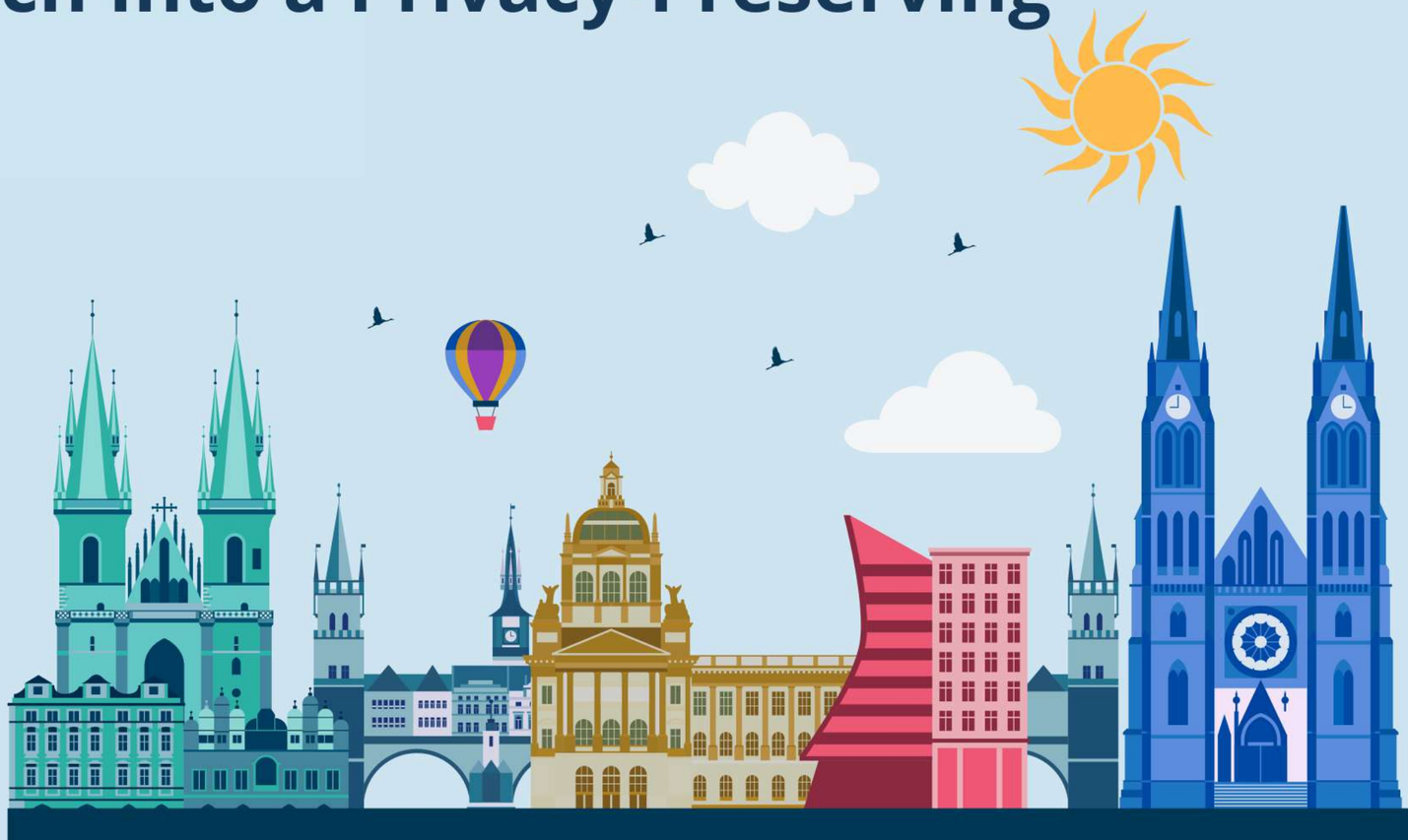
OpenSearchCon EUROPE

# The Forbidden Index:

## How We Turned OpenSearch Into a Privacy-Preserving Superhero



# The Speakers

OpenSearchCon EUROPE


AK

**Akshat Khanna**

Machine Learning Engineer-2

**Angle One**

**Unnati Mishra**

R&D Engg

**VMware by Broadcom**

# Agenda

01 The Problem: Why Search Engines Are Privacy Nightmares

02 The Regulations: GDPR, HIPAA, CCPA in Plain English

03 Our Toolkit: Tokenization, Masking & PII Redaction

04 LIVE DEMO: Privacy-Preserving OpenSearch in Action

05 Differential Privacy: The Dashboard That Lies

06 The Privacy Score Audit Tool (Open Source!)

07 Key Takeaways & Q&A

# Let's Start With a Story



A healthcare analytics startup used OpenSearch to power their patient dashboard. Fast queries. Beautiful UI. Investors loved it.

Then a researcher ran this query:

```
GET /patients/_search
{
  "query": {
    "match_all": {}
  }
```

What came back:

- Full names
- Dates of birth
- Diagnosis codes (ICD-10)
- Social Security fragments
- Home addresses

**All in plain JSON. Zero masking. Zero encryption at rest. Default config.**

# Why Search Leaks Your Secrets



**OpenSearch / Elasticsearch index characteristics that cause risk:**

- *Full-text indexing:* every word becomes searchable by default
- *Inverted index structure:* data lives in multiple places simultaneously
- Every word and ID becomes searchable
- If the “term” is an SSN, it’s instantly findable

# The Three Vectors of Exposure



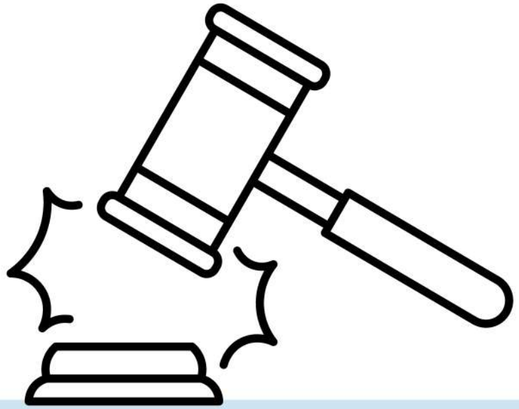
**The Three Vectors of Exposure:**

- Direct dump – match_all exposes every field
- Aggregation inference – “anonymous” stats pinpoint people
- Re-identification – combining harmless fields reveals identities
- Logs + metrics indexed too → you log the breach as it happens

# GDPR (Europe)

GDPR – for search infrastructure

- **Right to erasure:** delete PII everywhere, including indexes and logs
- **Data minimization**: don't index more than you need
- **Purpose limitation**: search cluster = processing, only for stated uses



# HIPAA
# (US Healthcare)

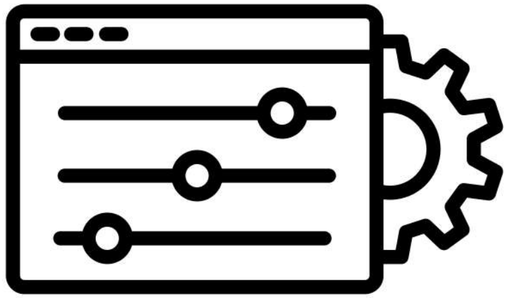
HIPAA – when search touches health data

- 18 PHI identifiers must be removed to call data “de-identified”
- Audit trails required for every PHI access
- BAAs needed for search vendor / cloud handling PHI

## CCPA (California)

CCPA – your logs are “data” too

- Consumers can see what you collect and how you use it
- Opt-out of “selling” / sharing personal data
- Fines up to $7,500 per intentional violation



# The Privacy Stack: Four Layers of Defense

OpenSearchCon EUROPE

## Layer 1

**Ingestion Gate**

PII detection before data enters the index

## Layer 2

**Tokenization Engine**

Replace sensitive values with reversible tokens

## Layer 3

**Dynamic Data Masking**

Role-based field visibility at query time

## Layer 4

**Differential Privacy Layer**

Noise injection on aggregation outputs

# Tokenization: Replacing Secrets With Safe Aliases

OpenSearchCon EUROPE

**What is tokenization?**

Imagine replacing your credit card number with a random code. The code is useless to attackers, but your system knows how to look up the real value when needed.

**In OpenSearch context:**

- Before indexing: SSN '123-45-6789' → token 'TKN_9xK2mP'
- The token is stored in the index
- The mapping (token → real value) lives in a SEPARATE encrypted vault
- Only authorized services can detokenize


# Tokenization: Replacing Secrets With Safe Aliases



**Implementation:**

```
// Pre-ingest tokenization hook
function tokenize(field, value) {
 const token = vault.store(field, value, ttl=90_days);
 return token; // Only this enters OpenSearch
}
```

**What this protects against:**

- Full-dump attacks (tokens are meaningless without vault)
- Log file leakage (tokens in query logs, not raw PII)
- Cross-index re-identification


# Dynamic Masking: Role-Based Field Visibility



**The same query, from different users, returns different data based on their role.**

Example - querying patient records:

**Analyst role sees:**

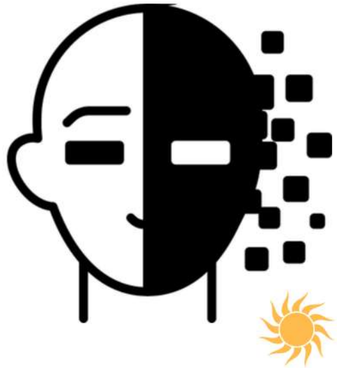
```
{ "name": "J*** D**", "dob": "****-**-1985", "diagnosis": "HIDDEN" }
```

**Doctor role sees:**

```
{ "name": "John Doe", "dob": "1985-03-14", "diagnosis": "E11.9" }
```

**Auditor role sees:**

```
{ "name": "J*** D**", "dob": "****-**-1985", "diagnosis": "Diabetes" }
```



# Auto-Redact Before It Ever Gets Indexed



The problem with relying on upstream teams:
**You can't guarantee every data producer will sanitize PII. The index is the last line of defense.**

**OpenSearch Ingest Plugin does:**

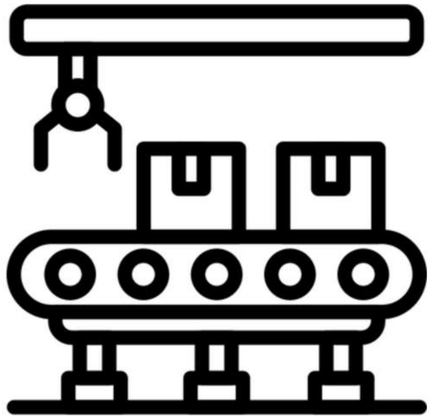
- Scans every document at ingest time using regex + ML-based NER
- Detects: emails, phone numbers, SSNs, credit cards, IP addresses, names, dates-of-birth
- Options per field: REDACT (remove), MASK (replace with ***), TOKENIZE (vault it), FLAG (alert + allow)

# Auto-Redact Before It Ever Gets Indexed

OpenSearchCon EUROPE

**Plugin config snippet:**

```
"pii_policy": {
 "email": "TOKENIZE",
 "phone": "MASK",
 "ssn": "REDACT",
 "free_text_fields": "NER_SCAN"
}
```



**Performance impact:**

~8ms overhead per document at ingest (benchmarked on m5.xlarge)

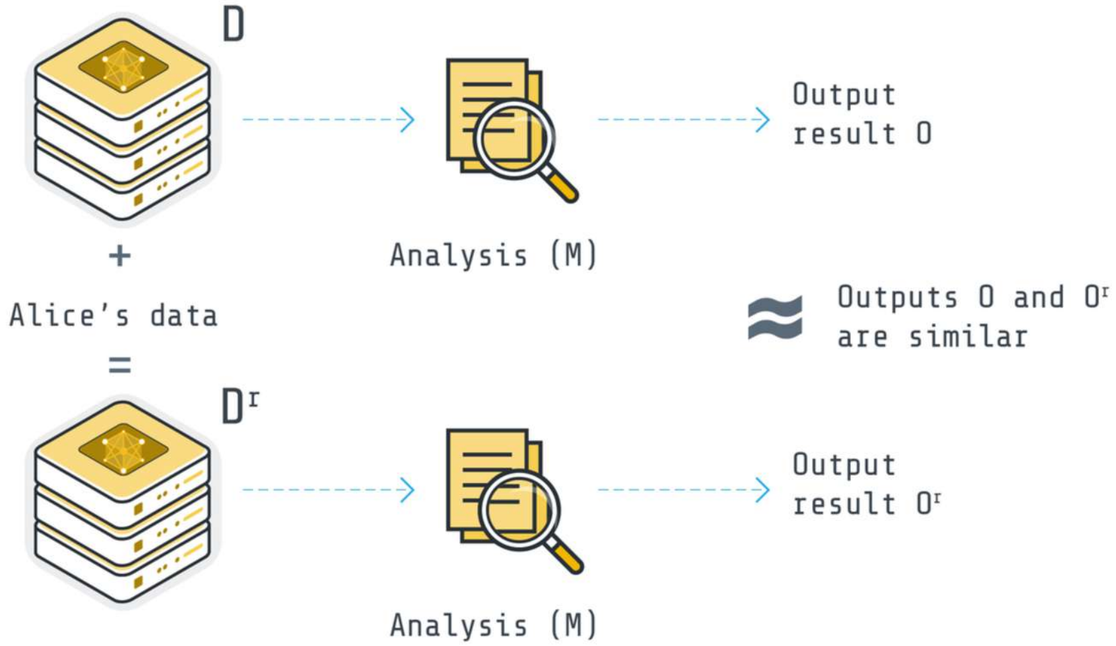
Async mode available for high-throughput pipelines

# LIVE DEMO: Let's Break It, Then Fix It

# Differential Privacy: The Math Made Human



- A query system is differentially private if the output looks essentially the same whether or not any single individual's data is included.
- In other words: An attacker who knows everything about everyone EXCEPT you cannot learn meaningfully more about you from the output.
- Epsilon (ε) trades privacy for accuracy




# Differential Privacy: The Math Made Human



**Real-world adoptions:**

- Apple: Uses DP for keyboard analytics on iOS
- Google: RAPPOR protocol for Chrome usage stats
- US Census Bureau: 2020 Census used DP to protect respondents

**Our implementation adds DP to OpenSearch aggregations via:**

- A post-processing hook on the aggregation pipeline
- Configurable epsilon per query type
- Privacy budget tracking per user session

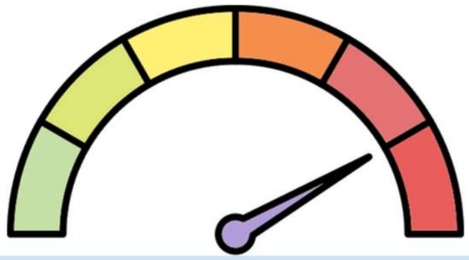
# The Privacy Score Audit Tool

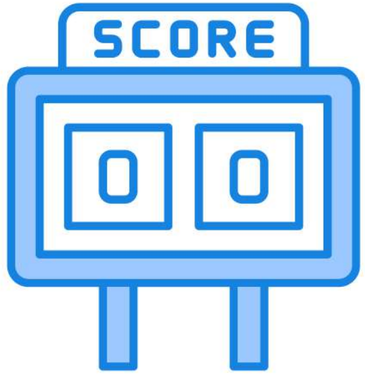
OpenSearchCon EUROPE

**An open-source CLI + dashboard that audits any OpenSearch index and gives it a privacy score from 0-100.**

**It scans for:**

- PII fields (via regex + NER model)
- Index mapping exposure (are sensitive fields stored as keyword/text?)
- k-anonymity violations in existing data
- Aggregation leakage potential
- Access control gaps (are field-level permissions set?)
- Audit log coverage (are all queries being logged?)






# The Privacy Score Audit Tool

OpenSearchCon EUROPE

## Sample output:

Untitled-1

```
Privacy Score: 34/100 ⚠️ HIGH RISK
Critical Issues: 3 | Warnings: 7 | Info: 12
→ Field 'ssn' stored as keyword — REDACT or TOKENIZE
→ No field-level security on 'diagnosis' — APPLY MASKING RULES
→ k-anonymity = 2 on 'zipcode+age+gender' — BELOW THRESHOLD
```

# The Privacy Score Audit Tool

OpenSearchCon EUROPE

**Available at:**

**https://github.com/khannakshat7/opensearch-privacy-score**

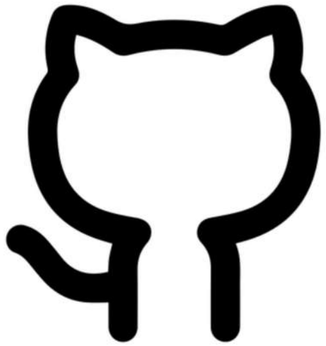
# Key Takeaways

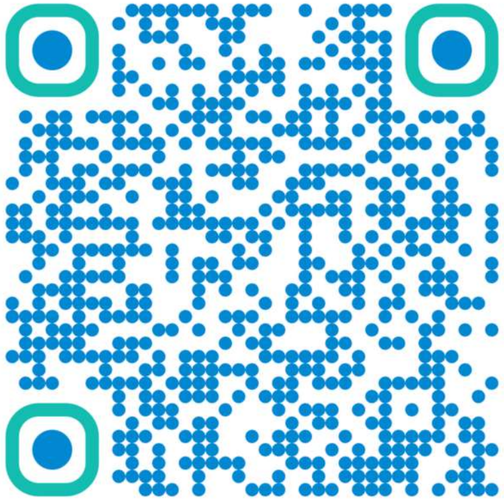


1. Your index is probably leaking PII → run the audit tool tonight.
2. Anonymization is a myth → you need differential privacy, not just masking.
3. Defense in depth wins → tokenization + masking + plugin + DP, not any one layer.
4. Privacy is a legal and ethical duty → your logs and indexes can be compelled by a court.

# Resource Links

OpenSearchCon EUROPE





**Scan for all Resources**

# Do you have any questions?



Thank you
so much!

# Let's Connect!

OpenSearchCon EUROPE

AK

## Akshat Khanna

akshatkhannatech@gmail.com

linkedin.com/in/akshatkhanna7

@khannakshat7

## Unnati Mishra

unnaticse2019@gmail.com

linkedin.com/in/pingunnatimishra

@ping_Unnati


OpenSearchCon
EUROPE