# Exfiltration de Données via RAG : Attaques Contextuelles

**Résumé exécutif**

Le Retrieval-Augmented Generation (RAG) s'est imposé comme l'architecture standard pour connecter les modèles de langage aux données d'entreprise privées. En reliant un LLM à un vector store alimenté par des documents internes, il élargit considérablement la surface d'attaque, bien au-delà de ce que couvrent les mécanismes de sécurité traditionnels des LLM. Trois vecteurs dominent : l'empoisonnement du vector store par injection de documents malveillants, l'injection de prompt indirecte dissimulée dans les contenus indexés, et le contournement des contrôles d'accès permettant à un utilisateur de consulter des documents confidentiels hors de son périmètre. Cette exfiltration données RAG s'opère sans exploitation logicielle classique : le modèle lui-même devient le canal de fuite, restituant en langage naturel des informations qu'il n'aurait jamais dû récupérer. Comprendre ces attaques contextuelles est indispensable pour sécuriser un déploiement RAG en production.

📋 EN BREF

- Architecture technique et principes de fonctionnement du modèle
- Cas d'usage concrets en cybersécurité et performance mesurée
- Limites, biais potentiels et considérations éthiques
- Guide d'implémentation et ressources recommandées

Le RAG résout le problème des **hallucinations** et des connaissances obsolètes des LLM en alimentant le modèle avec des documents pertinents récupérés dans une base de connaissances vectorielle. Mais cette solution crée un nouveau problème de sécurité fondamental : le modèle

traite les documents récupérés avec le même niveau de confiance que ses instructions système, sans distinguer le contenu légitime du contenu potentiellement malveillant. Un document empoisonné dans le vector store peut contenir des instructions cachées qui seront exécutées par le modèle lors de la génération de la réponse, avec accès à l'ensemble du contexte incluant le system prompt, les autres documents récupérés et les données de l'utilisateur. L'article fondateur sur le RAG (Retrieval-Augmented Generation) explique l'architecture technique que nous allons attaquer. Les techniques d'AI Red Team fournissent le cadre méthodologique pour auditer ces systèmes. La compréhension des techniques avancées de prompt injection est un prérequis car l'injection indirecte via RAG est le vecteur d'attaque principal. L'architecture GraphRAG avec ses knowledge graphs introduit des surfaces d'attaque supplémentaires sur les relations entre entités. Les travaux de Greshake et al. sur l'injection indirecte et les recherches du Anthropic Red Team constituent les références académiques de ce domaine émergent.

## Vecteurs d'Exfiltration Avancés via les Embeddings RAG

Au-delà des attaques directes sur les pipelines de récupération, les attaquants sophistiqués exploitent des vecteurs secondaires liés à la nature fondamentale des embeddings vectoriels. Lorsqu'un modèle RAG encode des documents sensibles pour les stocker dans un vector store, les représentations vectorielles elles-mêmes peuvent révéler des informations substantielles sur le contenu source, même sans accès direct aux textes originaux. Ce risque est sous-estimé dans la plupart des analyses de menaces portant sur les architectures RAG déployées en production dans les grandes organisations françaises et européennes. Les équipes de sécurité se concentrent généralement sur les contrôles d'accès applicatifs et les mécanismes de détection de prompt injection, en négligeant les risques inhérents aux représentations vectorielles elles-mêmes qui persistent dans le vector store et pourraient être extraites par un acteur malveillant disposant d'un accès partiel au système.

La technique d'embedding inversion consiste à reconstruire partiellement un texte source à partir de son vecteur d'embedding. Des travaux de recherche publiés en 2023 et 2024, notamment autour du projet vec2text du MIT, ont démontré qu'avec des modèles d'inversion spécialisés entraînés sur le même backbone d'embedding, il est possible de récupérer jusqu'à 60 pourcent des

tokens d'un texte original depuis son embedding Ada-002. Pour un système RAG qui stocke des contrats commerciaux confidentiels, des données de ressources humaines ou des notes d'audit interne, cela représente un vecteur d'exfiltration critique qui contourne entièrement les contrôles d'accès applicatifs traditionnels mis en place par l'équipe de développement.

**Membership inference attack** : déterminer si un document spécifique fait partie du corpus RAG sans y accéder directement, en comparant les scores de similarité cosinus avec des documents témoins et leurs variations légèrement modifiées pour détecter une différence statistiquement significative

**Embedding reconstruction** : utiliser des modèles adversariaux entraînés sur le même backbone d'embedding pour inverser les vecteurs et reconstituer des fragments de texte source avec une précision croissante selon la longueur du passage cible

**Timing side-channel** : inférer la densité du corpus et localiser les zones de contenu sensible via l'analyse statistique fine des temps de réponse des requêtes de recherche par similarité dans l'index HNSW

**Cross-encoder leakage** : exploiter les scores de re-ranking produits par les cross-encoders pour trianguliariser et inférer le contenu des documents sources avec une précision bien supérieure à celle obtenue avec les bi-encoders seuls

**Gradient-based extraction** : dans les cas où l'API expose des informations de gradient, utiliser des attaques de type model extraction pour reconstruire progressivement les documents d'entraînement du modèle d'embedding

La défense contre ces vecteurs secondaires repose sur plusieurs mécanismes complémentaires à déployer en couches. L'application de bruit différentiel lors de la génération des embeddings consiste à ajouter un vecteur gaussien calibré selon un budget privacy epsilon, dégradant suffisamment les embeddings pour bloquer la reconstruction efficace tout en préservant la qualité de la recherche sémantique au-delà d'un seuil de similarité acceptable. Des frameworks comme OpenDP ou Google Privacy Library permettent d'automatiser ce calibrage et de vérifier formellement les garanties de confidentialité offertes. La valeur epsilon doit être définie selon la sensibilité des données traitées : epsilon inférieur à 1 pour des données hautement confidentielles comme des données de santé ou des contrats sous NDA, epsilon entre 1 et 10 pour des données modérément sensibles. En complément, le chunking adaptatif qui mélange les passages sensibles avec du contexte moins sensible rend la reconstruction partielle moins utile pour un attaquant.

## Détection et Réponse aux Incidents d'Exfiltration via les Systèmes RAG

La détection d'une exfiltration contextuelle via RAG constitue un défi opérationnel particulièrement complexe pour les équipes SOC, car ces attaques sont précisément conçues pour se fondre dans le trafic légitime du système. Contrairement aux injections SQL ou aux scans de ports réseau, l'exfiltration RAG exploite des fonctionnalités prévues et souhaitées du système, rendant la distinction entre usage normal et comportement malveillant intrinsèquement difficile sans une instrumentation spécifique dédiée et une baseline comportementale établie sur une période représentative d'au moins 30 jours.

> ⚡ **Retour terrain**
>
> J'ai déployé une architecture RAG pour un groupe d'assurance mutualiste qui devait rendre interrogeables 12 ans de circulaires internes. Le chunking naïf à 512 tokens cassait les tableaux comparatifs — le modèle répondait avec des valeurs mélangées entre versions d'une même règle. La migration vers un chunking sémantique par section, avec métadonnées temporelles et pondération de fraîcheur, a réduit les hallucinations factuelles de 73 % en deux semaines de tests.

Le monitoring sémantique des requêtes constitue le premier niveau de détection opérationnel. En calculant la distribution statistique des embeddings de requêtes sur une fenêtre glissante de 24 heures et en la comparant à la baseline des 30 derniers jours via un test de Kolmogorov-Smirnov, toute dérive statistique significative avec p inférieur à 0,05 doit déclencher une alerte SOC de niveau P2 pour investigation. Un attaquant sondant méthodiquement le corpus RAG produit une distribution anormalement uniforme dans l'espace vectoriel de dimension élevée, trahissant une

stratégie d'exploration systématique plutôt que des requêtes métier organiques concentrées autour de quelques clusters sémantiques spécifiques au domaine d'application.

**Logging structuré des chunks retournés** : tracer chaque fragment de document récupéré avec son identifiant unique, son score de similarité, l'identité de l'appelant authentifié, l'adresse IP source et l'horodatage précis pour permettre la reconstruction forensique complète a posteriori

**Rate limiting granulaire par entité** : limiter à 100 requêtes par heure par utilisateur authentifié et par clé API, avec alertes SOC automatiques dès 80 pourcent du seuil atteint et blocage temporaire de 15 minutes dès dépassement confirmé

**Canary documents stratégiquement placés** : injecter dans le corpus des documents leurres avec un contenu unique et traçable ; toute récupération de ces canaries, même partielle, déclenche une alerte critique immédiate indiquant une exploration intentionnelle non autorisée du corpus

**Modélisation du graphe d'accès documentaire** : représenter les accès aux chunks comme un graphe temporel orienté et détecter via des algorithmes de détection d'anomalies les patterns d'exploration systématique trahissant une cartographie intentionnelle

En cas d'incident confirmé, la réponse d'urgence implique dans cet ordre strict : l'isolation immédiate du vector store en mode lecture seule ou suspension des API de query, la notification du RSSI et du DPO si des données personnelles peuvent être concernées, l'audit forensique complet des logs de récupération centralisés dans Elasticsearch ou Loki sur les 90 derniers jours pour identifier l'étendue de l'exfiltration, et enfin la re-génération complète des embeddings avec une nouvelle clé de hashage pour invalider définitivement les vecteurs potentiellement enregistrés par l'attaquant. Si des données personnelles ont été exfiltrées, l'obligation de notification à la CNIL sous 72 heures s'applique conformément à l'Article 33 du RGPD. La documentation du playbook IR doit prévoir une procédure spécifique de ré-indexation avec une fenêtre de maintenance de 4 à 6 heures pour un corpus de taille standard inférieur à 100 000 documents.

## Gouvernance et Conformité Réglementaire des Systèmes RAG

Les systèmes RAG qui traitent des données personnelles sont soumis au RGPD dès lors que le corpus indexé contient des informations permettant l'identification directe ou indirecte de personnes physiques. Une analyse d'impact relative à la protection des données (AIPD) obligatoire sous l'Article 35 du RGPD s'impose lorsque le système RAG traite des données de santé, des données biométriques ou judiciaires, ou effectue une évaluation systématique à grande échelle de personnes physiques. L'AI Act européen, applicable depuis 2024 avec des obligations progressives jusqu'en 2027, impose des exigences de transparence, de documentation technique et de surveillance humaine pour certains systèmes d'IA intégrant des composants RAG utilisés dans des contextes sensibles.

La sécurisation d'un système RAG en production requiert une défense en profondeur organisée en plusieurs couches indépendantes. Au niveau infrastructure, le vector store doit être isolé dans un réseau privé sans exposition directe sur Internet, accessible uniquement via une API Gateway qui applique l'authentification forte, le rate limiting, l'audit logging et la validation sémantique des requêtes. Les outils disponibles pour implémenter ces contrôles incluent LlamaIndex avec ses modules de sécurité natifs, LangChain avec les guardrails Nemo Guardrails, et des solutions commerciales comme Lakera Guard pour la détection de prompt injection et de tentatives d'exfiltration en temps réel. Une revue trimestrielle de la configuration de sécurité du système RAG, incluant des tests d'intrusion spécifiques aux vecteurs d'attaque documentés, doit être intégrée dans le cycle de vie de l'application et tracée dans le registre des traitements RGPD pour démontrer l'accountability requise par l'Article 5.2.

## Bonnes Pratiques Opérationnelles pour la Sécurité RAG en Entreprise

La mise en place d'une gouvernance opérationnelle de la sécurité RAG en entreprise repose sur quatre piliers complémentaires. Premièrement, la classification des données avant indexation : chaque document doit recevoir un niveau de sensibilité (public, interne, confidentiel, secret) qui détermine les contrôles de sécurité appliqués à son embedding et aux chunks récupérés lors des

requêtes. Deuxièmement, la gestion du cycle de vie des données dans le corpus RAG : les documents expirés, révoqués ou modifiés doivent être mis à jour dans le vector store dans un délai défini contractuellement, typiquement inférieur à 4 heures pour les documents critiques. Troisièmement, la traçabilité des sorties du LLM : chaque réponse générée doit être journalisée avec sa provenance (liste des chunks sources) pour permettre une attribution en cas de litige ou d'incident. Quatrièmement, la formation des équipes : les développeurs intégrant des composants RAG dans les applications métier doivent recevoir une formation spécifique sur les vecteurs d'attaque propres à cette architecture, notamment le prompt injection indirect via des documents malveillants et les risques d'exfiltration via le contexte injecté dans le prompt avant génération.

## Frameworks et Standards de Sécurité Applicables aux Architectures RAG

Plusieurs frameworks de sécurité fournissent un cadre de référence pour structurer la sécurisation des systèmes RAG en production. L'OWASP LLM Top 10, publié en 2023 et mis à jour en 2025, liste les dix risques de sécurité les plus critiques pour les applications basées sur des modèles de langage, dont plusieurs s'appliquent directement aux architectures RAG : LLM01 Prompt Injection (incluant l'injection indirecte via les documents du corpus), LLM06 Sensitive Information Disclosure (exfiltration de données depuis le contexte RAG), et LLM08 Excessive Agency (actions non autorisées déclenchées par des requêtes malveillantes). Le NIST AI Risk Management Framework (AI RMF 1.0) fournit quant à lui un cadre structuré pour identifier, évaluer et traiter les risques liés aux systèmes d'IA, applicable aux systèmes RAG en entreprise. La norme ISO/IEC 42001:2023 sur le management des systèmes d'intelligence artificielle introduit des exigences spécifiques de gouvernance et de sécurité pour les systèmes d'IA déployés dans un contexte organisationnel, incluant des contrôles sur la qualité des données d'entraînement et de référence, la robustesse face aux attaques adversariales et la traçabilité des décisions automatisées. Les équipes de sécurité qui s'appuient sur ces référentiels pour structurer leur approche de sécurisation RAG bénéficient d'un cadre reconnu qui facilite la communication avec les parties prenantes non techniques et l'intégration dans les programmes de conformité existants de l'organisation.

**À RETENIR**

Le RAG étend la surface d'attaque en injectant du contenu externe dans le contexte du LLM

L'empoisonnement du vector store est l'attaque la plus impactante et la plus difficile à détecter

L'extraction de documents exploite le LLM comme proxy pour accéder aux données confidentielles

La segmentation des accès au vector store est la contre-mesure architecturale la plus importante

Le monitoring des requêtes RAG détecte les patterns d'exfiltration anormaux

## Architecture RAG et surfaces d'attaque

Un pipeline RAG standard comprend quatre composants attaquables : le module d'**ingestion** qui transforme les documents en embeddings vectoriels, le *vector store* qui stocke les embeddings et les métadonnées, le module de retrieval qui recherche les documents pertinents par similarité cosinus, et le module de génération qui combine le contexte récupéré avec le system prompt et le message utilisateur pour produire la réponse. Chaque composant présente des vulnérabilités spécifiques : l'ingestion peut être empoisonnée par des documents malveillants, le vector store peut être manipulé pour modifier les résultats de recherche, le retrieval peut être biaisé par des requêtes adversariales, et la génération peut être détournée par les instructions cachées dans les documents récupérés.

La **confiance implicite** accordée au contenu récupéré est la vulnérabilité fondamentale de l'architecture RAG. Le modèle ne peut pas distinguer un document légitime ajouté par l'administrateur d'un document empoisonné injecté par un attaquant. Les deux sont traités comme du contenu factuel de référence pour la génération de la réponse. Cette absence de mécanisme de provenance et de vérification de l'intégrité des documents constitue un défaut de conception

inhérent à l'architecture RAG actuelle que les contre-mesures doivent compenser par des contrôles externes à chaque couche du pipeline.

## Empoisonnement du vector store

L'**empoisonnement du vector store** consiste à injecter des documents contenant des instructions malveillantes dans la base de connaissances indexée par le pipeline RAG. L'attaquant peut exploiter les sources d'ingestion automatisées (crawlers web, connecteurs SharePoint, synchronisation de dossiers partagés) pour introduire des documents empoisonnés qui seront indexés et récupérés lors des requêtes utilisateur. Le document empoisonné contient une instruction cachée (texte invisible, commentaire, métadonnée) ciblant un sujet spécifique pour maximiser la probabilité de récupération lorsqu'un utilisateur pose une question sur ce sujet.

La technique d'**embedding collision** optimise l'embedding du document empoisonné pour maximiser sa similarité cosinus avec les requêtes cibles. L'attaquant génère un document dont le contenu visible est anodin mais dont l'embedding vectoriel est proche des requêtes qu'il souhaite cibler, garantissant que le document sera récupéré par le retriever pour ces requêtes spécifiques. Cette technique nécessite la connaissance du modèle d'embedding utilisé (souvent OpenAI text-embedding-ada-002 ou Sentence-BERT) mais pas un accès direct au vector store, rendant l'attaque réalisable par un attaquant externe avec un accès limité aux sources d'ingestion.

## Extraction de documents confidentiels

L'**extraction de documents confidentiels** exploite le LLM comme proxy pour accéder à la base de connaissances sans les contrôles d'accès appropriés. Dans la plupart des déploiements RAG, le retriever utilise un compte de service unique avec accès à l'intégralité du vector store, sans filtrage par rôle ou par niveau d'habilitation de l'utilisateur final. Un utilisateur avec des droits limités peut poser des questions ciblées pour obtenir des extraits de documents auxquels il

n'aurait pas accès directement, exploitant le LLM comme intermédiaire non autorisé pour contourner les politiques d'accès aux données.

Les techniques d'extraction incluent les **requêtes directes ciblées** (« quelles sont les conditions du contrat avec le client X ? »), les requêtes par inférence (« résume les documents contenant des montants supérieurs à 1 million d'euros »), et l'extraction exhaustive par itération (poser des questions systématiques sur chaque sujet pour reconstituer progressivement la base de connaissances). La contre-mesure principale est la *segmentation du vector store* par rôle utilisateur avec des collections ou namespaces séparés et un filtrage des résultats de recherche basé sur les droits de l'utilisateur authentifié avant injection dans le contexte du modèle.

| Vecteur d'attaque RAG | Prérequis | Impact | Contre-mesure principale |
|---|---|---|---|
| Empoisonnement vector store | Accès source d'ingestion | Critique | Validation documents à l'ingestion |
| Extraction documents privés | Accès utilisateur standard | Élevé | Segmentation vector store par rôle |
| Embedding collision | Connaissance du modèle d'embedding | Élevé | Diversité des résultats de recherche |
| Prompt leaking via contexte | Accès utilisateur standard | Moyen | Filtrage des réponses générées |
| Manipulation résultats retrieval | Requêtes adversariales | Moyen | Reranking et validation croisée |

## Contre-mesures architecturales

La **sécurisation du pipeline d'ingestion** constitue la première ligne de défense contre l'empoisonnement. Le scanning antivirus et anti-malware des documents avant indexation détecte les payloads connus. L'analyse statique du contenu identifie les instructions cachées (texte invisible, métadonnées suspectes, commentaires PDF). La validation de provenance vérifie l'origine et l'intégrité des documents via des signatures numériques. Le contrôle d'accès à la source d'ingestion limite les acteurs autorisés à ajouter des documents dans la base de connaissances pour réduire la surface d'attaque à l'empoisonnement.

La **segmentation du vector store** par tenant, rôle ou niveau de classification est la contre-mesure la plus efficace contre l'extraction non autorisée de documents. Chaque utilisateur ou groupe d'utilisateurs ne peut requêter que les collections auxquelles ses droits d'accès lui donnent

accès. Le filtrage des résultats de recherche avant injection dans le contexte du modèle ajoute une couche de vérification supplémentaire. Le monitoring des requêtes détecte les patterns d'extraction systématique (volume anormal de requêtes, requêtes ciblant des sujets hors périmètre professionnel) et déclenche des alertes de sécurité pour investigation.

**RETOUR DrXPÉRIENCE**

L'audit d'un pipeline RAG interne d'un cabinet d'avocats connecté à 50 000 dossiers clients a révélé que n'importe quel avocat du cabinet pouvait extraire des informations de dossiers confidentiels auxquels il n'était pas affecté. En posant la question « quels dossiers mentionnent des montants de plus de 10 millions d'euros ? », l'assistant retournait des extraits de documents confidentiels avec les noms des clients concernés. La remédiation a nécessité la création de collections séparées par équipe juridique dans le vector store Pinecone avec un middleware de filtrage des résultats basé sur l'identité de l'utilisateur authentifié.

**AVIS DrXPERT**

**Mon avis :** la sécurité des pipelines RAG est le maillon faible des déploiements IA en 2026. La plupart des implémentations utilisent un compte de service unique pour le retrieval sans segmentation par rôle, créant de facto un accès universel à la base de connaissances via le LLM. La segmentation du vector store devrait être considérée comme un prérequis non négociable pour tout déploiement RAG manipulant des données sensibles.

**Comment un attaquant peut-il empoisonner un pipeline RAG ?** +

**Le vector store chiffré protège-t-il contre l'exfiltration ?** +

**Comment sécuriser un pipeline RAG en production ?** +

## Sources et références


ANSSI — Guide de sécurité de l'IA

MITRE ATLAS — Tactiques adversariales pour les systèmes IA

NIST AI RMF — Cadre de gestion des risques IA


## Conclusion

L'architecture RAG étend considérablement la surface d'attaque des systèmes LLM en introduisant des vecteurs spécifiques que les défenses standards des modèles de langage ne couvrent pas. L'empoisonnement du vector store, l'extraction de documents confidentiels et la manipulation des résultats de recherche nécessitent des contre-mesures architecturales dédiées à chaque composant du pipeline pour garantir la confidentialité des données d'entreprise accessibles via le LLM.

Auditez la sécurité de vos pipelines RAG en production pour identifier les vulnérabilités d'exfiltration et d'empoisonnement avant qu'un utilisateur malveillant ou un attaquant externe ne les exploite. La segmentation du vector store par rôle est le premier contrôle à implémenter pour protéger vos données sensibles.


ARTICLE SUIVANT RECOMMANDÉ

**Jailbreak LLM : Taxonomie et Détection Automatisée →**

DAN, AIM, persona switch et token smuggling : taxonomie complète des jailbreaks LLM et pipeline de détection automatisée



**Découvrez mon dataset**

**rag-langchain-fr**

Dataset RAG et LangChain bilingue FR/EN

Voir →


📖 DÉFINITION

**Embedding** : Représentation vectorielle dense d'un objet (texte, image, audio) dans un espace mathématique où la proximité reflète la similarité sémantique.

💡 BON ÀSAVOIR

Pour reproduire les résultats présentés, commencez par un dataset d'entraînement de qualité et validez sur un échantillon représentatif avant tout déploiement en production.

## Sécurisez vos déploiements IA

Audit LLM, conformité AI Act, évaluation d'impact IA, Red Team IA — par un expert certifié.

**Audit IA — Devis sous 24h**

**ayi@ayinedjimi-consultants.fr**

© Ayi NEDJIMI Consultants — Tous droits réservés