CONF42 DEVSECOPS 2024

Manage Alert Overload with AIOps

Dec 5th, 2024

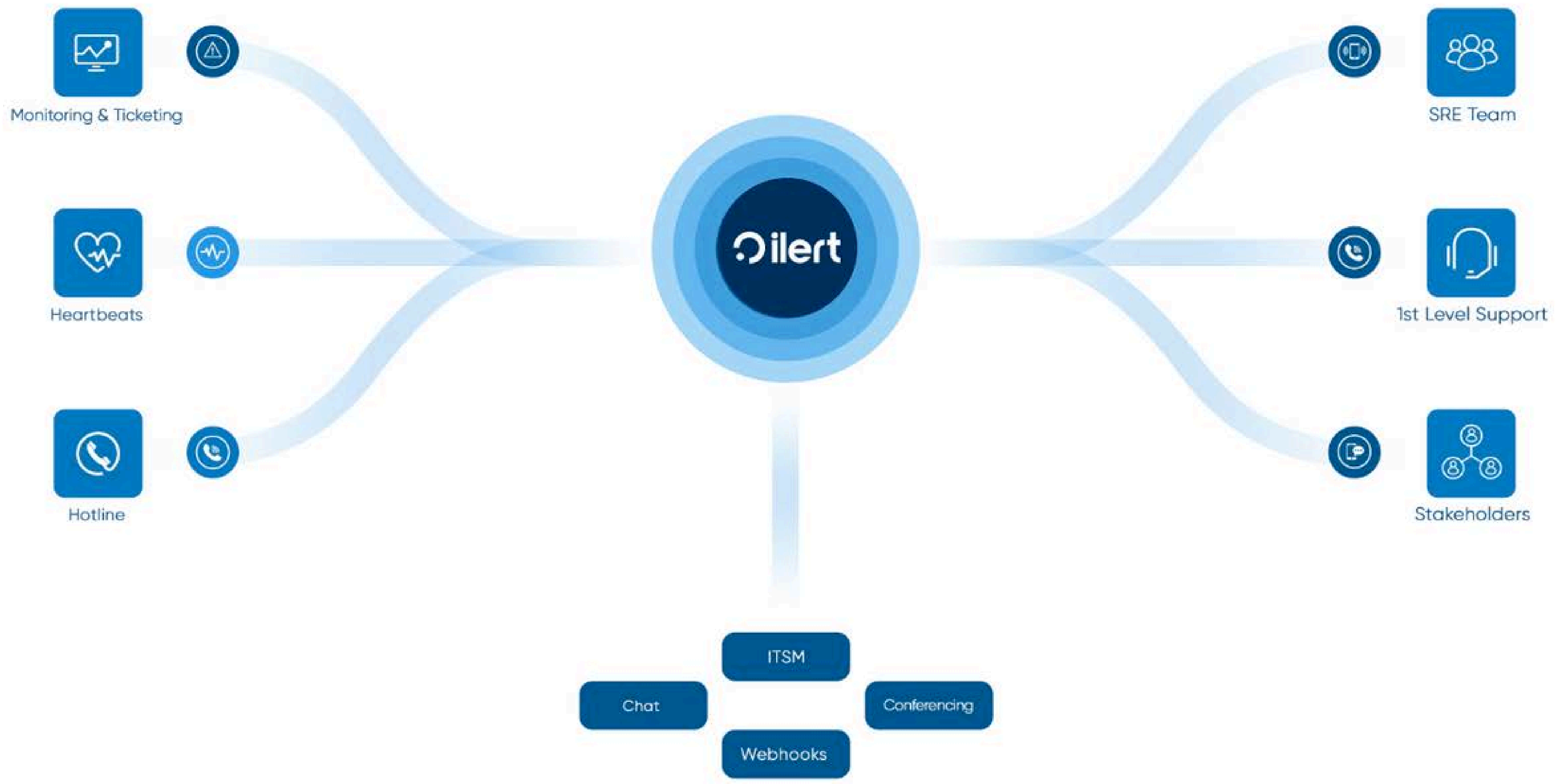
Speaker: CEO & Co-founder Birol Yildiz

Birol Yildiz

ilert

About ilert

Alerting - On-call Management - Status Pages



Used by:

- ☑ DevOps & SRE
- ☑ IT Ops
- ☑ IOT
- ☑ MSPs
- ☑ ITSM

Used to:

- ☑ Reduce MTTR & MTTA
- ☑ Increase Productivity
- ☑ Reduce Costs

Trusted by:

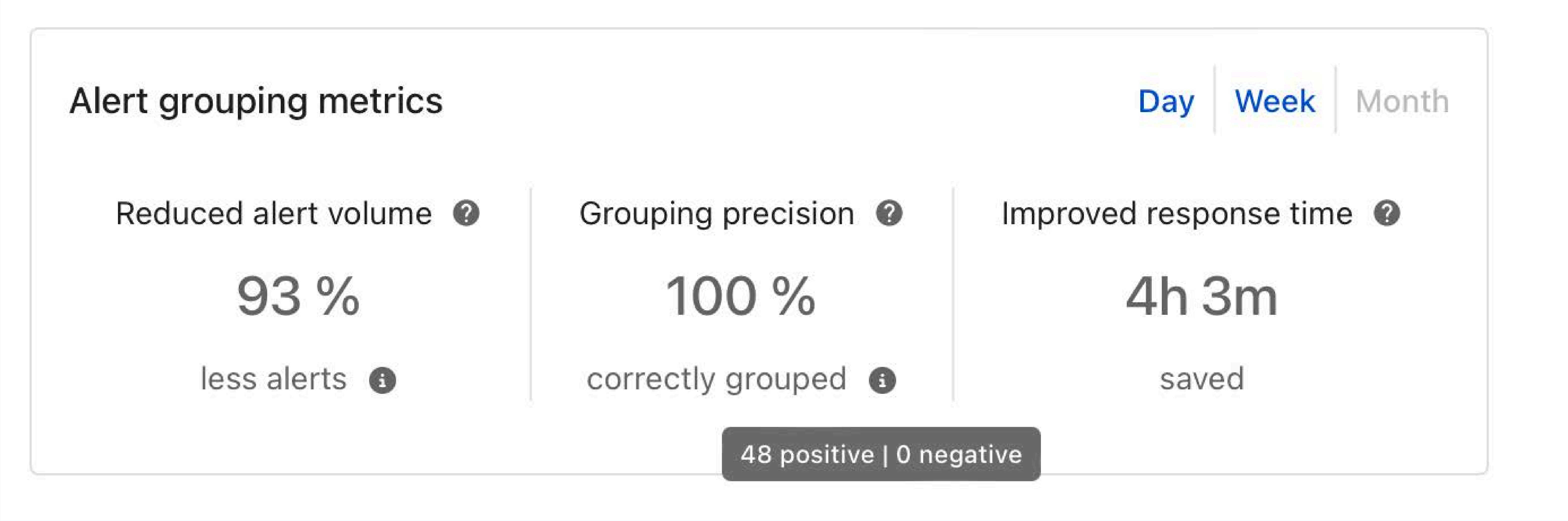
REWE GROUP · IKEA · NetCologne · airmalta · NTT Data · BERTELSMANN · Handelsblatt MEDIA GROUP · BECHTLE · DATAGROUP IT Services + Solutions

Let’s start with the why:

Why do we need to effectively manage alert overload?

- ⊗ High MTTR & MTTA
- ⊗ Decreased Efficiency
- ⊗ Increased Stress

First results after two weeks - 93% less alert volume



AIOps

Across the Stages of Incident Response Lifecycle

AI across the stages of incident response lifecycle

Prepare

Using AI Assistants for On-call Scheduling

Respond

Manage Alert Overload with AIOps

Communicate

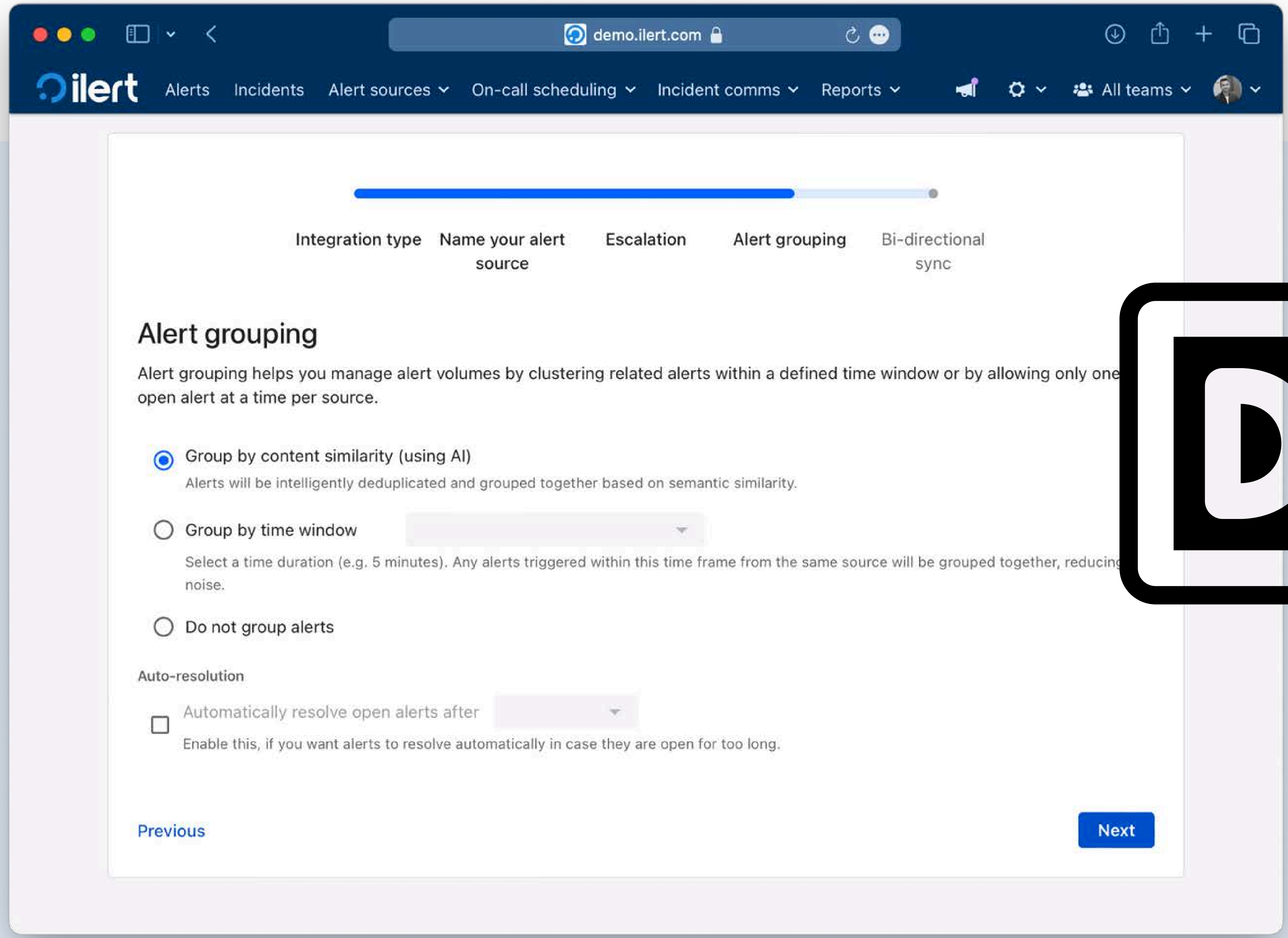
Enhancing Incident Communication with AI

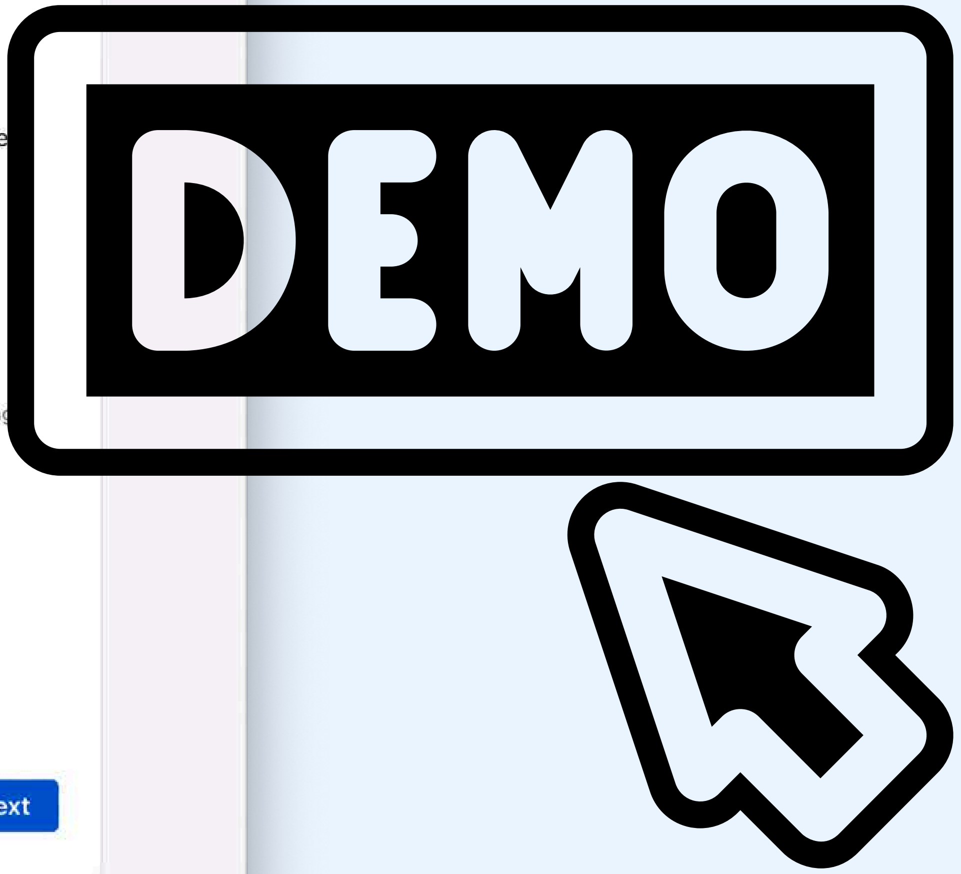
Learn

Leveraging AI for Postmortem Analysis

RESPOND

Alert Grouping in ilert





Alert Deduplication Process

1 Identify alerts that refer to the same issue.

2 Consolidate them into a single alert.

☑ Reduce noise

☑ Prevent burnout

Using Embeddings Similarity Search for Deduplication

We will work with an approach based on vector embeddings and the use of pre-trained models. To begin, we'll explore the necessary concepts for this method.

Vector Embeddings

A mathematical representation of data in a high-dimensional space, where each point (or vector) represents a specific piece of data, such as a word, sentence, or an entire document.

These embeddings capture the semantic relationships between data points, meaning that similar items are placed closer together in the vector space.

When you use ChatGPT, for example, your prompts are transformed into a series of numbers first (a vector). Similarly, we will transform alerts into vectors using an embedding model.

Embedding Model

A type of machine learning model that learns to represent complex data, such as words, sentences, images, or graphs, as dense vectors of real numbers in a lower-dimensional space.

```
// Input
"A sentence like this will be transformed into a series of (thousands) numbers"

// Output
[
 -0.006929283495992422,
 -0.005336422007530928,
 -4.547132266452536e-05,
 -0.024047505110502243,
 ... // thousands more numbers
]
```

OK, but how can we use this for alert deduplication?

We will transform alerts into vector embeddings using a text embedding model. By comparing these vectors, we identify and deduplicate alerts that are semantically similar, even if they do not match exactly on a textual level.

- *Step 1:* Preprocessing Alerts
- *Step 2:* Vectorization / Generating Text Embeddings
- *Step 3:* Deduplication Logic
- *Step 4:* Feedback Loop

Step 1: Preprocessing Alerts

Normalization

Standardize the format of incoming alerts to ensure consistency. If you're using an alerting system like ilert, which sits on top of multiple alert sources and observability tools, alerts are already normalized into a common format.

Cleaning

Remove irrelevant information or noise from alerts, such as timestamps.
Use plain text and avoid markdown or JSON. This will not only reduce the number of tokens used, but will also exclude that the format will account for deduplication.

Step 2: Vectorization / Generating Text Embeddings

Text Embeddings Model Selection

Choose an appropriate text embeddings model that can convert alert messages into numerical vectors. Models like BERT, OpenAI’s text embeddings, or Sentence-BERT (specially designed for sentence embeddings) can be suitable.

Vectorization

Each incoming alert is transformed into a vector using the selected model and stored in a vector database. Models trained on large datasets, including natural language text, can capture a wide range of semantic meanings, making them suitable for encoding the information contained in alerts.

Step 3: Deduplication Logic

Threshold Setting

A threshold is set to determine when two alerts are considered duplicates. If the similarity score between an incoming alert and any existing alert exceeds this threshold, the alerts are considered duplicates.

Deduplication and Clustering

When two alerts are identified as duplicates, they are consolidated into a single alert record, with a counter to indicate the number of duplicate alerts received.

Optional Summary Generation

Use a GenAI model to generate concise summaries for clusters of duplicate alerts. This step can aggregate the key information from multiple alerts into a single, easily digestible notification.

Step 4: Feedback Loop

Advantages

- Semantic Understanding
- Flexibility
- Scalability

Considerations

- Model Selection
- Threshold Tuning
- Continuous Learning

Grab a copy of our AI & Incident Management Guide!

Questions?

SCAN ME