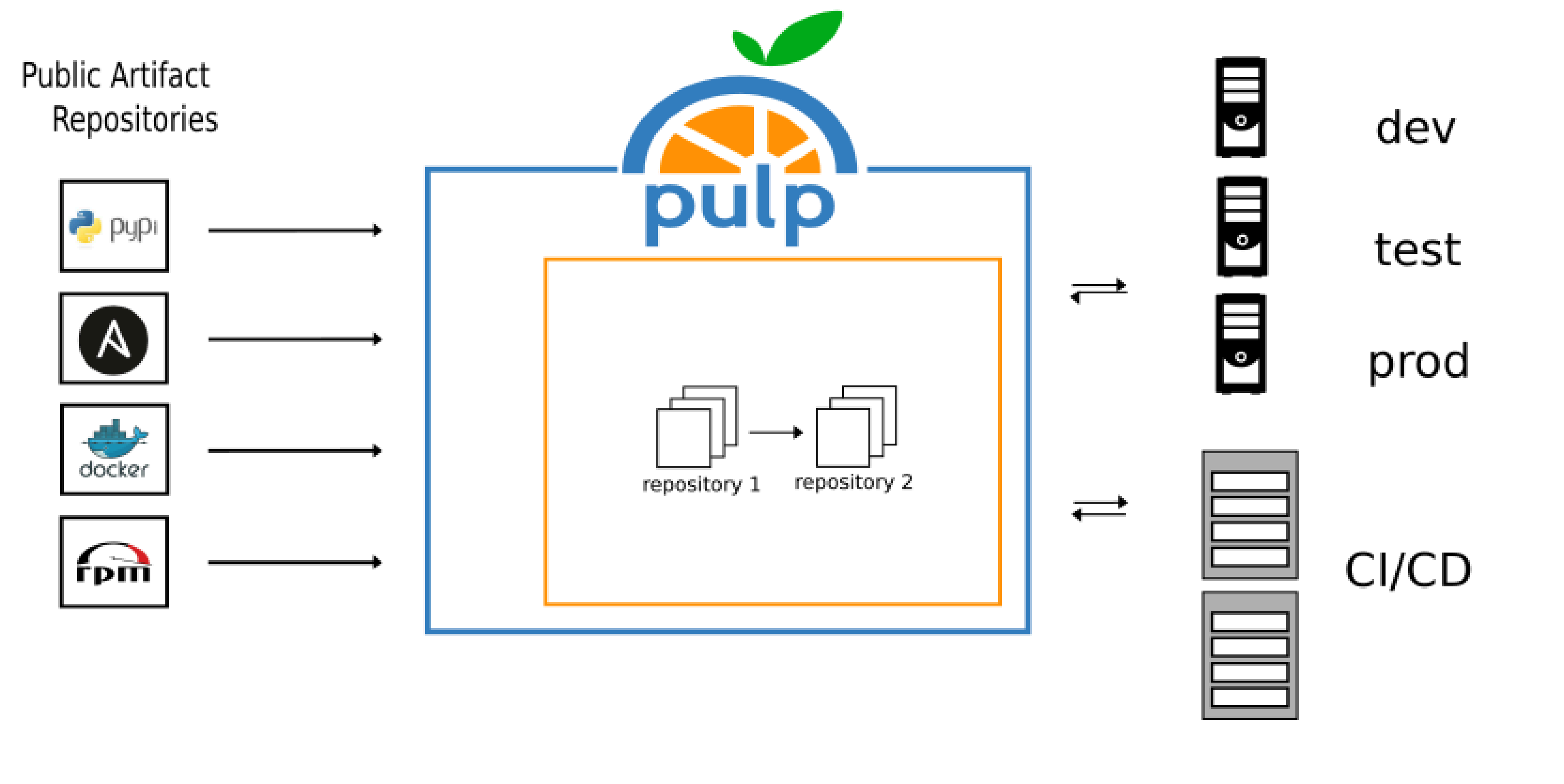
Overview

1. Background
2. Cooking in Pulp’s kitchen
3. New menu
4. Final words

1. Background

What is Pulp?

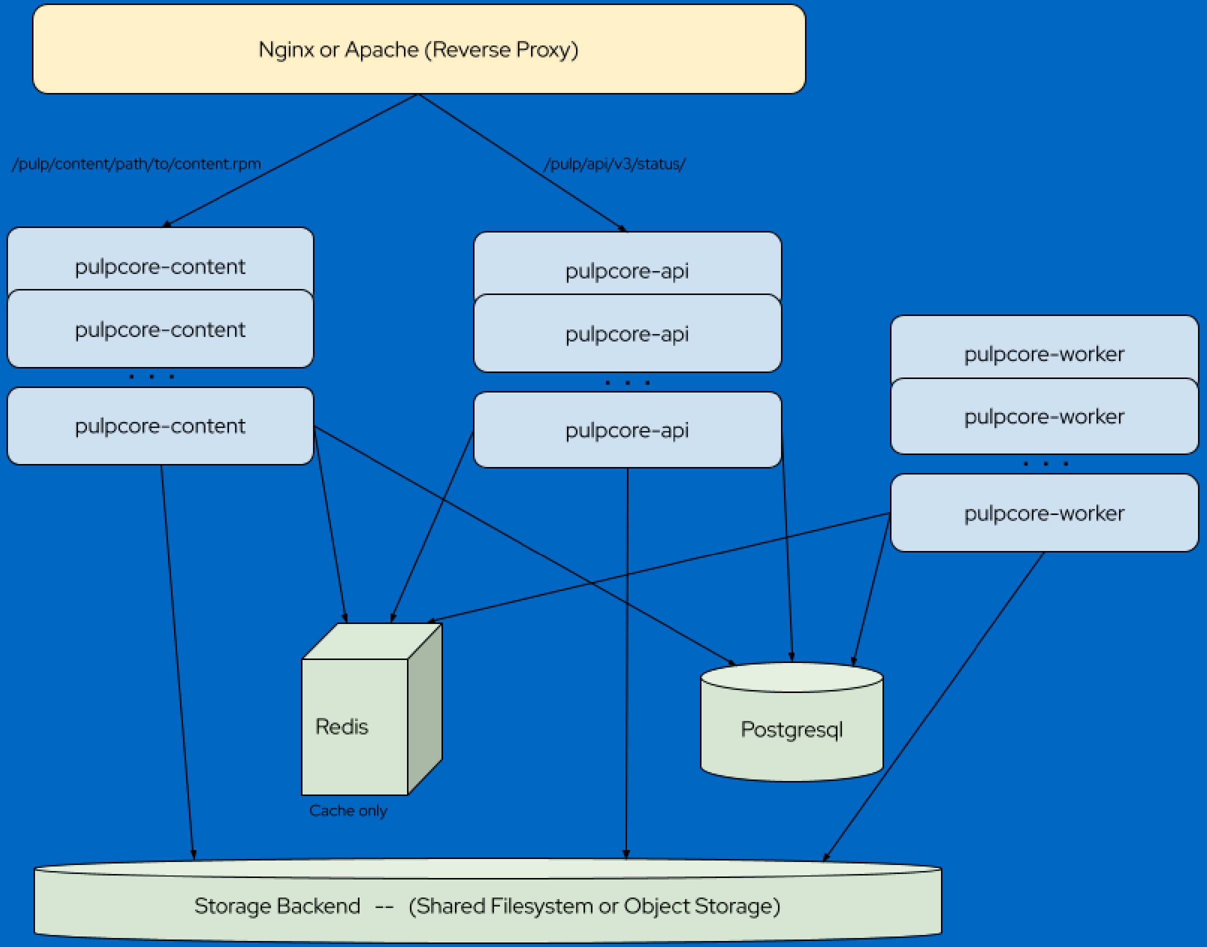
What is Pulp?


Public Artifact Repositories
pypi
docker
rpm
pulp
repository 1
repository 2
dev
test
prod
CI/CD

What is Pulp?

- An application for content management that can:
  - Serve content
  - Sync content from remote
  - Process uploaded content
- All these operations (and a lot more, of course) can trigger tasks!

Pulp’s architecture



Why a tasking system?

Why a tasking system?

Handle long running tasks

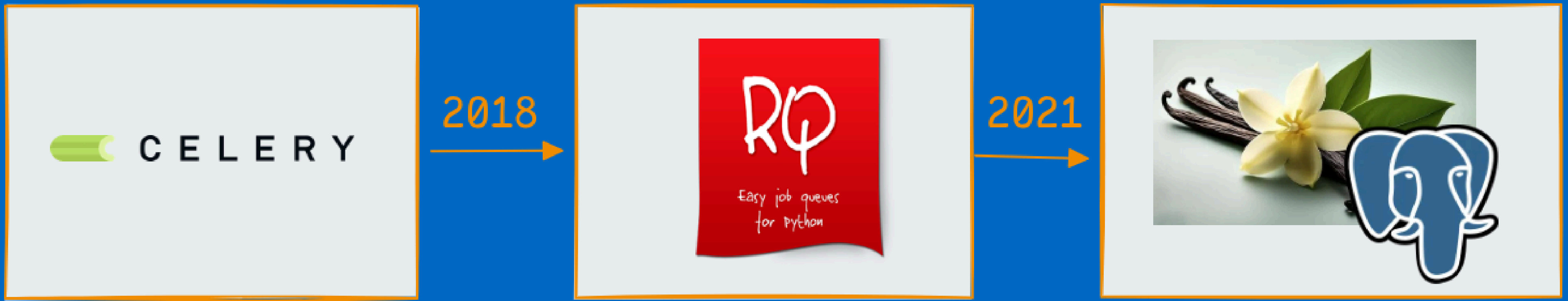
Why a tasking system?

Handle long running tasks

Serialize operations where concurrency is unsafe

Previous systems

Previous systems



(PULP 2)

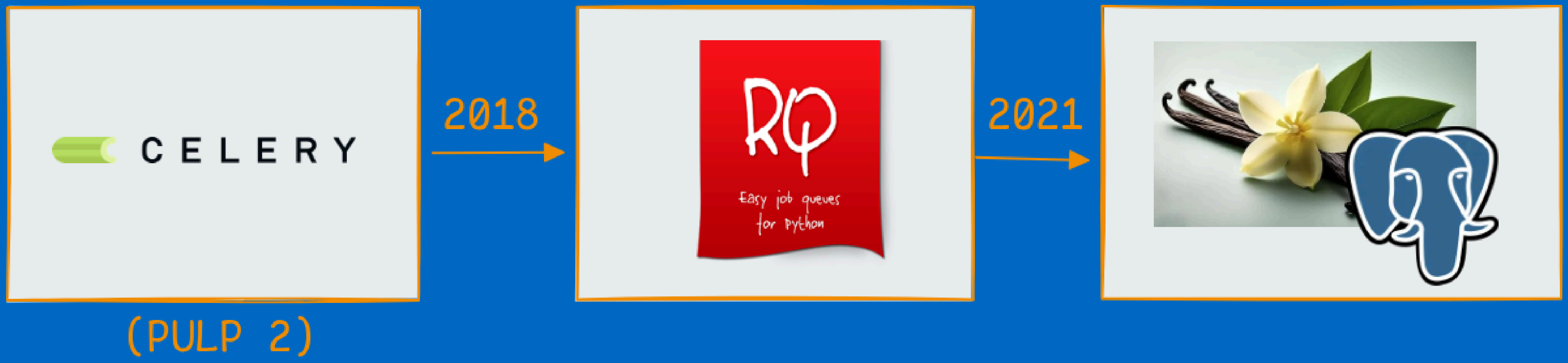
2018

2021

Previous systems

RQ pitfals:

- Resource manager bottleneck (for task processing)
- Problems with data duplication (redis and postgres)



2018

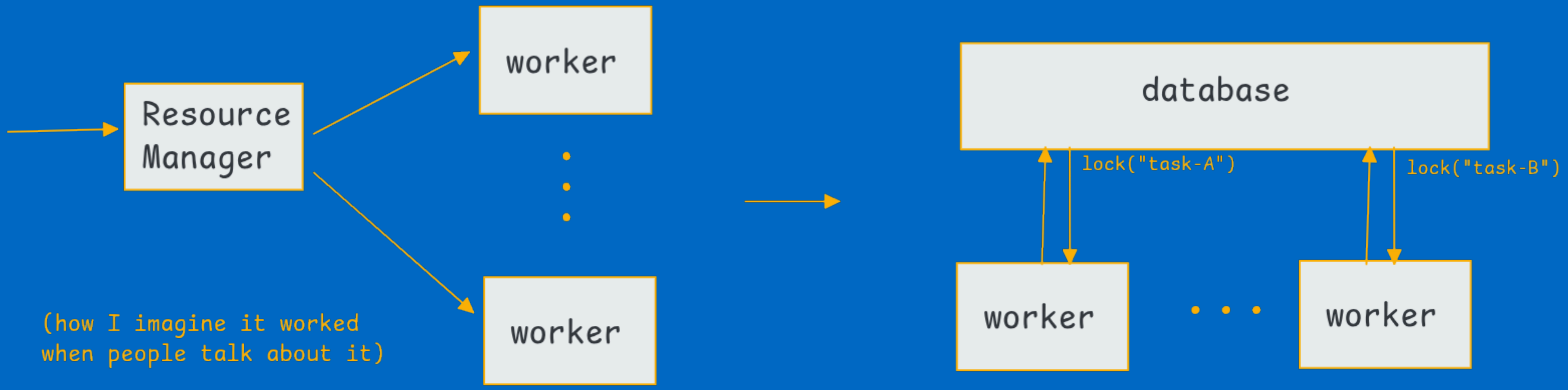
2021

(PULP 2)

System in 2021

System in 2021

1. Distributed task pulling



Resource Manager

worker

worker

(how I imagine it worked
when people talk about it)

database

lock("task-A")

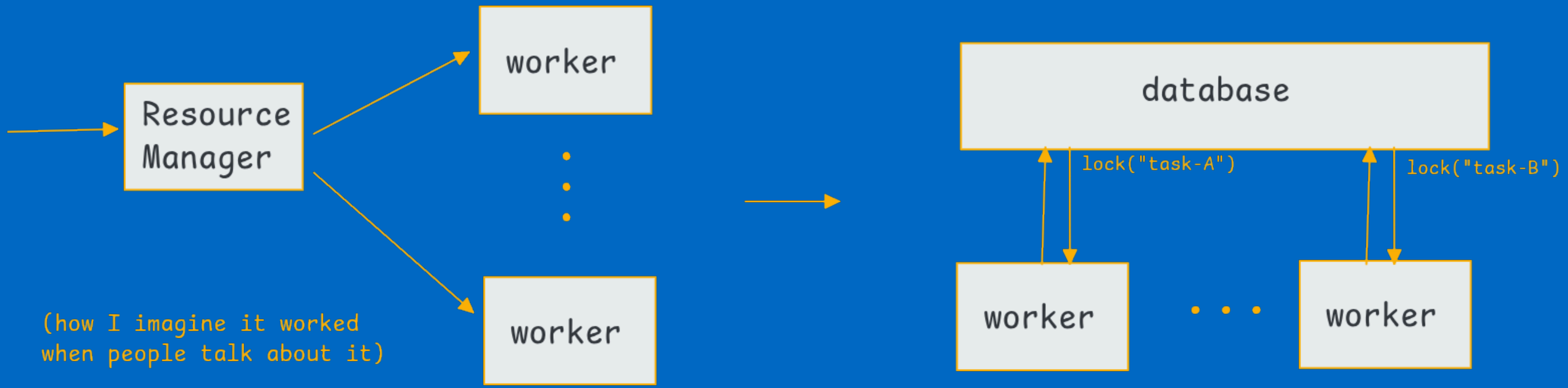
lock("task-B")

worker

worker

System in 2021

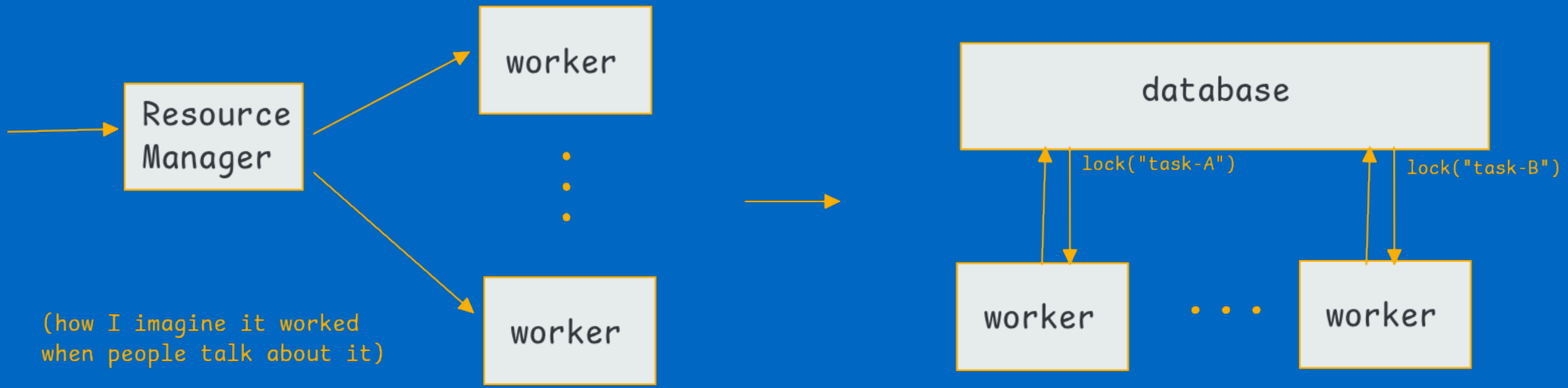
1. Distributed task pulling
2. Single source of truth for tasks (Postgres)



(how I imagine it worked when people talk about it)

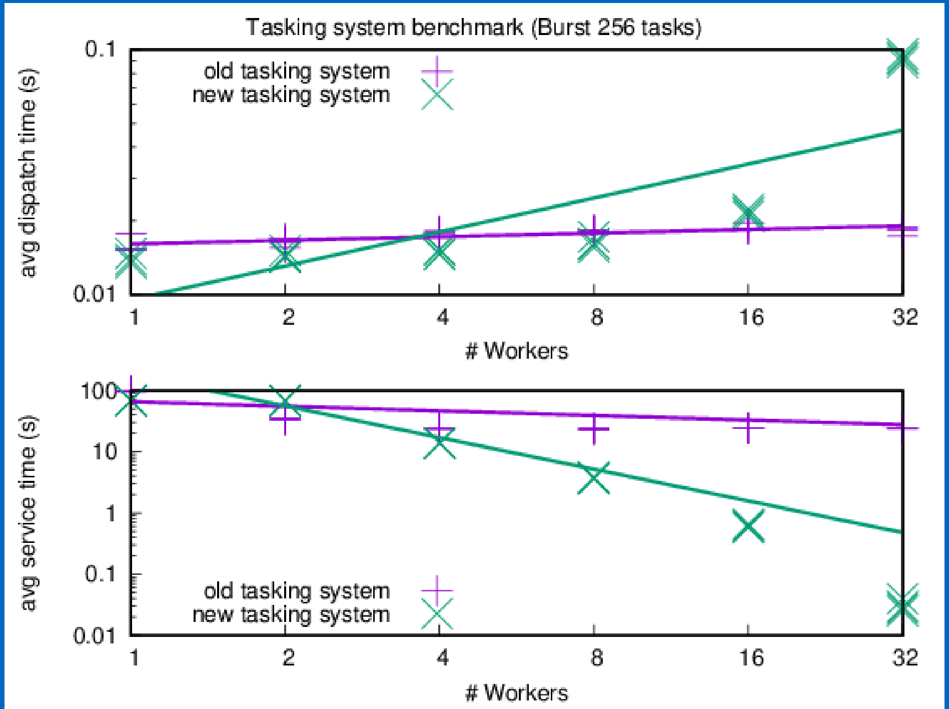
System in 2021

1. Distributed task pulling
2. Single source of truth for tasks (Postgres)
3. Auto resource management with PG Advisory Locks



(how I imagine it worked when people talk about it)

System in 2021



- Better scalability
- More robust against race-conditions and deadlocks

Recap and Questions

Recap and Questions

- What Pulp is, it’s architecture and operations that trigger tasks

Recap and Questions

- What Pulp is, it’s architecture and operations that trigger tasks
- Some motivations for Pulp’s tasking system

Recap and Questions

- What Pulp is, it's architecture and operations that trigger tasks
- Some motivations for Pulp's tasking system
- Some improvements over past tasking systems

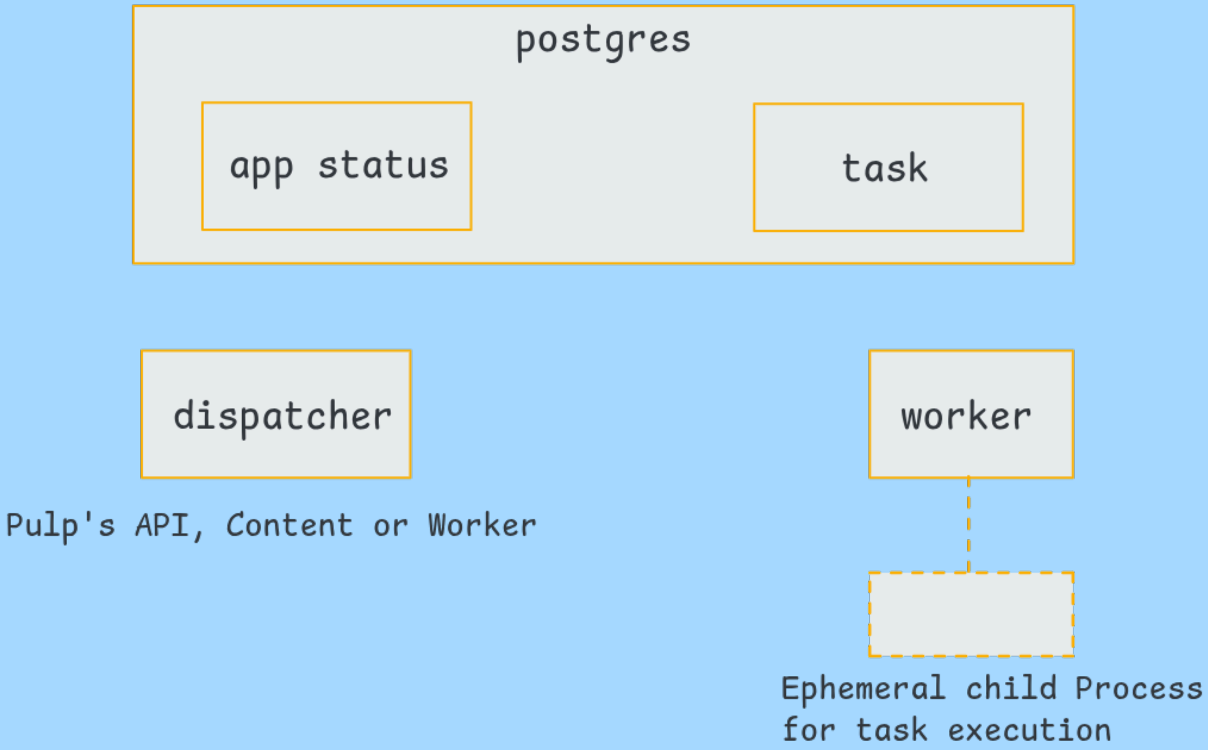
Questions? Thoughts?

Reference

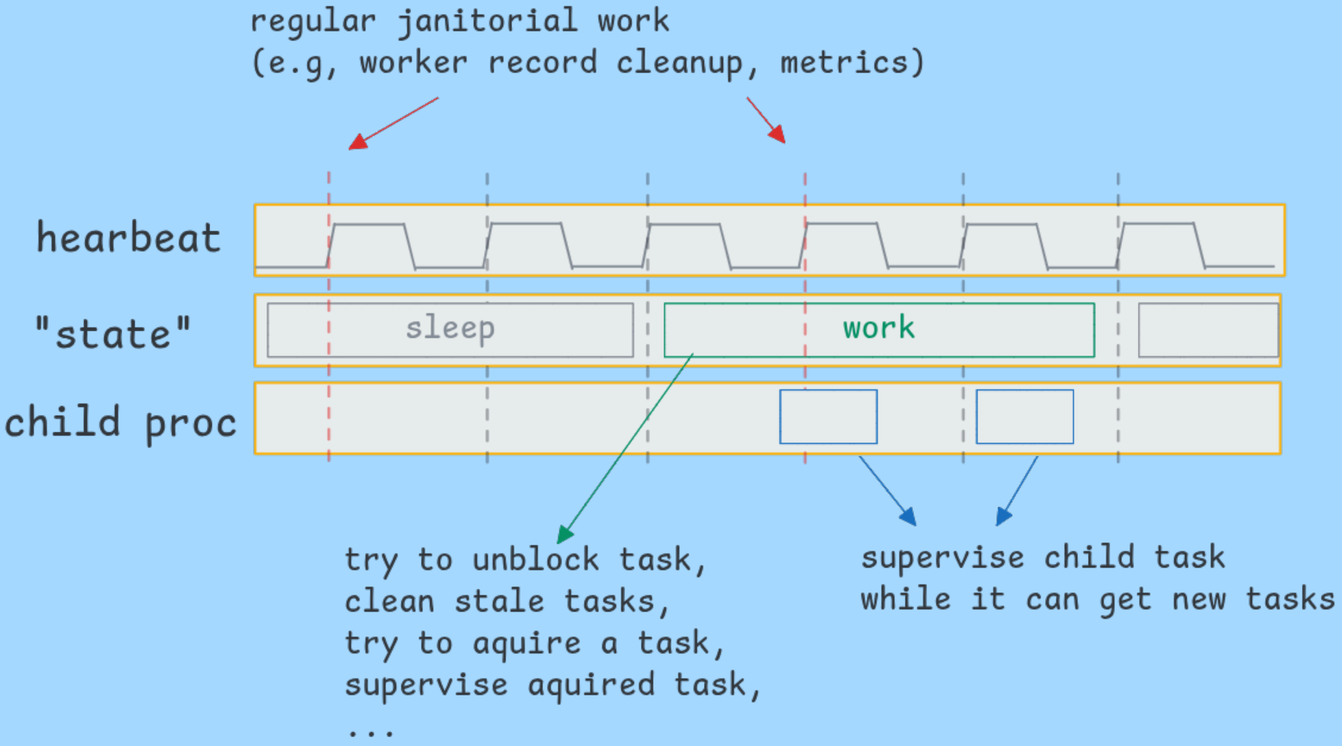
- Pulp’s architecture:
  - https://pulpproject.org/pulpcore/docs/admin/learn/architecture/
- From celery to RQ
  - https://pulpproject.org/blog/2021/06/21/a-new-tale---pulp-gets-queueless-distributed-task-workers/
- From RQ to vanilla
  - https://pulpproject.org/blog/2018/05/08/pulp-3-moving-to-rq-for-tasking/
  - https://www.youtube.com/watch?v=YWKw4RYluPM&pp=0gcJCSIKAYcqIYzv

2. Cooking in Pulp’s kitchen

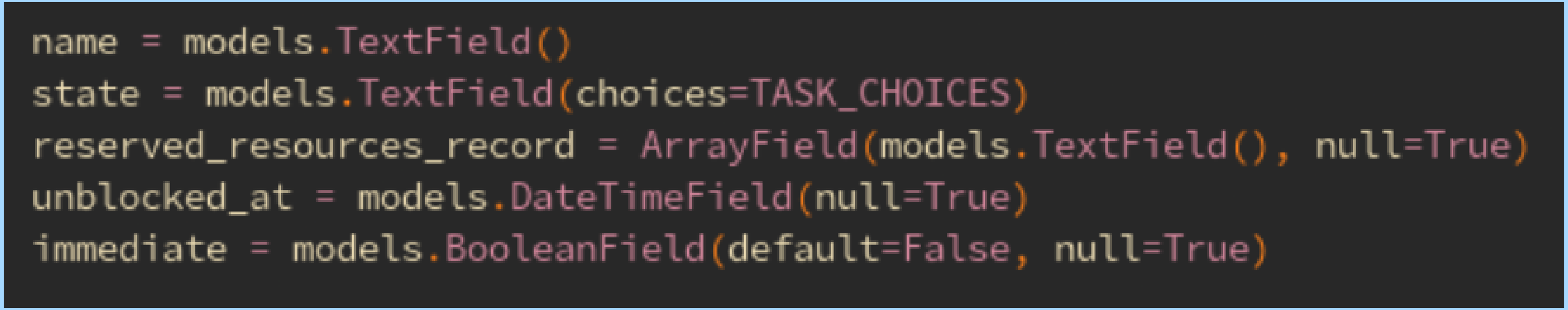
Components


postgres
app status
task
dispatcher
worker
Pulp's API, Content or Worker
Ephemeral child Process
for task execution

Workers

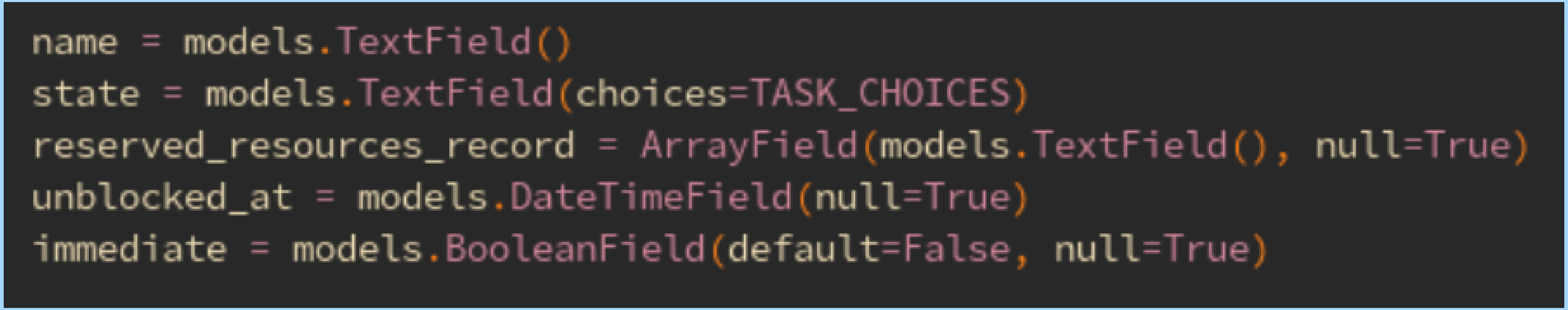

regular janitorial work
(e.g, worker record cleanup, metrics)
hearbeat
"state"
sleep
work
child proc
try to unblock task,
clean stale tasks,
try to aquire a task,
supervise aquired task,
...
supervise child task
while it can get new tasks

Tasks



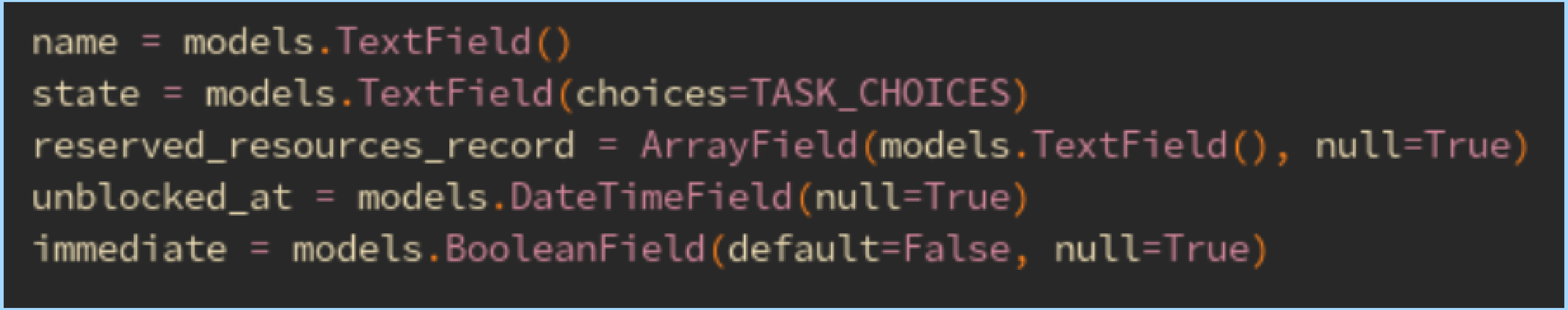
```
name = models.TextField()
state = models.TextField(choices=TASK_CHOICES)
reserved_resources_record = ArrayField(models.TextField(), null=True)
unblocked_at = models.DateTimeField(null=True)
immediate = models.BooleanField(default=False, null=True)
```

Tasks



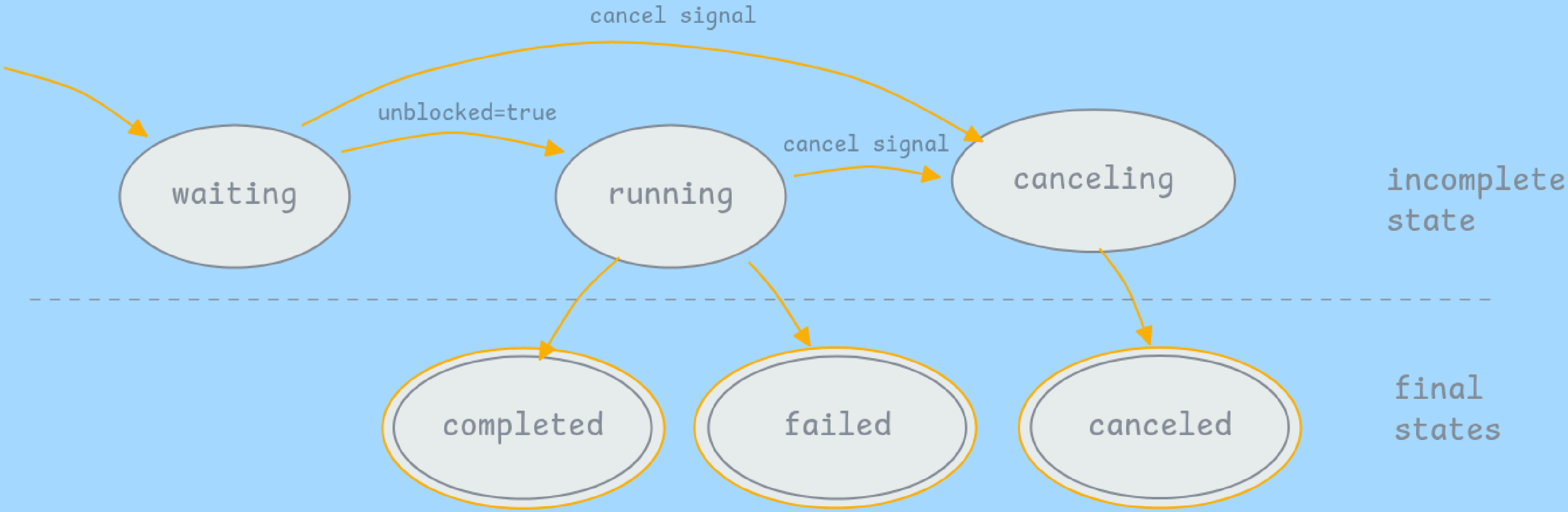
```
name = models.TextField()
state = models.TextField(choices=TASK_CHOICES)
reserved_resources_record = ArrayField(models.TextField(), null=True)
unblocked_at = models.DateTimeField(null=True)
immediate = models.BooleanField(default=False, null=True)
```

Tasks

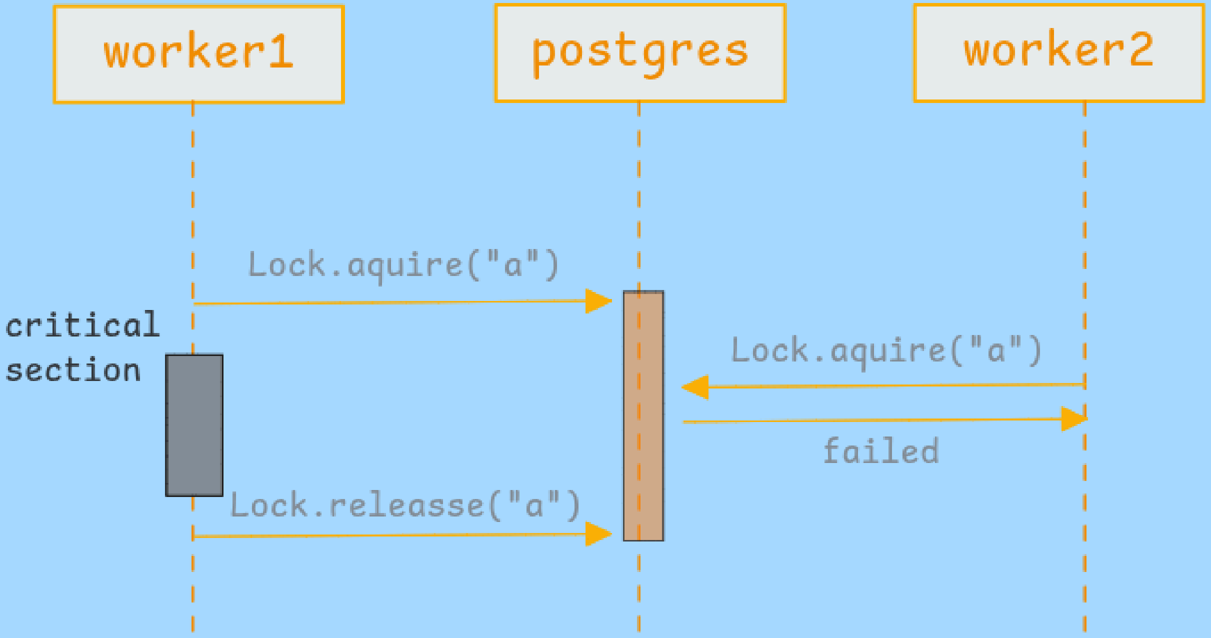


```
name = models.TextField()
state = models.TextField(choices=TASK_CHOICES)
reserved_resources_record = ArrayField(models.TextField(), null=True)
unblocked_at = models.DateTimeField(null=True)
immediate = models.BooleanField(default=False, null=True)
```

Tasks

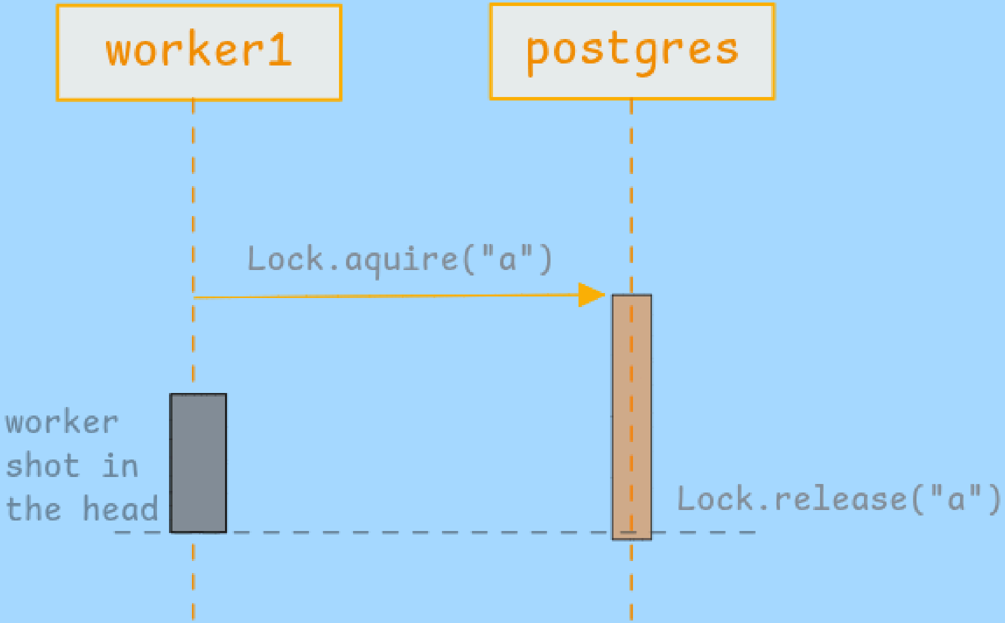

cancel signal
unblocked=true
cancel signal
waiting
running
canceling
incomplete state
completed
failed
canceled
final states

Locks



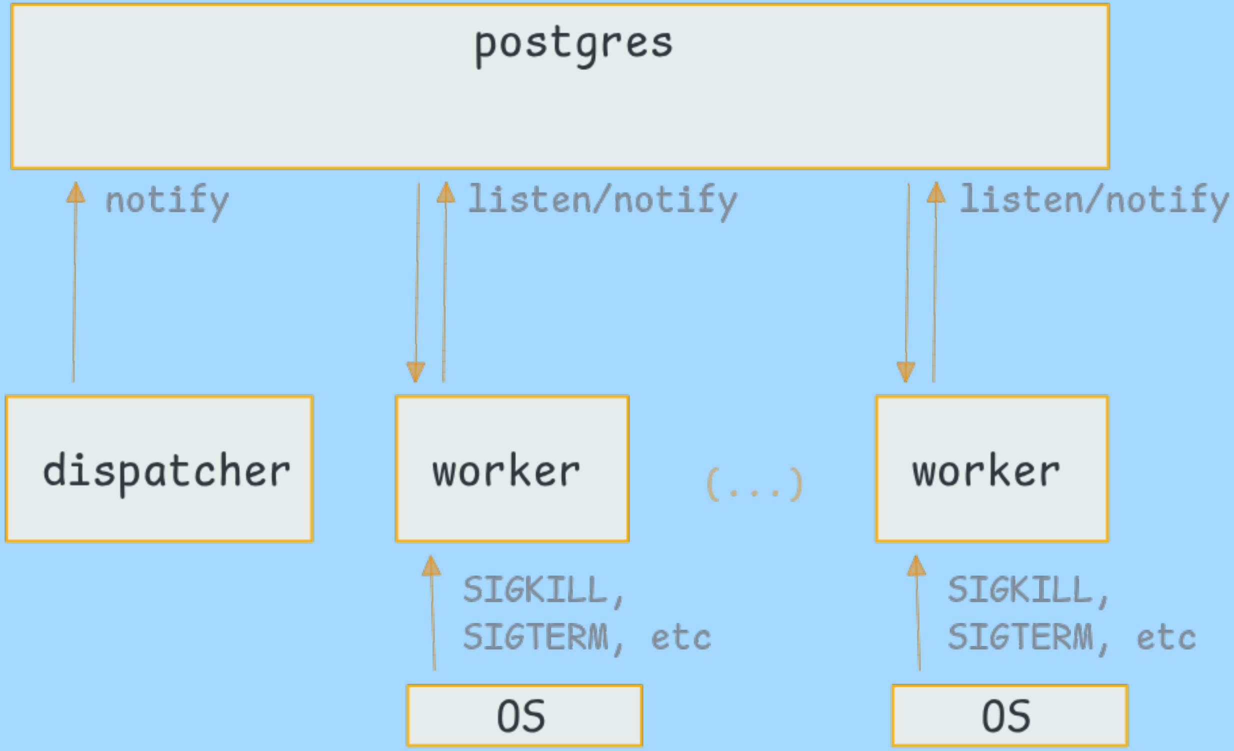
General Advisory Lock dynamic

Locks

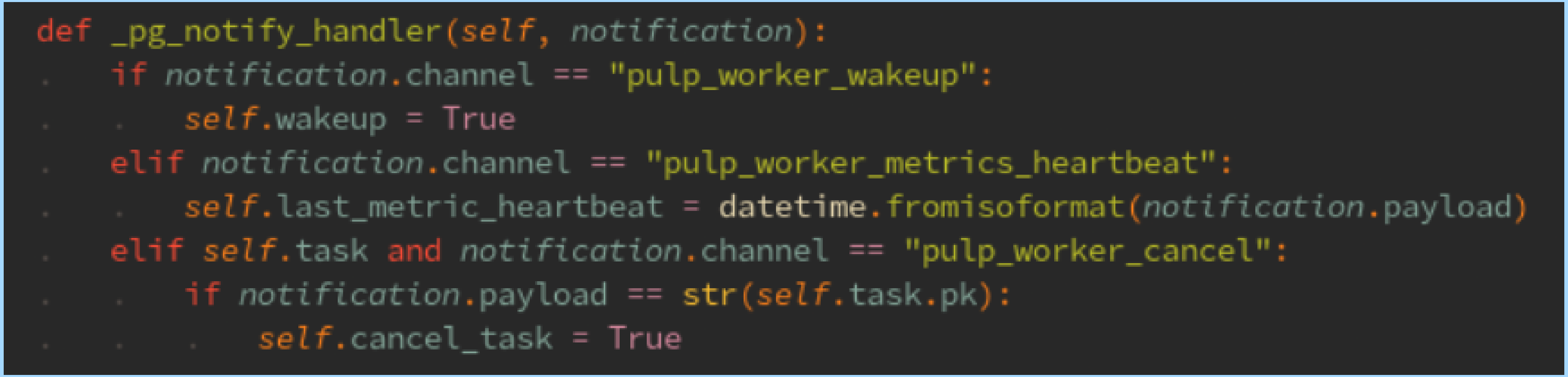


Session-based Advisory Locks

Signals


postgres
notify
listen/notify
listen/notify
dispatcher
worker
(...)
worker
SIGKILL,
SIGTERM, etc
SIGKILL,
SIGTERM, etc
OS
OS

Signals



```python
def _pg_notify_handler(self, notification):
    if notification.channel == "pulp_worker_wakeup":
        self.wakeup = True
    elif notification.channel == "pulp_worker_metrics_heartbeat":
        self.last_metric_heartbeat = datetime.fromisoformat(notification.payload)
    elif self.task and notification.channel == "pulp_worker_cancel":
        if notification.payload == str(self.task.pk):
            self.cancel_task = True
```

Recap and Questions

Recap and Questions

- Overview of tasking system components

Recap and Questions

- Overview of tasking system components
- Workers, tasks and locks

Recap and Questions

- Overview of tasking system components
- Workers, tasks and locks
- Inter-app communication with signals

Recap and Questions

- Overview of tasking system components
- Workers, tasks and locks
- Inter-app communication with signals

Questions? Thoughts?

Reference

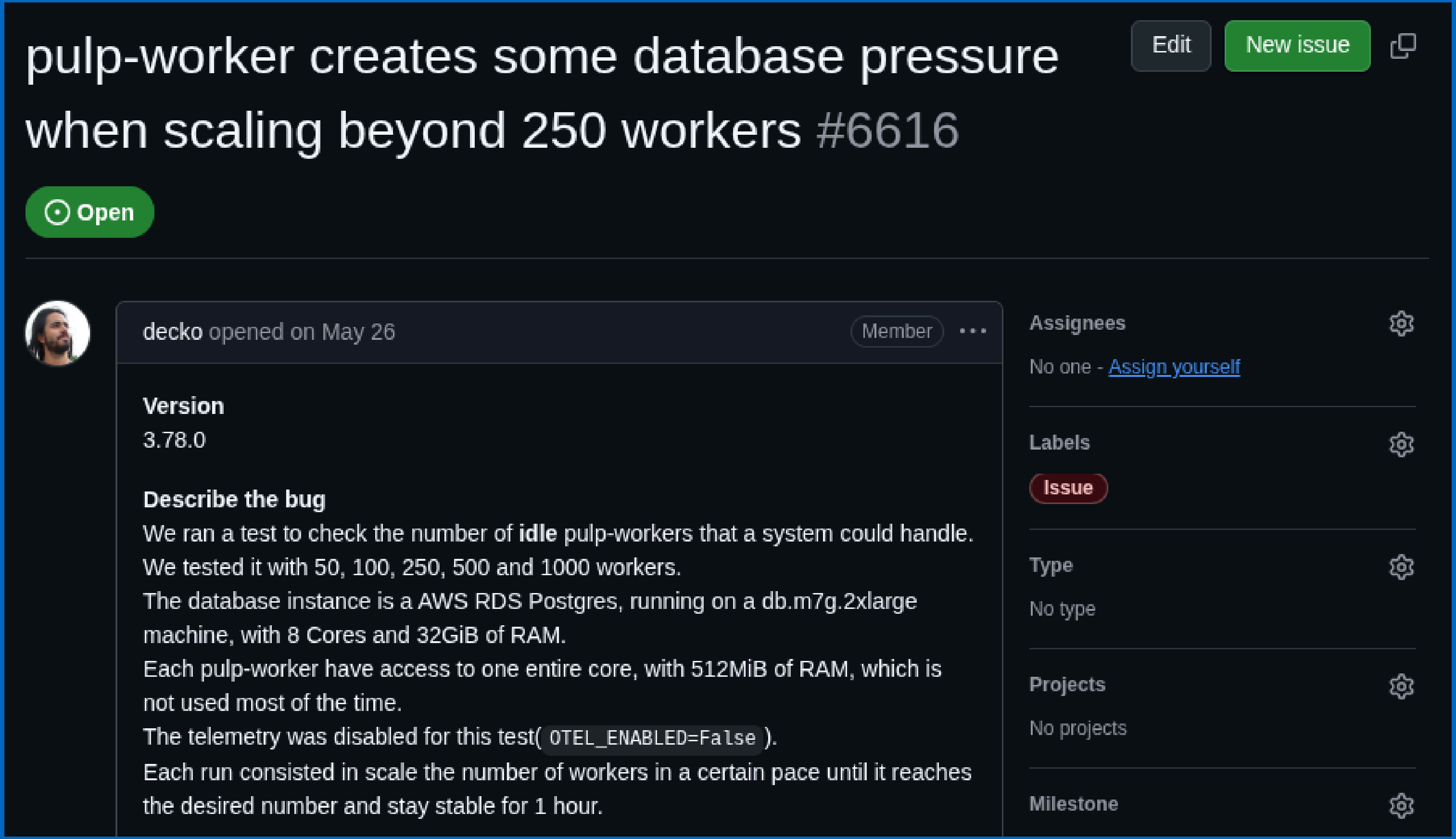
- Documentation:
  - https://pulpproject.org/pulpcore/docs/dev/learn/plugin-concepts/#tasks
- Pulpcore's codebase:
  - `pulpcore.app.models.task::Task`
  - `pulpcore.app.models.status`
  - `pulpcore.tasking.entrypoint`
  - `pulpcore.tasking.worker`
  - `pulpcore.tasking.tasking` (specially `dispatch`)
- Obs: the contents of this section are roughtly refering to `pulpcore<3.85`

3. New Menu

Requirments

- Pulp Services demand rapidly increasing. They need:
  - Throughput of 75-250k task/day (baseline 10k)
  - Connection pooling (Amazon's RDS Proxy)

Requirments



pulp-worker creates some database pressure when scaling beyond 250 workers #6616

decko opened on May 26

Version
3.78.0

Describe the bug
We ran a test to check the number of **idle** pulp-workers that a system could handle.
We tested it with 50, 100, 250, 500 and 1000 workers.
The database instance is a AWS RDS Postgres, running on a db.m7g.2xlarge machine, with 8 Cores and 32GiB of RAM.
Each pulp-worker have access to one entire core, with 512MiB of RAM, which is not used most of the time.
The telemetry was disabled for this test(`OTEL_ENABLED=False`).
Each run consisted in scale the number of workers in a certain pace until it reaches the desired number and stay stable for 1 hour.

Assignees
No one - Assign yourself

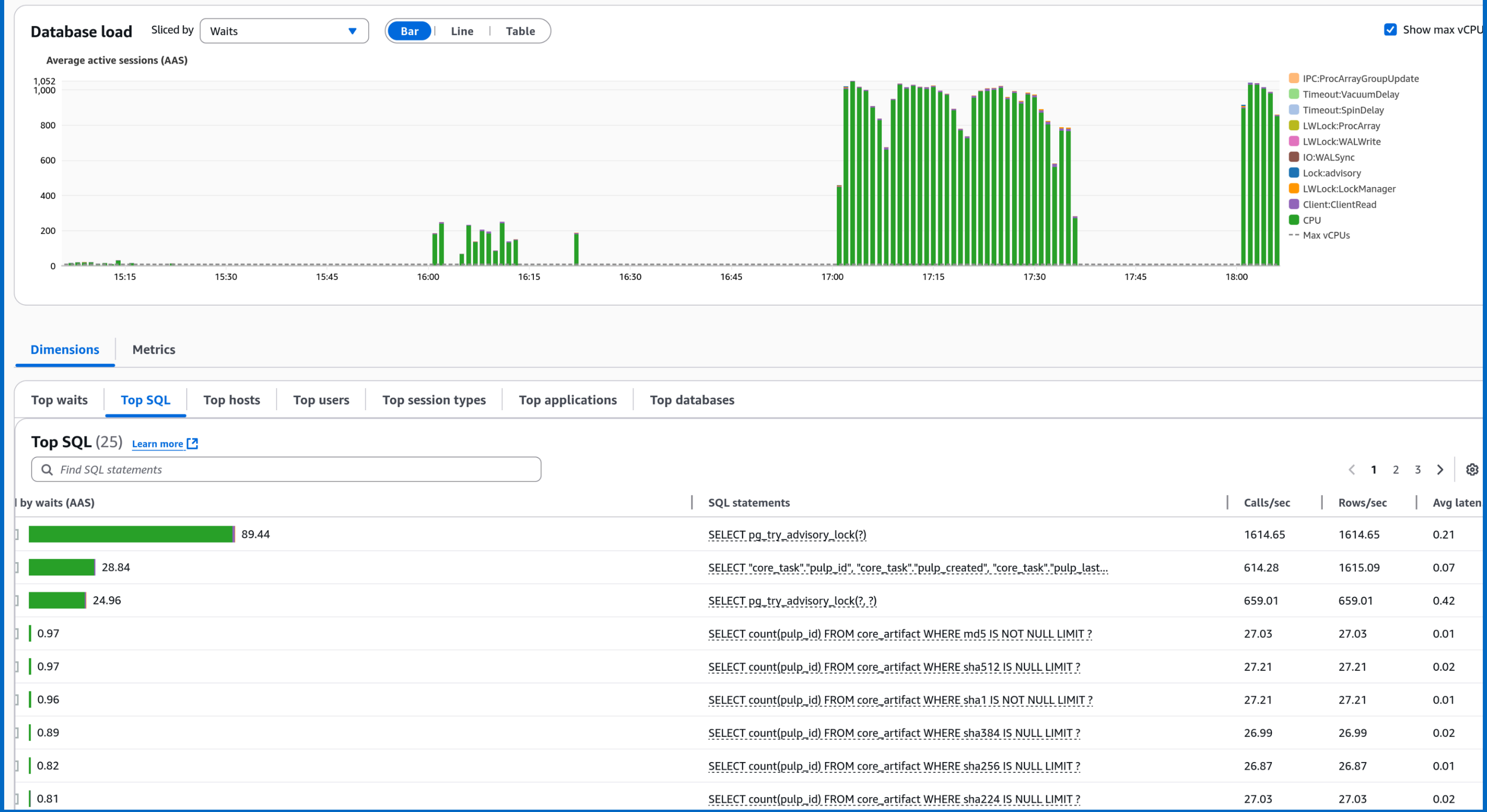
Labels
Issue

Type
No type

Projects
No projects

Milestone

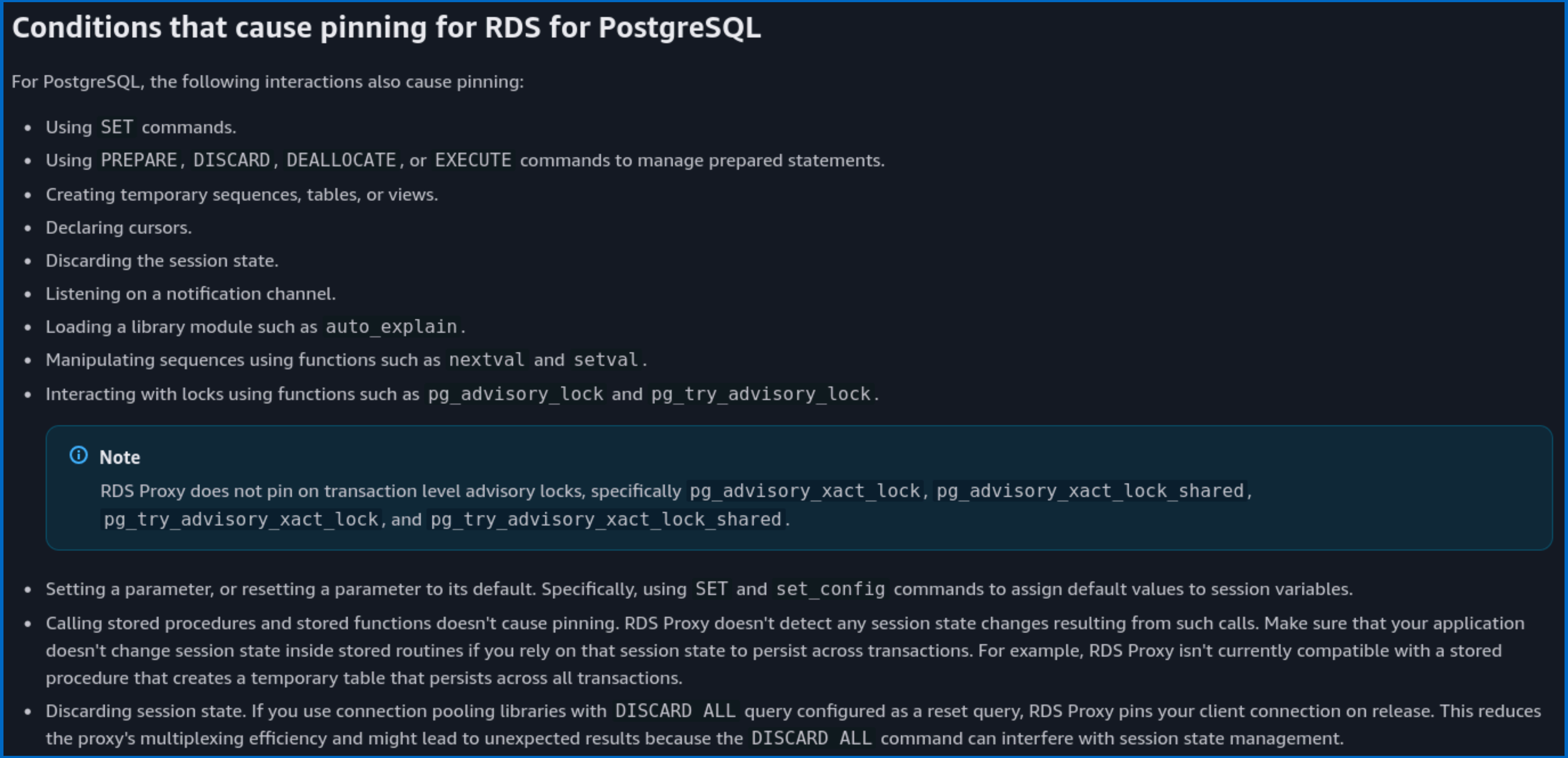
Requirments



Top SQL (25)

by waits (AAS)	SQL statements	Calls/sec	Rows/sec	Avg laten
89.44	SELECT pg_try_advisory_lock(?)	1614.65	1614.65	0.21
28.84	SELECT "core_task"."pulp_id", "core_task"."pulp_created", "core_task"."pulp_last...	614.28	1615.09	0.07
24.96	SELECT pg_try_advisory_lock(?, ?)	659.01	659.01	0.42
0.97	SELECT count(pulp_id) FROM core_artifact WHERE md5 IS NOT NULL LIMIT ?	27.03	27.03	0.01
0.97	SELECT count(pulp_id) FROM core_artifact WHERE sha512 IS NULL LIMIT ?	27.21	27.21	0.02
0.96	SELECT count(pulp_id) FROM core_artifact WHERE sha1 IS NOT NULL LIMIT ?	27.21	27.21	0.01
0.89	SELECT count(pulp_id) FROM core_artifact WHERE sha384 IS NULL LIMIT ?	26.99	26.99	0.02
0.82	SELECT count(pulp_id) FROM core_artifact WHERE sha256 IS NULL LIMIT ?	26.87	26.87	0.01
0.81	SELECT count(pulp_id) FROM core_artifact WHERE sha224 IS NULL LIMIT ?	27.03	27.03	0.02

Requirments



Conditions that cause pinning for RDS for PostgreSQL

For PostgreSQL, the following interactions also cause pinning:

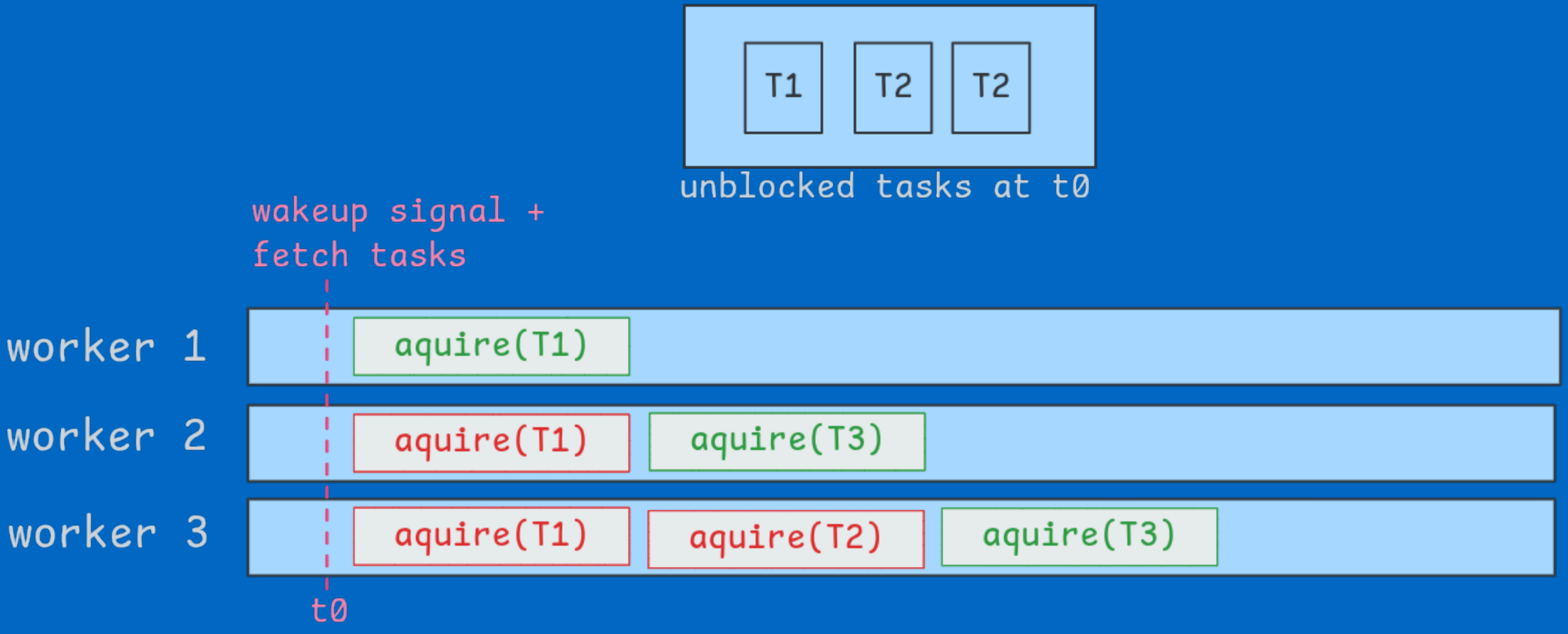
- Using `SET` commands.
- Using `PREPARE`, `DISCARD`, `DEALLOCATE`, or `EXECUTE` commands to manage prepared statements.
- Creating temporary sequences, tables, or views.
- Declaring cursors.
- Discarding the session state.
- Listening on a notification channel.
- Loading a library module such as `auto_explain`.
- Manipulating sequences using functions such as `nextval` and `setval`.
- Interacting with locks using functions such as `pg_advisory_lock` and `pg_try_advisory_lock`.

> **Note**
>
> RDS Proxy does not pin on transaction level advisory locks, specifically `pg_advisory_xact_lock`, `pg_advisory_xact_lock_shared`, `pg_try_advisory_xact_lock`, and `pg_try_advisory_xact_lock_shared`.

- Setting a parameter, or resetting a parameter to its default. Specifically, using `SET` and `set_config` commands to assign default values to session variables.
- Calling stored procedures and stored functions doesn't cause pinning. RDS Proxy doesn't detect any session state changes resulting from such calls. Make sure that your application doesn't change session state inside stored routines if you rely on that session state to persist across transactions. For example, RDS Proxy isn't currently compatible with a stored procedure that creates a temporary table that persists across all transactions.
- Discarding session state. If you use connection pooling libraries with `DISCARD ALL` query configured as a reset query, RDS Proxy pins your client connection on release. This reduces the proxy's multiplexing efficiency and might lead to unexpected results because the `DISCARD ALL` command can interfere with session state management.

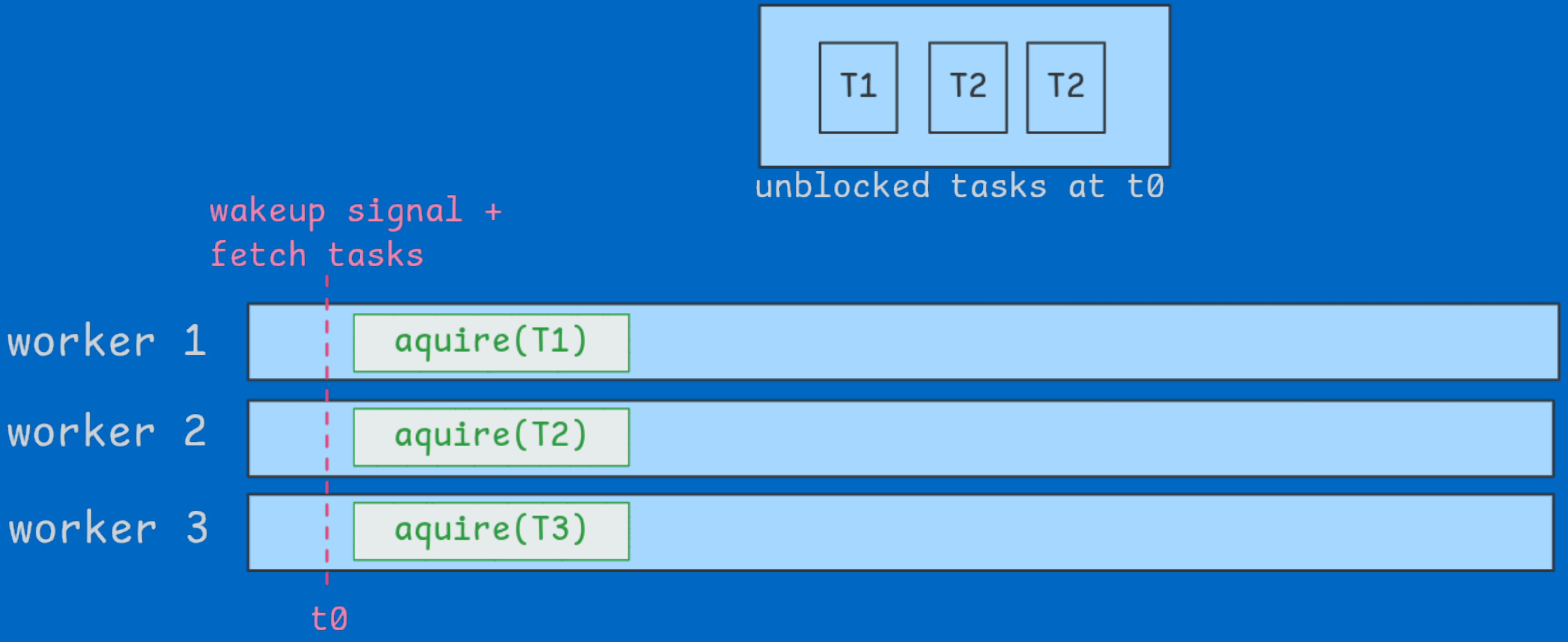
Changes: Worker's overhead

- Randomization on acquiring task



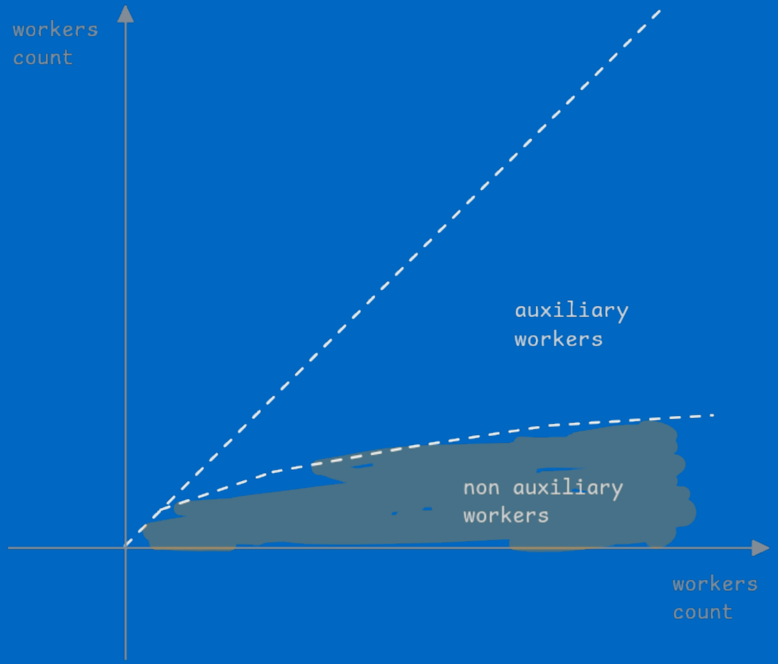
Changes: Worker’s overhead

- Randomization on acquiring task



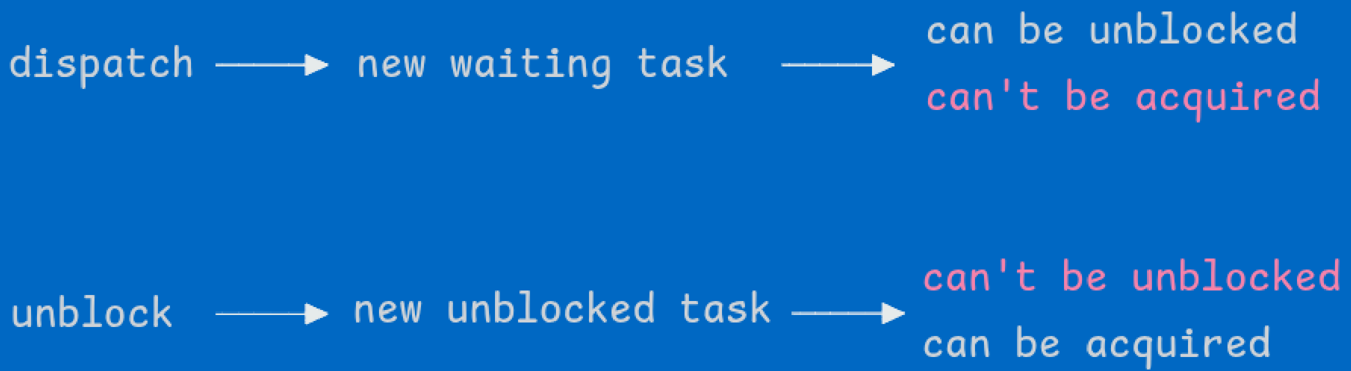
Changes: Worker's overhead

- Auxiliary workers: only supervise tasks



Changes: Worker’s overhead

- Wakeup unblock/handle signals

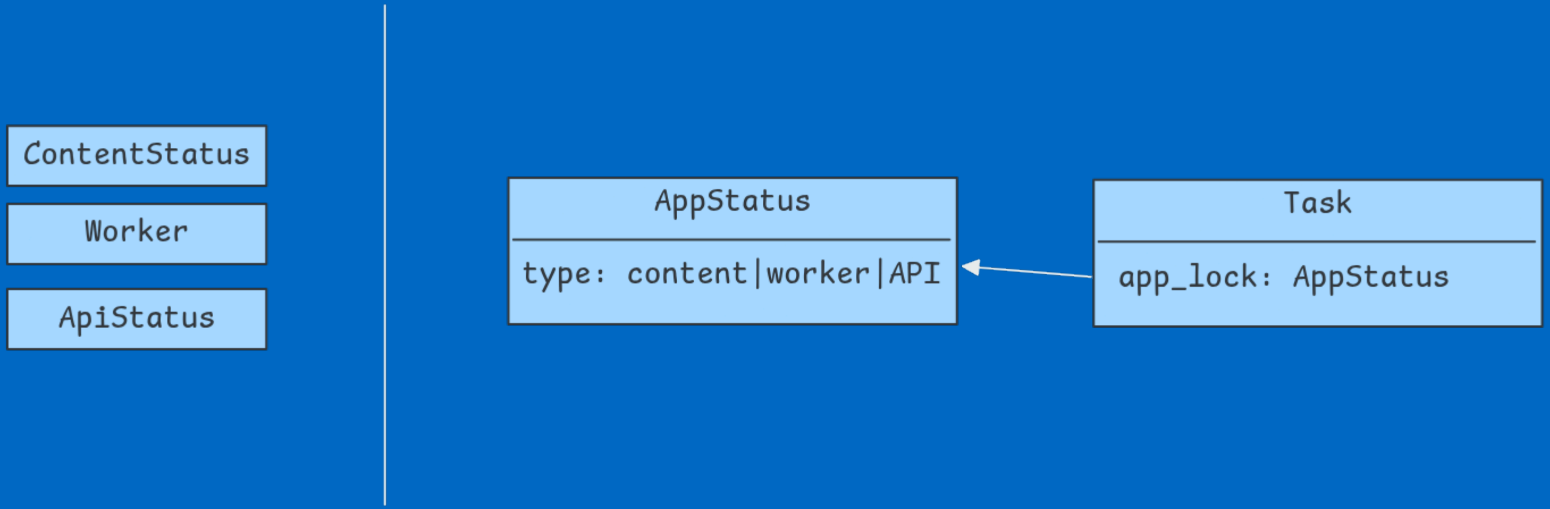


dispatch → new waiting task → can be unblocked / can't be acquired

unblock → new unblocked task → can't be unblocked / can be acquired

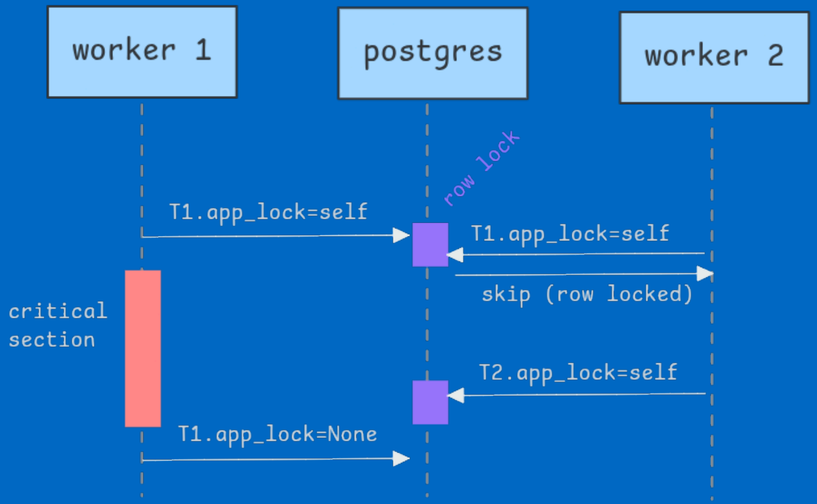
Changes: RDS Proxy support

- App Status consolidation
  - Required to build `app_lock` foreign key to any pulp app



Changes: RDS Proxy support

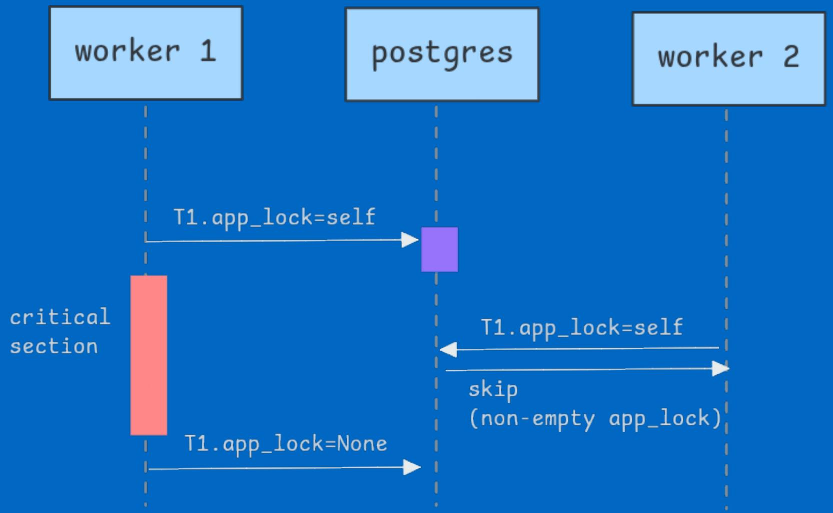
- Custom PG update query to replace advisory locks



skip locked

Changes: RDS Proxy support

- Remove session-based advisory locks



skip non-empty field

Recap and Questions

Recap and Questions

- Requirements from Pulp Services installation

Recap and Questions

- Requirements from Pulp Services installation
- Changes related to worker’s overhead

Recap and Questions

- Requirements from Pulp Services installation
- Changes related to worker's overhead
- Changes related to providing support for connection pooling

Recap and Questions

- Requirements from Pulp Services installation
- Changes related to worker’s overhead
- Changes related to providing support for connection pooling

Questions? Thoughts?

Reference

- Pulp Services' Jira
  - https://issues.redhat.com/browse/PULP-674
- Decko's issue with important results
  - https://github.com/pulp/pulpcore/issues/661
- See workers code for `pulpcore>3.85`

4. Final words

Conclusion

Conclusion

- This is *some* of the actual work done so far
  - ommited a lot of disussions, ideas, etc to fit time

Conclusion

- This is *some* of the actual work done so far
  - ommited a lot of disussions, ideas, etc to fit time
- Personal thought
  - lack of upstream instrumentation for understanding key metrics of the system

Conclusion

- This is *some* of the actual work done so far
  - ommited a lot of disussions, ideas, etc to fit time
- Personal thought
  - lack of upstream instrumentation for understanding key metrics of the system
- Kudos again to the tasking working group!

Conclusion

- This is *some* of the actual work done so far
  - ommited a lot of disussions, ideas, etc to fit time
- Personal thought
  - lack of upstream instrumentation for understanding key metrics of the system
- Kudos again to the tasking working group!

THE END