dbi services
by Sequotech

The cost of security debt in PostgreSQL when implementing AI workflows

ABOUT ME

Adrien OBERNESSER

Consultant DBA / data engineer / Observability engineer / AI Harness engineer / context engineer / etc…

Open Infrastructure - Nyon (VD)

Contact me

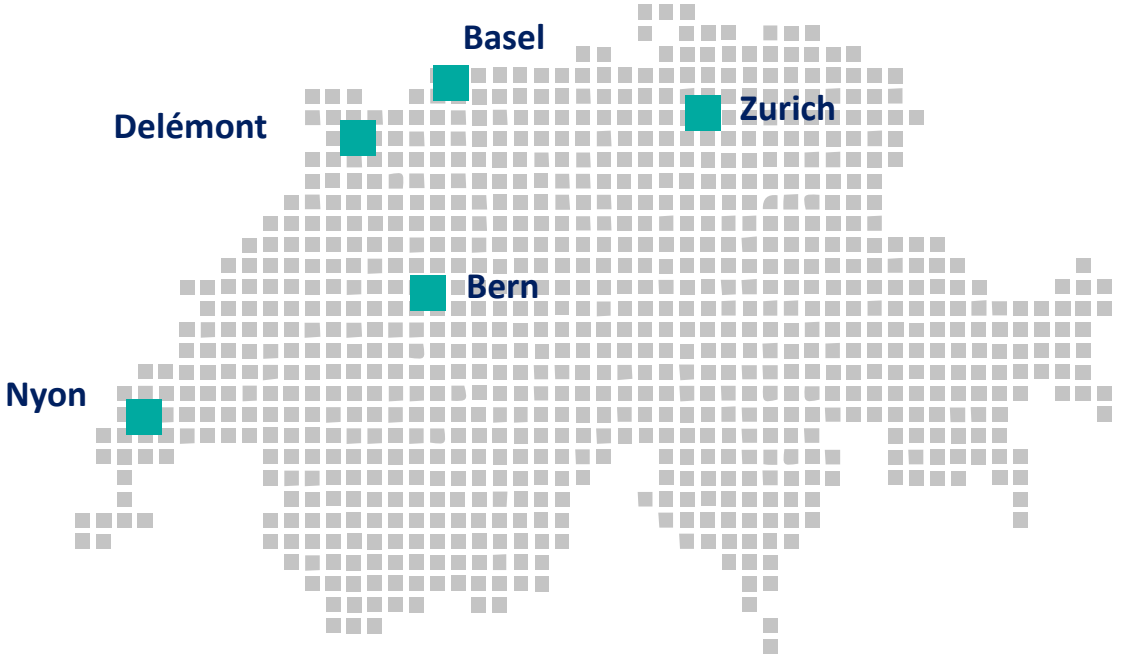
adrien.obernesser[at]dbi-services.com

PostgreSQL Observability Summit

WHO WE ARE

The company

- Founded in 2010
- 80 employees
- Specialized in data infrastructure and platforms
- Customers in Switzerland and all over Europe



Our offer

- Consulting
- Service Level Agreements
- Trainings
- License Management
- Products

A personal note

Three years building AI workflows on PostgreSQL

«Observability is a precondition to AI»

dbi services
by Sequotech

The rules have changed

The old model

Security as a binary gate

GRANT / REVOKE

Row Level Security: allow or deny

Query succeeds or returns 'permission denied'

Outcome: binary - it works or it doesn't

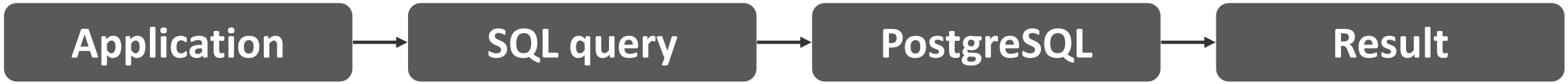
GRANT ——→ ✓ Access

REVOKE ——→ ✗ Denied

The new access pattern

Agents, RAG pipelines, MCP servers

Traditional



AI workflow

The quality of the answer depends entirely on what the model was allowed to see.

The paradigm shift

From binary gate to continuous function

Binary: ✓ or ✗

Continuous: 0.0 ←——————————→ 1.0

(blocked) (full access)

Every RLS policy, every anonymization rule, every role restriction no longer just allows or blocks.

It shapes the quality of the AI's output.

Why security debt is more expensive now

The compounding effect

Traditional :

- Missing RLS → compliance risk
- Missing audit → audit failure

Consequence: binary risk

With AI:

- Missing RLS → compliance risk
  + data leakage through AI answers
  + uncontrolled output quality
  + no audit trail for AI decisions

Consequence: compounding risk

The cost of unpaid debt

Reusing yesterday's mistake — the framework trap

1. THE FRAMEWORK DEFAULT

   LangChain's PGVectorStore — single connection, single role, full access.

   RLS exists in the database. The application never engages it.

2. THE PARALLEL STORE

   Vectors in Pinecone / Weaviate. Operational data in PostgreSQL.

   Two sources of truth. PostgreSQL's RLS only governs half the system.

3. "TRUST THE LLM" AS ACCESS CONTROL

   LLM agent connects via MCP with a service-account role.

   LLMs are not security boundaries.

dbi services
by Sequotech

PostgreSQL's security arsenal

Row Level Security (RLS)

Not just access control anymore

```
CREATE POLICY customer_isolation ON documents
  USING (tenant_id = current_setting('app.tenant_id'));
```

Traditional: controls which rows a user sees

AI workflow: controls which rows the embedding search returns

→ directly shapes the LLM's context window

pgaudit

Traditional audit vs AI workflow audit

Traditional audit:

✓ Who accessed what

✓ When was it accessed

✓ Which query was executed

AI workflow audit :

✓ Who accessed what, when

✗ What was retrieved for AI reasoning?

✗ What answer was produced?

✗ Was the retrieval quality sufficient?

Data anonymization

The semantic drift problem

Real data:

"Big Pharma Basel CHF 2.3M Q4 portfolio"

→ embedding vector [0.23, -0.41, 0.87, ...]

Anonymized:

"Company_A Location_X Amount_Y Period_Z"

→ embedding vector [0.71, 0.12, -0.33, ...]

Transparent Data Encryption

The at-rest layer

Critical insight for AI:

Embeddings encode sensitive business information.

An embedding of a confidential contract still contains

semantic information about that contract.

TDE protects the knowledge vectors encode.

Not in community PostgreSQL (incl. 18) — available via Percona pg_tde, Cybertec, EDB, Fujistsu…

You can't encrypt the vector column and still ANN-search it, TDE protects disk, not the live retrieval path.

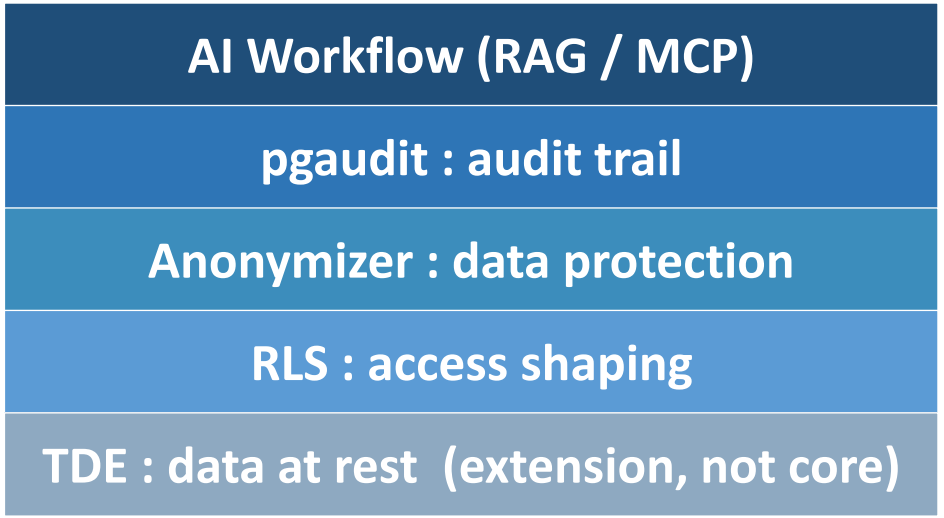
The governance framework

How the stack fits together

Each layer is necessary.

But what is the cumulative cost of all these controls on what your AI retrieves?

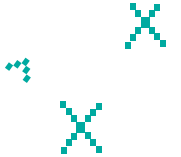
That's where we need a new kind of measurement.



dbi services
by Sequotech

Measuring the cost

A new dashboard for the DBA



dbi services
by Sequotech

Metrics that measure security impact

- Recall@k
- Precision@k
- nDCG@k

Recall@k

The coverage cost of your security controls

Of all relevant rows that exist,how many did your policies let through?

$$\text{Recall@k} = \frac{\text{relevant rows retrieved}}{\text{total relevant rows in DB}}$$

Apply RLS policy → Recall drops from 0.95 to 0.93 → acceptable trade-off

Apply anonymization → Recall drops to 0.60 → governance decision needed

Recall quantifies the coverage cost of each security control.

Precision@k

Semantic drift from security controls

Of everything the model retrieved,
how much was actually relevant?

$$\text{Precision@k} = \frac{\text{relevant rows in top-k}}{k}$$

Anonymization shifts embedding distances.
Vector search pulls in rows that look close
but aren't actually relevant.

Low precision = security controls injecting noise into AI reasoning.
A form of silent failure no traditional monitoring catches.

nDCG@k

The ranking impact you can't see

Even if the right rows are retrieved,
are they ranked correctly?

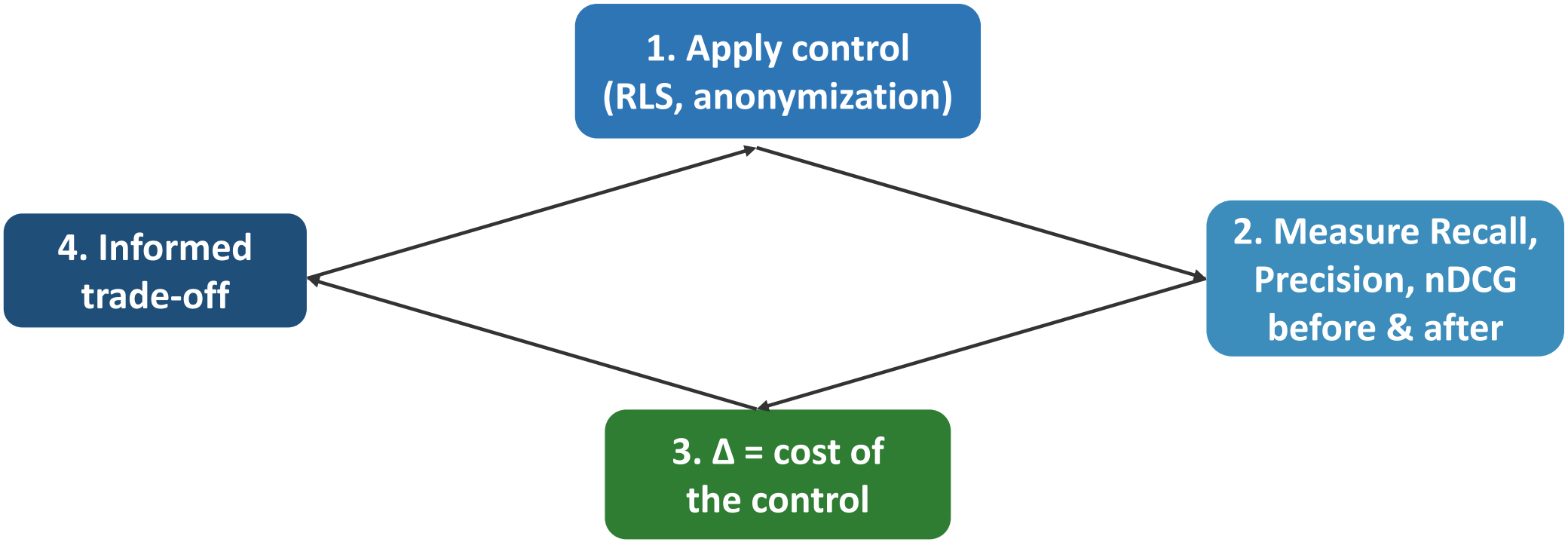
Before anonymization: Best match at position 1
After anonymization: Best match at position 6
Pipeline reads top-5: Best match never seen by LLM

The query succeeded. pgaudit logged it. RLS allowed it.
But the answer is wrong.

nDCG catches the failures invisible
to every other monitoring tool in your stack.

The measurement framework

Apply → Measure → Quantify → Decide



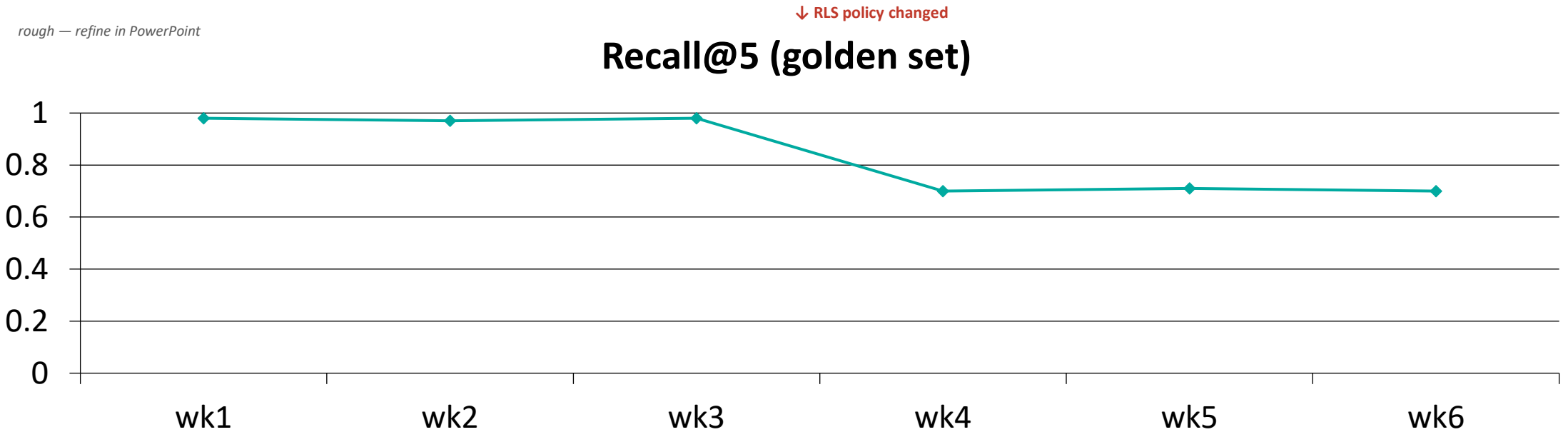
The formula

$$\text{Security Control Cost} = \Delta(\text{Recall}, \text{Precision}, \text{nDCG})$$

Anomaly detection signals

When metrics change unexpectedly

rough — refine in PowerPoint



1. Regression gate — run a golden eval set in CI and after any
   RLS / anonymization / embedding-model change; block the deploy on a drop.
2. Drift detection — watch the eval-set metrics trend over time.
3. Proxies for live traffic — LLM-as-judge, user feedback.

Don't just log access — gate on the quality of what that access produces.

What regulators actually want to see

Evidence of effectiveness over time

Traditional audit (still the bedrock):

→ Controls documented and in place

→ Access logs : who, what, when

→ Change-management trail

→ Incident records, residual-risk decisions

The metrics don't replace any of that.

CONTINUOUS PROOF THAT CONTROLS REMAIN EFFECTIVE.

Old answer:

"RLS is enabled. pgaudit is logging."

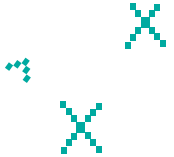
(True on audit day.)

Better answer:

"RLS is enabled. pgaudit is logging.

And our retrieval metrics have held within 2% for 90 days, proving the controls are still doing their job."

Vertical defensibility through PostgreSQL



dbi services
by Sequotech

Sovereignty is the only durable moat

→ Your data stays in YOUR infrastructure
→ Vectors live next to the operational data they describe
→ RLS governs transactional rows AND retrieval
→ Anonymization applies once, consistently
→ One audit trail. One governance model. One source of truth.

That is PostgreSQL's vertical defensibility.

The DBA's evolving role

Security + AI observability = new superpowers

Traditional DBA:

- Index tuning
- Backup management
- Performance monitoring
- Security policy enforcement

AI-era DBA:

- + HNSW/DiskANN tuning for ANN recall
- + Retrieval quality monitoring
- + Security impact measurement
- + Governance framework operator

Not replacement — up-skilling.

Security debt has never been more expensive.
Now you can measure exactly what you owe,
and pay it down where the data lives.

dbi services
by Sequotech

DEMO

Beyond PostgreSQL

Where the rest of the stack picks up

PostgreSQL is the data-layer foundation.

The complete picture also includes:

→ PII DETECTION & DLP

Microsoft Presidio (OSS), Microsoft Purview, Google Cloud DLP, AWS Macie

Content-level detection — complements anonymization

→ LLM GUARDRAILS

NeMo Guardrails, LLM Guard, Lakera Guard

Prompt injection, output filtering at inference

→ AI GATEWAYS

LiteLLM, Portkey, Helicone

Per-role model routing, rate limits, cost controls

→ RETRIEVAL OBSERVABILITY

Langfuse, LangSmith, Arize Phoenix

pgaudit's cousin at the AI layer

Resources

Lab & Demo Code:
github.com/boutaga/pgvector_RAG_search_lab

PostgreSQL Extensions:
- pgvector — github.com/pgvector/pgvector
- pgaudit — github.com/pgaudit/pgaudit
- postgresql_anonymizer — postgresql-anonymizer.readthedocs.io

Further Reading:

- pgvectorscale — github.com/timescale/pgvectorscale
- FINMA Circular 2023/1 — Operational risks and resilience
- Bruce Momjian — momjian.us/main/blogs

dbi services
by Sequotech

Confidentiality: public

THANK YOU!

ANY QUESTIONS? PLEASE, DO ASK

- info@dbi-services.com
- www.dbi-services.com

OUR OFFICES

Basel | Bern | Delémont | Nyon | Zurich