Revocation Window in RAG: Quantifying Unauthorized Retrieval Exposure During Access-Control Propagation

Asst. Prof. Snehal Patil[1], Pragya Rai[2], Chetna Umakant Shete[2], Omkar Sanjay Jadhav[2], Swaraj Santosh Godalkar[2]
[1] Department of Master of Computer Applications, D.Y. Patil Institute of Master of Computer Applications and Management, Akurdi, Pune, Maharashtra, India.
[2] Students, Department of Master of Computer Applications, D.Y. Patil Institute of Master of Computer Applications and Management, Akurdi, Pune, Maharashtra, India.

Abstract—Retrieval-Augmented Generation (RAG) systems often store document chunks in a vector index together with a copy of their access-control metadata. When a user's permission is revoked in the authoritative permission store, this indexed copy may remain stale, allowing the vector database to keep returning protected content. This paper measures the duration of this revocation window and separates two outcomes: unauthorized index retrieval, in which the vector database returns a protected chunk, and LLM-context exposure, in which that chunk reaches the context assembled for the language model. We built a controlled synthetic testbed with PostgreSQL as the authoritative permission store and Qdrant as the vector index, and evaluated six enforcement configurations: periodic synchronization, event-driven synchronization, a no-worker control, query-time authorization, hybrid authorization, and query-time authorization with a 10-second decision cache. Each configuration retained 27 trials (5 for the control) within a 35-second observation window. The median context-level revocation propagation time was 9.92 s for 20-second periodic synchronization, 14 ms for event-driven synchronization, and 4.98 s for the cached query-time configuration. Under periodic synchronization, 87.7% of relevant queries placed the protected chunk in the context before revocation was enforced. Query-time and hybrid authorization blocked protected content at the first post-revocation probe (0 ms at probe resolution) even though the stale index kept returning it, while the no-worker control never achieved stable enforcement within the window. These findings describe the tested synthetic testbed and are not production guarantees.

Keywords: Retrieval-Augmented Generation (RAG), Access Control, Permission Revocation, Vector Databases, Authorization, Revocation Propagation Time, RAG Security

I. Introduction

Retrieval-Augmented Generation (RAG) enables a language model to answer questions using passages retrieved from an external document collection. In many enterprise RAG designs, the document collection is divided into chunks, converted into embeddings, and stored in a vector index. Each chunk may also carry a copy of the access-control metadata associated with its source document, allowing retrieval to be restricted according to the requesting user's permissions. However, the authoritative record of these permissions is often maintained separately, for example, in a relational database. When an administrator revokes a user's access, the corresponding permission information in the vector index may remain outdated until an update mechanism propagates the change.

Security guidance recognizes the importance of this issue. OWASP recommends carrying access-control information onto individual vector chunks [1], AWS describes metadata filtering and layered access controls for generative AI workloads [2], and Microsoft's Azure AI Search documentation discusses permission updates and their propagation behavior in its preview access-control features [3]. These sources provide implementation guidance for securing retrieval-augmented generation, an approach introduced by Lewis et al. [4]. However, the cited materials do not provide directly comparable empirical measurements of how long revoked permissions remain effective in a RAG retrieval pipeline or how frequently protected content continues to be retrieved during that period.

We refer to this period as the "revocation window". Two distinct outcomes can occur during this window. First, the vector index may continue returning a chunk from a document that the user is no longer authorized to access. We refer to this as "unauthorized index retrieval". Second, the retrieved chunk may pass through the authorization and context-building stages and reach the context assembled for the language model. We refer to this as "LLM-context exposure". A system may experience unauthorized index retrieval without LLM-context exposure, for example, when a fresh authorization check removes the protected chunk before the context is constructed. Therefore, measuring only index-level retrieval or only context-level exposure does not provide a complete picture of revocation behavior.

This study measures the revocation window using a controlled synthetic testbed built with PostgreSQL as the authoritative permission store and Qdrant as the vector

index. Six enforcement configurations are evaluated: periodic synchronization, event-driven synchronization, a no-worker control, query-time authorization, hybrid authorization, and query-time authorization with a 10-second decision cache. The study makes the following contributions:

- We define and evaluate separate metrics for unauthorized index retrieval and LLM-context exposure. We define both index-level and context-level Revocation Propagation Time (RPT), and report context-level RPT results alongside index-level and context-level exposure rates, retaining right-censored trials in the analysis.
- We provide a reproducible experimental testbed and experiment suite, with implementation code, experiment logs, and cleaned trial data made available in the project repository.
- We report empirical measurements across all six enforcement configurations. Within a 35-second observation window, the median context-level revocation propagation time was 9.92 s for 20-second periodic synchronization and 14 ms for event-driven synchronization. Query-time and hybrid authorization blocked protected content at the first post-revocation probe, even while the stale index continued returning it. In contrast, query-time authorization with a 10-second decision cache produced a median context-level revocation propagation time of 4.98 s.

The event-driven configuration enforced revocation in every retained trial; however, only 13 relevant context probes occurred within its short revocation interval, which was insufficient to obtain a reliable direct estimate of its context-exposure rate. All results were obtained using a synthetic dataset on a single machine and should not be interpreted as production performance guarantees.

The remainder of this paper is organized as follows: Section II reviews related work, Section III presents the research methodology and experimental testbed, Section IV describes the metrics and experimental setup, Section V reports the results and analysis, Section VI discusses the findings, Section VII presents the limitations, and Section VIII concludes the study and outlines future work.

II. Literature Review

A. Security Guidance for Permission-Aware Retrieval

Practitioner guidance treats access control as a property that must be maintained throughout the retrieval pipeline. OWASP recommends propagating access-control metadata to every vector chunk [1]. AWS describes a multilayered strategy that combines metadata filtering, access controls, and data redaction at different stages of a RAG pipeline [2]. Microsoft Azure AI Search provides several approaches to document-level access control, including security filters, ACL and RBAC scopes, and sensitivity labels. Its native query-time enforcement features evaluate the caller's identity against permission metadata associated with indexed documents [3]. However, permission freshness depends on the ingestion mechanism and data source. Permission changes may be reflected after a subsequent indexer run, resynchronization, or reingestion, depending on the configuration [3]. This guidance recognizes that indexed permission metadata can become stale when source permissions change, but it does not provide controlled empirical measurements of how long a revoked user remains able to retrieve protected content.

B. Permission-Aware RAG Systems

Several recent studies place access control directly within the retrieval pipeline. Jeong and Lee propose a permission-aware RAG framework that validates user permissions through identity and access management (IAM) information across multiple resources [5]. Chen et al. present a proof-of-concept implementation integrating access control with RAG for sensitive patient profiles [6]. Chen and Taipalus evaluate a metadata-aware design that combines role-based access control with metadata filtering [7]. These works demonstrate approaches for enforcing permissions during retrieval. Their reported evaluations focus on permission filtering, prototype functionality, and retrieval performance rather than controlled measurements of the time between a permission change and its enforcement across the retrieval pipeline.

C. Privacy and Security of RAG

A second line of research investigates information leakage and privacy risks in RAG systems. Zeng et al. empirically demonstrate that RAG systems can expose private information contained in their retrieval databases [8]. A recent survey by Mu et al. reviews security threats, defense mechanisms, and benchmarks for secure RAG systems [9]. These studies establish that retrieved content can introduce confidentiality risks. However, they do not specifically connect these risks to the timing of access-control revocation, which is the focus of the present study.

D. Access-Control Models and Revocation as a Consistency Problem

Classical access-control models define which users or principals may access particular resources. Kuhn et al. extend role-based access control by incorporating attributes [10], while Hu et al. provide a foundation for attribute-based access control [11]. These models specify

how access decisions are determined, but do not by themselves establish how quickly a changed authorization decision propagates across replicated or cached permission records.

Revocation delay has also been examined as a consistency problem outside RAG. Parakhin models authorization revocation in multi-agent systems using concepts from cache coherence and derives bounds on unauthorized operations under bounded-staleness assumptions. For example, the paper illustrates that a 60-second revocation window can permit approximately 6,000 unauthorized API calls under its assumed rate of 100 operations per tick [12]. Tallam formalizes authorization propagation in multi-agent AI systems and identifies temporal validity as a core concern [13]. Kleppmann and Riccomini discuss the challenges of maintaining consistency across replicated and cached state in distributed systems [14]. These works provide conceptual foundations for understanding revocation delay, but they do not specifically evaluate stale authorization metadata in vector retrieval pipelines or measure the separate outcomes of index retrieval and LLM-context exposure.

E. Stale State in Vector Indexes

The closely related work by Chakraborttii et al. demonstrates that soft-deleted embeddings can remain physically present and reconstructible in HNSW vector databases [15]. Their study focuses on the persistence and recovery of deleted data within vector indexes. In contrast, the present study investigates how stale permission metadata allows a revoked user to continue retrieving protected content and whether that content reaches the context assembled for the language model.

F. Summary and Research Gap

TABLE I summarizes the relationship between the reviewed research areas and the present study.

TABLE I. SUMMARY OF REVIEWED RESEARCH

Area	Representative Work	What it Addresses	Gap Relative to this Study
Industry guidance	[1], [2], [3]	Metadata on chunks, layered controls, and document-level access	Controlled measurements of revocation delay and exposure
Permission-aware RAG	[5], [6], [7]	Enforcing permissions during retrieval	Revocation propagation time across the pipeline
RAG privacy and security	[8], [9]	Information leakage, threats, and defenses	Specific link between permission-revocation timing and exposure
Access control models	[10], [11]	Authorization policies and access decisions	Propagation delay of changed permissions
Revocation consistency	[12], [13], [14]	Bounded staleness and consistency in distributed systems	Measurements in RAG retrieval pipelines
Vector index state	[15]	Persistence and reconstruction of deleted embeddings	Stale permissions and LLM-context exposure

To the best of our knowledge, the studies reviewed here do not quantify revocation delay across the asynchronous layers of a RAG pipeline while separately measuring unauthorized index retrieval and LLM-context exposure. This statement is limited to the literature reviewed and does not imply that no related work exists. The present study addresses this gap through controlled measurements of six enforcement configurations.

III. Research Methodology and Experimental Testbed

A. Research Design

This study uses a quantitative, controlled laboratory experiment with a comparative design. The same type of permission revocation is repeated under six enforcement configurations, and the resulting exposure is measured and compared. The unit of analysis is one permission revocation (one trial). All data are primary data recorded by the testbed's own logs; no human participants are involved.

The study has four objectives: (1) to build a controlled RAG testbed in which a user's permission can be revoked while retrieval requests continue; (2) to measure how long a revocation takes to be enforced; (3) to measure, during that time, how often the index returns the protected document and how often its text reaches the language model's context; and (4) to compare the six configurations on window length, exposure, and latency. We evaluate four hypotheses against the results:

- H1. Under periodic synchronization, the revocation window is greater than zero and approximately bounded by the synchronization interval, allowing for measurement resolution and the stability-detection rule.
- H2. While the index is out of date, a revoked user's questions about the document frequently place its text in the language model's context.
- H3. Event-driven synchronization substantially reduces the revocation window compared with periodic synchronization.
- H4. Checking authorization against the authoritative store when the context is built (query-time and hybrid) prevents context exposure even while the index is out of date.

B. Testbed Architecture

PostgreSQL is the authoritative store for users, groups, group memberships, documents, and the access-control list (doc_acl); a revocation deletes a row from doc_acl. Qdrant stores the document chunks and serves similarity search. Each chunk carries a copy of its document's access list in an indexed allowed_principals payload (for example, user:u12, group:g_proj_3, public:*). Text is embedded with all-MiniLM-L6-v2 [16], [17], which produces 384-dimensional normalized embeddings; Qdrant uses cosine distance. For each question, the retriever fetches the 20 nearest chunks, and the context builder keeps the top 5 that pass any authorization gate. No text is generated; the study measures exposure to the context.

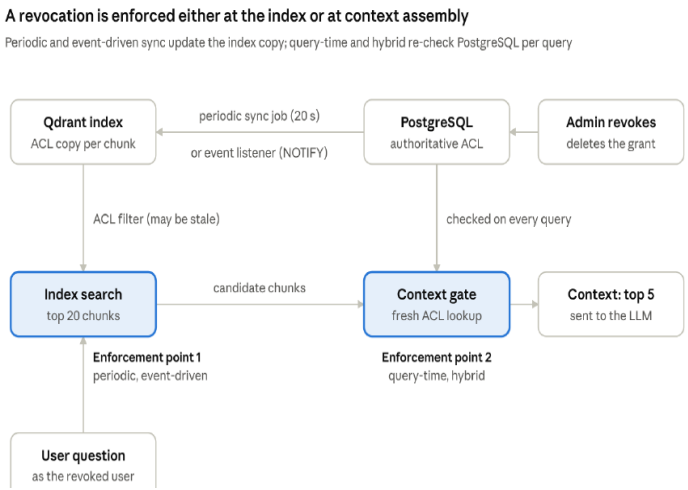


Fig. 1. Testbed architecture showing the two enforcement points: index filtering for periodic/event-driven synchronization and a fresh PostgreSQL context gate for query-time/hybrid authorization.

Table II summarizes the experimental environment.

TABLE II. TESTBED ENVIRONMENT

Components	Versions / Configurations
Machine	Free-tier GitHub Codespace, 2 CPU cores, 7.8 GiB RAM
Python	3.14.2
PostgreSQL	16.15 (Docker image postgres:16.15)
Qdrant	1.20.0 (Docker image qdrant/qdrant:v1.20.0), telemetry disabled
Docker and Docker Compose	Default versions in the Codespaces image (not pinned)
Embedding Model	all-MiniLM-L6-v2 via Sentence Transformers, CPU
Code Version	Git commit c2233d7; exact library versions in requirements.lock

All components run on one machine and communicate over localhost.

C. Synthetic Dataset

All data are synthetic and generated with a fixed random seed (42), so every run starts from the same generated data. The dataset contains 1,000 users in 10 departments, 60 groups (10 department groups and 50 project groups), 2,547 group memberships, 2,500 documents, and 4,897 chunks of 300 words with a 30-word overlap. There are 3,501 ACL rows: about 60% of documents are public, about 30% are group-restricted, and about 10% are restricted to named users. A cleaning step ran 16 integrity checks (duplicates, missing fields, dangling references, documents without an access list, and chunk access lists that differ from the ACL table); none affected any row. After loading, a consistency check confirmed that every chunk's payload matched the PostgreSQL access list for its document. The generated data folders are not stored in the repository; the setup script rebuilds them. The collection was searched exactly: the saved setup output records that no HNSW index was built (0 of 4,897 vectors indexed), so all searches were brute force. Results for large collections served by approximate search may differ.

D. Enforcement Configurations

Table III lists the six configurations. In the query-time and hybrid configurations, the index is deliberately never updated after a revocation, so the PostgreSQL gate alone must enforce it.

TABLE III. ENFORCEMENT CONFIGURATIONS

Configuration	Enforced at	Mechanism
Periodic synchronization (20 s)	Index filter	Every 20 s a job copies the access list of each document flagged as changed from PostgreSQL to Qdrant.
Event-driven synchronization	Index filter	The revoking transaction sends a pg_notify event; a listener updates that document's access list in Qdrant. The change is also flagged durably in case the event is lost.
Control (no worker)	Nowhere	The propagation worker is disabled; the revocation is recorded in PostgreSQL but is not propagated to Qdrant.
Query-time authorization	Context gate	The index is searched without a permission filter; before the context is built, a fresh PostgreSQL query keeps only documents the user may read now.
Hybrid authorization	Index filter, then context gate	The possibly stale index filter narrows the candidates; the fresh PostgreSQL check makes the final decision.
Query-time, 10 s cache	Context gate	Query-time authorization that reuses each authorization decision until a 10-second cache lifetime expires.

The no-worker control is a stale-state reference, not an enforcement strategy.

E. Validity, Reproducibility, and Ethics

Only pairs whose deleted permission was the user's only access were used, and each trial first confirmed that the user could retrieve the document. Periodic revocations occurred at random points in the cycle, and enforcement time (reference search) and leakage (workload search) were measured separately at two points. Configurations used the same dataset, machine, and protocol but different randomly sampled pairs, so the comparison is not matched-pair. A separate validation run (Section IV-F) checked the query model. The repository provides a fixed random seed, pinned Python dependencies, a setup script, and an automatic consistency check; an independent rebuild on another machine is not documented. All experiments ran in an isolated environment with synthetic users and documents; no real permissions, confidential data, or external systems were involved.

IV. METRICS AND EXPERIMENTAL SETUP

A. Revocation Procedure

Each trial measures one revocation of a direct user grant. The procedure is the same for every configuration:

- **Select a pair.** A user–document pair is chosen at random from the 455 eligible pairs, in which the user's direct grant is their only path to a restricted document (no public row and no group the user belongs to). Deleting the grant therefore removes the user's access.
- **Baseline.** The user must first retrieve the document into the context. Otherwise the trial is logged as a baseline failure and skipped.
- **Random timing (periodic synchronization only).** The script waits a random time within the synchronization interval, so revocations land at random points in the cycle.
- **Revoke.** The grant is deleted in PostgreSQL, with a dirty flag and change notification where the configuration requires them. T_revocation is recorded when the transaction commits.
- **Probe.** For 35 seconds the pipeline is probed repeatedly as the revoked user. Each probe runs two searches.
- **Restore.** The grant is re-inserted in a finally block, so it also happens after a crash or interruption. Such trials are logged and excluded.

The two searches in each probe have different purposes. The reference search uses the exact vector of a chunk of the protected document and measures when enforcement occurs. The workload search models a realistic question and measures how much leaks. With probability 0.5 it concerns the protected document, with query vector $\alpha \cdot$(document chunk) + $(1 - \alpha) \cdot$(chunk of a different document) and α drawn uniformly from 0.2 to 1. Otherwise it is an unrelated question. Each probe records which kind it was. In the query-time and hybrid configurations, each probe also runs the PostgreSQL gate queries.

B. Experimental Settings

Table IV lists the settings used for all final runs.

TABLE IV. EXPERIMENTAL SETTINGS

Parameter	Setting
Observation window after revocation	35 s
Periodic synchronization interval	20 s
Probe target gap	100 ms (periodic), 10 ms (event-driven and gate configurations)
Retrieved chunks / context chunks	20 / 5
Probability a workload query is relevant	0.5
Relevance weight α	Uniform, 0.2 to 1
Trials	30 per configuration, 5 for the no-worker control
Warm-up excluded	First 3 trials of each configuration (control exempt)
Bootstrap resamples	2,000
Random seed	42

The probe gaps are measurement settings, not estimates of real user request rates. The periodic

loop targets about 10 probes per second and the other loops about 100. The realized rate is lower when an operation takes longer than the target gap. Trials ran one after another, because they modify the same database and collection. Of 155 trials, the 15 warm-up trials were excluded, leaving 140 analyzed: 27 for each of the five non-control configurations and 5 for the control.

C. Metrics

Two measuring points are checked for every probe. The index is the set of 20 chunks that Qdrant returns, and the context is the 5 chunks handed to the language model. Protected text in the context is the security-relevant outcome, because the model can repeat it. Table V summarizes the metrics, and the formulas follow.

TABLE V. METRICS

Metric	Stage	Meaning
Index RPT	Index	Time until the index stops returning the protected chunk
Context RPT	Context	Time until the protected chunk stops reaching the LLM context
URR	Index	Share of post-revocation index probes that return the protected chunk
CER	Context	Share of post-revocation context probes that include protected content
Latency	Both	Mean time of one probe

Revocation propagation time (RPT) is the length of the revocation window, from revocation to stable enforcement:

$$RPT = t_enforcement - t_revocation \quad (1)$$

Each reference probe records two outcomes: whether the protected chunk is among the 20 chunks returned by the index, and whether it is among the 5 chunks placed in the context. Stable enforcement is evaluated separately for each outcome. It is the first probe of the final run of at least three consecutive probes in which the chunk is absent, and the chunk stays absent until the end of the trial. Index RPT uses the first outcome and context RPT uses the second. This rule also catches a revocation that is briefly enforced and then lost. A reported 0 ms means the gate blocked protected content at the first post-revocation probe, at the resolution of the probe interval. It does not mean zero physical processing time.

The unauthorized retrieval rate and context exposure rate are:

$$URR = N_leak / N_idx \quad (2)$$

$$CER = N_ctx / N_probe \quad (3)$$

where N_leak is the number of post-revocation index probes that return the protected chunk, N_idx is the total number of post-revocation index probes, N_ctx is the number of post-revocation context probes that include protected content, and N_probe is the total number of post-revocation context probes. The two denominators are separate because retrieval and context construction are distinct stages. Whole-window rates depend on the observation window and workload, so they are compared only across runs with identical settings.

The results report the relevant-query rates URR_rel and CER_rel, which count only workload probes about the protected document, evaluated over the interval before enforcement. Unrelated questions would not return the protected document anyway, and counting them would make the leak look smaller. These rates answer a conditional question: when a revoked user asks about the protected document, how often does the index return it, and how often does it reach the context? The interval endpoints differ. The index interval ends when the index stops returning the chunk, and the context interval ends when the chunk stops reaching the context. In query-time and hybrid modes the index is deliberately left stale, so the index interval spans the whole observation window. A rate whose

denominator is zero is reported as undefined, not as zero.

D. Expected Exposures

To give the rates a practical scale, we estimate the expected number of context exposures per revocation as:

$$E = (q \times s \times CER_rel \times mean(RPT_ctx)) / 60 \quad (4)$$

where q is the assumed query rate in queries per minute, s the assumed share of queries about the protected document, and mean(RPT_ctx) the mean context RPT in seconds. We assume q = 1 and s = 0.1. These are assumptions, not observed user behavior. A sensitivity analysis varies q from 0.2 to 5 queries per minute and s from 5% to 30%. Each estimate carries a basis label:

measured when enough relevant probes were observed in the interval;

measured (blocked from the first probe) when no probe fell before enforcement;

estimated when fewer than 30 relevant context probes fell inside the interval, so CER_rel is replaced by the stale-state rate of the no-worker control (0.884);

not estimable when all trials were right-censored within the observation window.

E. Statistical Analysis

A single script computes every metric in the same way for all configurations. Probes that end in an HTTP error are excluded rather than counted as blocked, and trials with more than 5% failed searches are dropped. Every exclusion is logged with its reason. A trial is censored when stable blocking is not observed within the 35-second window. Censored trials are retained, and median and 95th-percentile RPT are estimated with the Kaplan–Meier method [18]. If every trial of a configuration is censored, the result is reported as not enforced within the window, not as a measured time. Rates are pooled over probes, so a short-window trial does not weigh as much as a long-window trial. Their 95% confidence intervals come from 2,000 bootstrap resamples of whole trials, because probes within a trial are correlated. We also report the maximum and mean RPT with a bootstrap interval, and mean probe latency.

F. Validation Run

A separate validation run checked that the workload behaves as intended. It used periodic synchronization at a 10-second interval, a 25-second window and 10 trials, with no warm-up exclusions. Unrelated probes returned the revoked document in 1.4% of 210 probes, and 90.2% of 224 relevant probes placed it in the context. This run is not part of the six-configuration comparison.

V. RESULTS AND ANALYSIS

Table VI reports the context-level revocation propagation time, Table VII the relevant-query exposure rates, and Table VIII the expected exposures. Fig. 2 shows trial-level RPT and Fig. 3 compares the exposure rates. All values come from the final cleaned runs: 27 trials for each of the five non-control configurations and 5 for the control. The separate validation run is reported on its own in Section IV-F and is not pooled into these results.

A. Revocation Propagation Time

TABLE VI. CONTEXT REVOCATION PROPAGATION TIME (35 s WINDOW)

Configuration	Trials	Not enforced	Median (KM)	p95 (KM)	Maximum	Mean RPT [95% CI]
Periodic (20 s)	27	0	9.92 s	19.13 s	20.04 s	10.32 s [7.95, 12.77]
Event-driven	27	0	14 ms	51 ms	52 ms	19 ms [15, 24]
Control (no worker)	5	5	not reached	not reached	never enforced	> 35 s
Query-time	27	0	0 ms	0 ms	0 ms	0 ms [0, 0]
Query-time (10 s cache)	27	0	4.98 s	9.04 s	9.70 s	4.88 s [3.89, 5.83]

Hybrid	27	0	0 ms	0 ms	0 ms	0 ms [0, 0]

Note: The context gate blocked protected content at the first post-revocation probe. This is the probe-resolution result, not instantaneous enforcement.

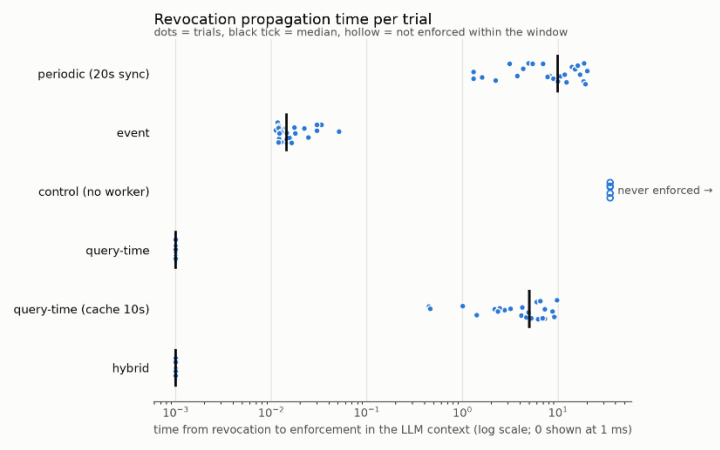


Fig. 2. Revocation propagation time per trial (log scale). Dots are trials, the black tick is the median, and hollow dots are trials not enforced within the 35-second window. Zero values are drawn at 1 ms.

Under 20-second periodic synchronization, context RPT reached up to 20.04 s with a mean of 10.32 s (median 9.92 s). This is consistent with revocations landing at random points in a 20-second cycle, which gives an expected mean near half the interval. The small excess over 20 s is consistent with probe resolution and the three-probe stability rule. Event-driven synchronization enforced revocation in every trial within 52 ms (median 14 ms). Query-time and hybrid authorization blocked protected content at the first post-revocation probe in all 27 trials each. With the 10-second decision cache, the window ranged up to 9.70 s, inside the cache lifetime, with a median of 4.98 s. All five no-worker control trials were still unenforced after 35 s. Because revocations land at random points in the cycle, the synchronization interval and the cache lifetime determine the nominal window; the small excess (maximum 20.04 s for a 20 s interval) reflects probe resolution and the stability rule.

B. Exposure During the Window

TABLE VII. RELEVANT-QUERY EXPOSURE BEFORE ENFORCEMENT

Configuration	Index exposure (URR_rel) [95% CI]	Context exposure (CER_rel) [95% CI]
Periodic (20 s)	0.912 [0.890, 0.937]	0.877 [0.855, 0.904]
Event-driven	0.462 [0.200, 0.733] (few probes)	too few probes to measure (13)
Control (no worker)	0.917 [0.896, 0.947]	0.884 [0.861, 0.916]
Query-time	0.925 [0.914, 0.934]	0 (blocked from the first probe)
Query-time (10 s cache)	0.916 [0.901, 0.929]	0.897 [0.881, 0.912]
Hybrid	0.933 [0.922, 0.944]	0 (blocked from the first probe)

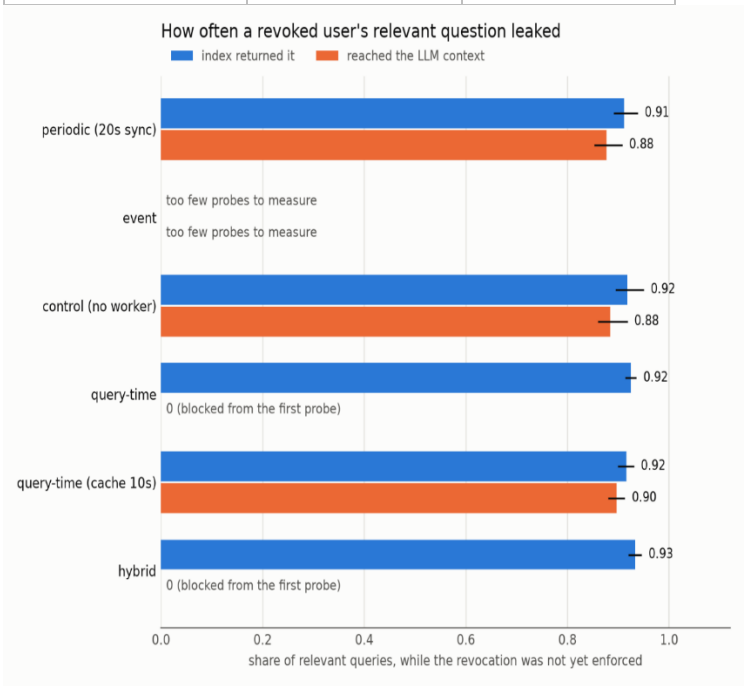


Fig. 3. Share of relevant questions that the index returned (blue) and that reached the LLM context (orange) while revocation was not yet enforced, with 95% confidence intervals. Event-driven synchronization had too few questions inside its window to measure.

While the index was stale, 87.7% of relevant questions under periodic synchronization placed the protected chunk in the context, and almost the same share (88.4%) did so in the no-worker control. In the query-time and hybrid configurations the index still returned the protected chunk for 92.5%

and 93.3% of relevant questions, yet none of it reached the context, because the fresh PostgreSQL check removed every protected chunk. With a 10-second decision cache, 89.7% of relevant questions exposed the chunk during the cache lifetime. Event-driven synchronization closed its window so quickly that only 13 relevant context probes fell inside it across all retained trials, below the 30-probe threshold, so no direct context-exposure rate is reported. Its index-exposure estimate (0.462) also has a wide interval for the same reason.

C. Expected Exposures

TABLE VIII. EXPECTED CONTEXT EXPOSURES PER REVOCATION (q = 1 QUERY/MIN, s = 0.1)

Configuration	Expected exposures	Basis
Periodic (20 s)	0.015	measured
Event-driven	0.000028	estimated: only 13 relevant probes; uses the control stale-state rate (0.884)
Control (no worker)	not estimable	all trials right-censored at 35 s
Query-time	0	measured (blocked from the first probe)
Query-time (10 s cache)	0.0073	measured
Hybrid	0	measured (blocked from the first probe)

Under the assumed workload, periodic synchronization yields about 0.015 expected exposures per revocation, roughly 536 times the estimated event-driven value. The event-driven value is modeled, not directly measured. The query-time and hybrid zeros mean that protected text was blocked at the first post-revocation probe in the observed trials, not that every possible implementation has zero risk. The estimates scale linearly with the assumed workload. Across the sensitivity range (0.2 to 5 queries per minute, 5% to 30% relevant share), periodic synchronization ranges from 0.0015 to 0.226 expected exposures and the cached variant from 0.0007 to 0.109. Event-driven stays between 3×10^{-6} and 4.2×10^{-4}, and query-time and hybrid remain zero throughout.

D. Probe Latency

TABLE IX. MEAN PROBE LATENCY

Configuration	Mean latency (ms)
Periodic (20 s)	9.7
Event-driven	9.9
Control (no worker)	10.0
Query-time	12.5
Query-time (10 s cache)	9.9
Hybrid	12.8

The two uncached gate configurations were about 2.8 to 3.1 ms slower per probe than the periodic baseline (roughly 30%). Because each probe contains two searches plus the gate queries, the added cost per user request is likely smaller than the per-probe figure. Caching the decision removed this cost (9.9 ms) but brought back a window with a median of 4.98 s. These differences are reported descriptively and were not tested for significance. Probe latency covers only the operations in the trial code and is not end-to-end LLM response time.

E. Hypotheses

TABLE X. OUTCOME OF THE HYPOTHESES

Hypothesis	Outcome
H1. Periodic synchronization: window above zero and approximately bounded by the interval	Supported: median 9.92 s and maximum 20.04 s against a 20 s interval (the small excess is consistent with probe resolution)
H2. Stale index: frequent context exposure	Supported: CER_rel 0.877 (periodic) and 0.884 (control)
H3. Event-driven synchronization substantially reduces the revocation window compared with periodic synchronization.	Supported: Median 14 ms and maximum 52 ms, compared with a median of 9.92 s under periodic synchronization. Recovery after a lost event was not evaluated.
H4. Query-time and hybrid: no context exposure	Supported: CER_rel 0 in all 54 trials, but not when the decision is cached (0.897 with a 10 s cache)

VI. DISCUSSION

A. Is the Revocation Window a Security Issue?

A delay by itself is not a security problem. It becomes one when protected data reaches a user who is no longer allowed to see it. The results show this step by step. A gap exists: under periodic synchronization the median context RPT was 9.92 s, and with no propagation worker the revocation was never enforced within 35 s. The revoked user could still retrieve the document, with relevant-query URR of 0.912 under periodic synchronization. The protected text reached the language model's context for 87.7% of relevant questions. And it was preventable: query-time and hybrid authorization produced no context exposure in any of the 54 trials. A stale permission copy in the index is therefore a context-exposure risk in a RAG system that relies on it alone. Because no model answers were generated, the experiment shows a risk of disclosure, not user-visible disclosure.

B. Synchronization Narrows the Window but Cannot Close It

Under periodic synchronization, revocations arrive at random points in the cycle, so the window is spread between zero and the interval, with a mean near half of it (10.32 s for a 20 s interval, maximum 20.04 s). The interval is therefore a security parameter, not only a performance setting. Under Equation (4), expected exposure scales with the mean window, so halving the interval would roughly halve the expected exposure but could not bring it to zero. Event-driven synchronization reduced the window by orders of magnitude (median 14 ms, maximum 52 ms), but the no-worker control shows that this protection depends entirely on the listener running: with no propagation, no trial was enforced within the window. The implementation records a durable dirty flag but does not run a reconciliation sweep, so worst-case recovery after a lost event or a crashed listener was not tested. A periodic sweep or a durable queue is a recommended safeguard here, not a tested feature.

C. Enforcing at Context Assembly Contains a Stale Index

Where a revocation is enforced mattered more than how fast the index was updated. In the query-time and hybrid configurations the index was never updated and kept returning the protected chunk (URR_rel 0.925 and 0.933), yet nothing reached the context, because the fresh PostgreSQL check removed it. In practice, the access list stored in the index is best treated as a performance filter, and the authorization decision belongs at context assembly, against the authoritative source. This is in line with the layered controls described in industry guidance [1], [2]. The price is one PostgreSQL query per request: about 2.8 to 3.1 ms more per probe (roughly 30%) in this testbed. Hybrid authorization narrows candidates with the index filter first, which could reduce the number of retrieved chunks the gate must discard, but this study did not measure that benefit, and its latency (12.8 ms) was no lower than query-time authorization (12.5 ms). A stale index can still matter on its own: if retrieval results are logged, cached, or returned to other services, index-level remnants remain a risk, as work on soft-deleted embeddings also suggests [15].

D. A Cache Is a Revocation Window of Its Own

Caching the authorization decision reduced the mean probe latency to 9.9 ms, close to the periodic baseline of 9.7 ms. However, it reintroduced a revocation window, with a median of 4.98 s and 89.7% relevant-query context exposure. The median near half the cache lifetime and the maximum just below it are consistent with revocations arriving at random points within a cache entry's lifetime. Query-time authorization is therefore only as fresh as its cache. This echoes the bounded-staleness view of revocation in distributed systems [12]: the lifetime of a cached decision sets a nominal upper bound on the exposure window and should be chosen and documented as a security parameter.

E. Measure Index and Context Exposure Separately

A study that counted only what the index returns would have reported the query-time and hybrid configurations as leaking for most relevant questions (0.925 and 0.933), although no protected text reached the context. Measuring both stages separates "the index is stale"

from “protected content reached the model”. We recommend that RAG evaluations report both.

VII. LIMITATIONS

The results should be read with the following limitations.

- **Synthetic data and workload:** Users, documents, ACLs, and questions were generated. Questions were built by mixing the protected document’s vector with an unrelated chunk’s vector, which let us control how closely each question matched the document. This affects mainly the exposure percentages (CER), not the window lengths or the comparison between configurations. Real documents have clearer topics, so questions would probably find them at least as easily; a validation run found that unrelated questions returned the protected document in only 1.4% of 210 probes.
- **One kind of revocation:** We removed a user’s direct permission on a document. Removing a user from a group works differently: the document’s permission list does not change, so the index stays correct, but the user’s cached group list (for example in a session) stays stale until refreshed. That window depends on the identity system, not on index synchronization, and was not tested.
- **Single machine:** The database, vector index, and synchronization process shared a free-tier 2-core Codespace and communicated over localhost. A distributed deployment may add network delays but can also change scheduling and contention, so the reported values describe this testbed and are not a production lower bound.
- **Exact search:** With 4,897 chunks, no HNSW index was built and all searches were exact. Latency and ranking behavior would differ at scale; the permission-leak mechanism does not depend on the search method.
- **Probe resolution:** Probes ran every 100 ms (periodic) or 10 ms (others), so a window cannot be measured more finely. A 0 ms result means “blocked at the first probe”.
- **Event-driven exposure not directly measured**: Only 13 relevant context probes fell inside the event-driven window, so its CER was not measurable; its expected exposure is an estimate based on the control’s stale-state rate.
- **Assumed user behavior:** Expected exposures depend on the assumed query rate and relevant-query share. The sensitivity analysis shows the scaling but is not a measurement of real users.
- **Sample size and comparison design:** About 27 trials per configuration (5 for the control) support medians, 95th percentiles, and confidence intervals, but not rare extremes. Configurations used the same dataset, machine, and protocol but different sampled pairs, so the comparison is not matched-pair.
- **No generated answers:** The study stops at the context, so it does not measure how often the model’s answer would repeat protected text. Latency covers only the trial code, not end-to-end response time.
- **Recovery not tested:** The event-driven implementation does not include a reconciliation sweep, so recovery after a lost event or a failed listener was not evaluated.

VIII. CONCLUSION AND FUTURE WORK

A. Conclusion

This paper measured the revocation window in a controlled RAG testbed built on PostgreSQL and Qdrant, and separated unauthorized index retrieval from LLM-context exposure. Under 20-second periodic synchronization, protected text kept reaching the context for a median of 9.92 s (maximum 20.04 s), and 87.7% of relevant questions in that period exposed it. Event-driven synchronization reduced the median window to 14 ms (maximum 52 ms), but its protection depends on the propagation worker: with no worker, stable revocation was not observed in any of the five

trials within the 35-second observation window. Query-time and hybrid authorization blocked protected text at the first post-revocation probe in all 54 trials, despite stale index retrieval, with about 3 ms (2.8–3.1 ms) of additional per-probe latency compared with periodic synchronization in this testbed. A 10-second decision cache brought a window back, with a median of 4.98 s and 89.7% relevant-query context exposure.

These results lead to four practical conclusions for the tested architecture. First, the final authorization decision should be made at context assembly against the authoritative permission store, and the access list in the index is best treated as a performance filter. Second, the synchronization interval and any authorization-cache lifetime are security parameters and should be chosen and documented as such. Third, event-driven synchronization needs a recovery mechanism for lost events or a failed listener. Fourth, index retrieval and context exposure should be reported separately, because a stale index does not by itself mean that protected content reached the model.

These findings describe a synthetic testbed on a single machine. They demonstrate a context-exposure risk, not user-visible disclosure, because no model answers were generated, and they are not production guarantees.

B. Future Work

Several extensions would broaden these results:

- **Group-membership revocation and token lifetimes:** Removing a user from a group leaves the document's access list unchanged but can leave the user's cached group list stale. Measuring this second window depends on the identity system.
- **Authorization-cache lifetimes:** The relationship between cache lifetime and window length could be measured across several TTL values, rather than only the 10-second configuration evaluated in this study.
- **Bulk revocations:** Offboarding a team would show how synchronization-based designs degrade under load.
- **Recovery for event-driven synchronization:** Adding a durable queue and periodic reconciliation would allow the worst-case window after a lost event or a crashed listener to be measured.
- **Scale and realism:** Larger real-text corpora, approximate (HNSW) search, and a distributed deployment would test whether the window lengths and latency costs hold outside a single 2-core machine.
- **Hybrid authorization:** Measuring how often the index pre-filter reduces the number of chunks the gate must discard would show whether hybrid design has a benefit beyond query-time authorization.
- **A stronger experimental design**: Matched user-document pairs across configurations and more trials would support comparisons between configurations and estimates of rare extremes.
- **End-to-end disclosure:** Running a language model on the exposed context would measure how often context exposure becomes disclosure in the generated answer.

DATA AVAILABILITY

The code, experiment logs, and cleaned trial data are available at:

https://github.com/prgyya/rag-revocation-window.

REPRODUCIBILITY

The setup script regenerates the synthetic dataset using a fixed random seed (42). Python dependencies are documented in requirements.lock.

REFERENCES

[1] OWASP, "RAG Security Cheat Sheet," OWASP Cheat Sheet Series. [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/RAG_Security_Cheat_Sheet.html. [Accessed: Nov. 10, 2026].

[2] Amazon Web Services, "Capability 3: Providing secure access to data and systems for generative AI," AWS Prescriptive Guidance: Security Reference

Architecture for Generative AI. [Online]. Available: https://docs.aws.amazon.com/prescriptive-guidance/latest/security-reference-architecture-generative-ai/gen-ai-agents.html. [Accessed: Nov. 10, 2026].

[3] Microsoft, “Document-level access control in Azure AI Search,” Microsoft Learn. [Online]. Available: https://learn.microsoft.com/en-us/azure/search/search-document-level-access-overview. [Accessed: Nov. 10, 2026].

[4] P. Lewis et al., “Retrieval-augmented generation for knowledge-intensive NLP tasks,” in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020.

[5] J. Jeong and S. Lee, “Permission-aware RAG: Identity and access management (IAM)-based access filtering in multi-resource environments,” IEEE Access, vol. 13, pp. 192819–192835, 2025, doi: 10.1109/ACCESS.2025.3628960.

[6] B. Chen, J. Tackman, M. Setälä, T. Poranen, and Z. Zhang, “Integrating access control with retrieval-augmented generation: A proof of concept for managing sensitive patient profiles,” in Proc. 40th ACM/SIGAPP Symp. Applied Computing (SAC), 2025, pp. 915–919, doi: 10.1145/3672608.3707848.

[7] B. Chen and T. Taipalus, “Metadata-aware RAG for enforcing access control and metadata-based filtering: Proof of concept and evaluation,” in Proc. Int. Conf. Research in Adaptive and Convergent Systems (RACS ’25), Ho Chi Minh City, Vietnam, Nov. 2025, pp. 1–7, doi: 10.1145/3769002.3769952.

[8] S. Zeng et al., “The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG),” in Findings of the Association for Computational Linguistics: ACL 2024, 2024, pp. 4505–4524, doi: 10.18653/v1/2024.findings-acl.267.

[9] Y. Mu et al., “Towards secure retrieval-augmented generation: A comprehensive review of threats, defenses and benchmarks,” arXiv:2603.21654, 2026.

[10] D. R. Kuhn, E. J. Coyne, and T. R. Weil, “Adding attributes to role-based access control,” Computer, vol. 43, no. 6, pp. 79–81, Jun. 2010, doi: 10.1109/MC.2010.155.

[11] V. C. Hu, D. F. Ferraiolo, R. Chandramouli, and D. R. Kuhn, Attribute-Based Access Control. Norwood, MA, USA: Artech House, 2017.

[12] V. Parakhin, “The bureaucracy of speed: Structural equivalence between memory consistency models and multi-agent authorization revocation,” arXiv:2603.09875, 2026.

[13] K. Tallam, “Authorization propagation in multi-agent AI systems: Identity governance as infrastructure,” arXiv:2605.05440, 2026.

[14] M. Kleppmann and C. Riccomini, Designing Data-Intensive Applications, 2nd ed. Sebastopol, CA, USA: O’Reilly Media, 2026.

[15] C. Chakraborttii, J. García Alvarado, S. Abdulofizova, and S. Dwivedi, “Ghost vectors: Soft-deleted embeddings remain reconstructible in HNSW vector databases,” arXiv:2606.18497, 2026.

[16] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence embeddings using Siamese BERT-networks,” in Proc. 2019 Conf. Empirical Methods in Natural Language Processing and 9th Int. Joint Conf. Natural Language Processing (EMNLP-IJCNLP), 2019, pp. 3982–3992, doi: 10.18653/v1/D19-1410.

[17] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, “MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers,” in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020.

[18] E. L. Kaplan and P. Meier, “Nonparametric estimation from incomplete observations,” Journal of the American Statistical Association, vol. 53, no. 282, pp. 457–481, 1958.