How Bluesky Transformed Their Safety Operations with ROOST Osprey

February 2026

Overview

Fraud, abuse, and harmful content are rampant across digital platforms, in part due to the scaling power AI gives malicious actors. The challenge of combatting abuse is only compounded when a platform is experiencing explosive user growth, as seen in the case of Bluesky, which more than doubled their user base in a single year, from ~15 million total users in November 2024 to ~39 million as of September 2025. Such scale can outpace a team's ability to respond – especially when they are hamstrung by brittle tools built for a different digital era.

Bluesky sought a new solution for their safety investigations that could support the volume and complexity of abuse they were experiencing and they anticipate in this AI-driven world. This search led to the adoption of Osprey, ROOST's open source, high-performance investigation and rules engine for addressing abuse at scale.

Since being implemented in production, Osprey has seamlessly scaled to handle Bluesky's growing volume of threats. Osprey processes over 45 million events per day, with over 100,000 daily enforcement actions based on the safety team's automated rules. Equally importantly, the analyst-friendly user interface has made investigating new threats more efficient – empowering Bluesky's safety team to spend less time fighting with their tools and more time fighting for their users. Osprey's strong performance also unlocked cost savings. Early estimates suggest a ~70% reduction in annual data storage costs compared to the Bluesky safety team's previous solutions.

> ***"I certainly cannot overstate how useful Osprey is and how many problems it fixes at scale."***
>
> *– Trust & Safety Engineering, Bluesky*

Tooling Challenges: User friction and limited customization

Bluesky faced two major challenges in getting their prior rules engine, AutoMod, to operate at the scale and speed Bluesky needed.

1. **A roundtrip to engineering for analyst changes:** AutoMod is a rules engine custom-built by Bluesky to auto-enforce policy rules. AutoMod is written in the Go programming language and has no frontend user interface; any rule changes had to be committed into the AutoMod codebase in Go.

   The safety analyst team's expertise is in data analysis and investigation, so changes that analysts uncovered through their investigations meant relying on an engineering team member to implement the change. This roundtrip was an ineffective use of resources and slowed response times.

2. **Limited customization constrained investigation abilities:** To address the UI gap, Bluesky had to use third party platforms for investigations. This created two challenges: 1) a multi-vendor, fragmented tech stack to accomplish what was intended to be a cohesive workflow and 2) investigations were constrained to the pre-determined fields of the tools, limiting how "Bluesky-specific" investigations could be.

The Case for Osprey: Performance at Scale, Functionality and Ease of Use, and Efficiency Gains

Osprey was well positioned to solve both problems. Because it is both written in SML, a SQL-inspired query language designed to make rules easier to craft, and equipped with a built-in UI, safety analysts were able to do investigations, pattern matching, and rules creation independently. Rules are now "easy to understand" and "a huge step forward in readability and feasibility," according to Bluesky staff. And because Osprey's queries are user-defined, investigations can now be tailored to the specific threats that Bluesky faces within one centralized system and without sacrificing on quality.

> ***"I was impressed by how quickly engineering implemented Osprey and got it up and running."***
>
> *– Head of Trust & Safety, Bluesky*

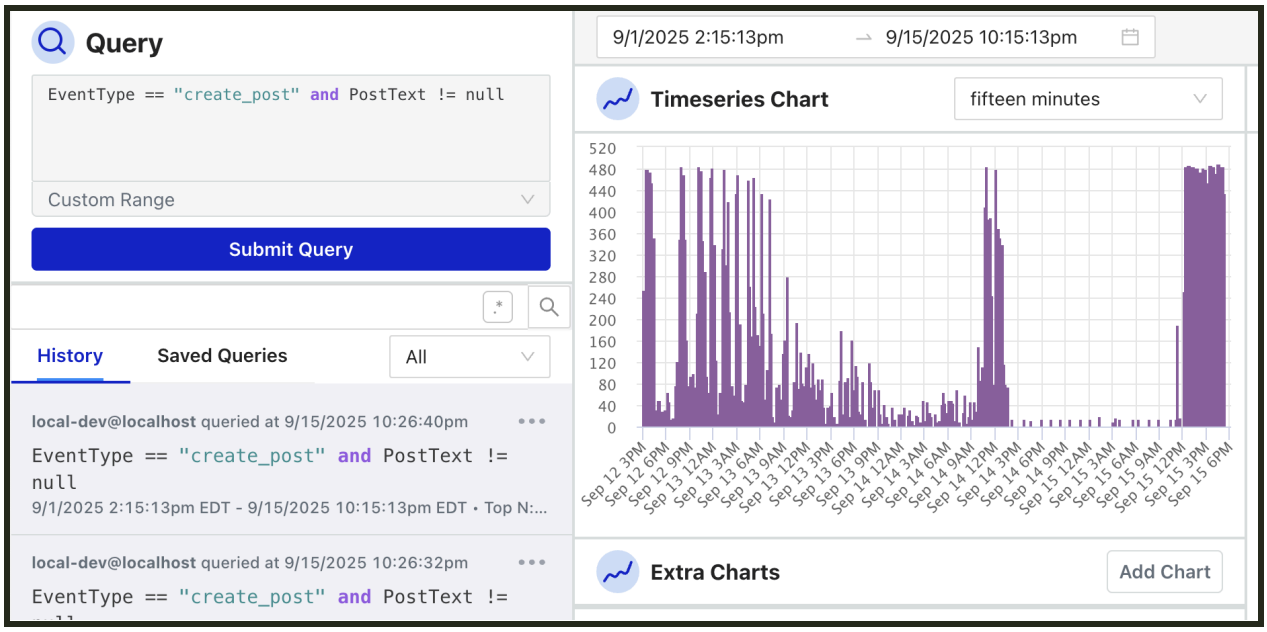

Figure 1: Osprey's built-in analysis UI allows operators to query events in real-time and visualize trends, like the burst in suspicious display name changes seen here, without leaving the platform.

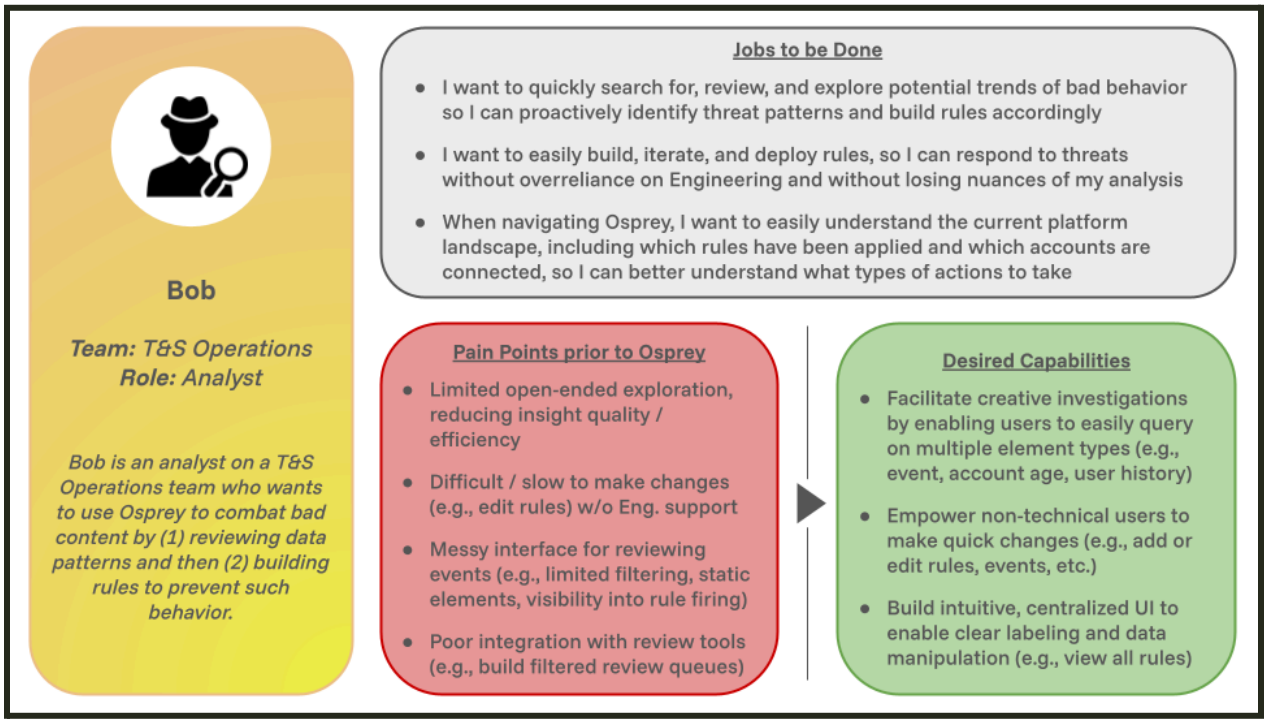

Figure 2: An example user persona of a Trust & Safety Analyst

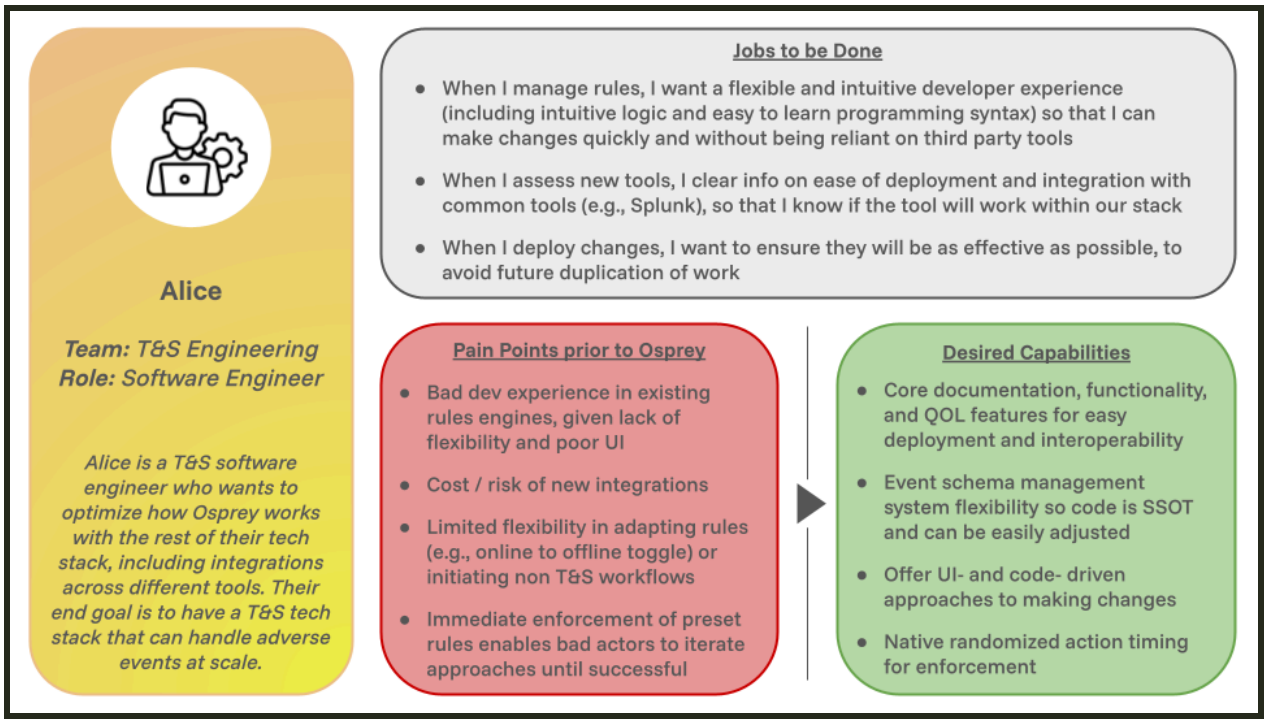

Figure 3: An example user persona of a Trust & Safety Engineer

For Bluesky, the switch to Osprey brought multiple improvements that made a tangible difference in the team's ability to fight abuse:

> ***"I really appreciate that [Osprey] is horizontally scalable…[there were] no stability issues at all. It just runs."***
>
> *– Trust & Safety Engineering, Bluesky*

- **Performance at Scale:** Osprey processes over 45 million independent events per day, and up to 5,900+ events per second. Even with a slim set of rules, this means that over 100,000 enforcement actions are processed daily. All this without any drops in stability across event and rule types, enabling smoother responses during high traffic spikes.

- **Functionality and Customization:** Osprey is designed for easy adoption, with foundational safety workflow features and clear rule-writing logic that make it easy to start using. Its analysis UI also enables flexible and powerful querying, time series charts for deeper visual investigations, bulk labeling capabilities, and a real-time event stream. These capabilities have helped Bluesky identify platform-specific threats quickly

> *"I was very impressed – everything that we had, Osprey also had…I didn't have to do any work; I just migrated [our rules] over."*
>
> *– Trust & Safety Engineering, Bluesky*

and proactively change enforcement rules, rather than waiting for users to be affected. For example, Bluesky has been able to generate 22 rules per month in Osprey, up from 13 rules per month with previous solutions – suggesting increasingly flexible and proactive responses.

- **Efficiency Gains:** The flexibility of an open source solution meant there was no limit on how Bluesky could customize the source code (and no fee required to do so), and the modularity lets teams define and optimize Osprey's deployment. That meant an estimated annual savings of over ~$70,000 in data storage costs alone for Bluesky's safety team. Osprey also improves teams' operational efficiency, saving time otherwise spent on less valuable tasks such as constructing complicated queries to launch an investigation.

An Easy Integration Process

Good tools are only good if you can actually use them. Thankfully, Osprey was designed to be "really easy for other people to come in and use." Because Osprey connects to other services via APIs and handles event data through Apache Kafka, it can integrate with codebases regardless of programming language or most typical system dependencies. Here's a simplified version of Bluesky's integration process:

- Set up Osprey to consume the atproto event stream (the "firehose") so it can ingest data
- Create Bluesky specific user designed functions (UDFs) to enable different moderation actions and enforcement outputs
- Build an Apache Kafka output sink to process, package, and share the outputs for each event
- Build an Osprey effects receiver on Bluesky's servers, which takes action in Ozone (Bluesky's moderation console) based on the received enforcement action

> *"This is a significant cost savings for us. Relying on third party vendors has been a hassle…cost wise, we're very, very happy."*
>
> *– Trust & Safety Engineering, Bluesky*

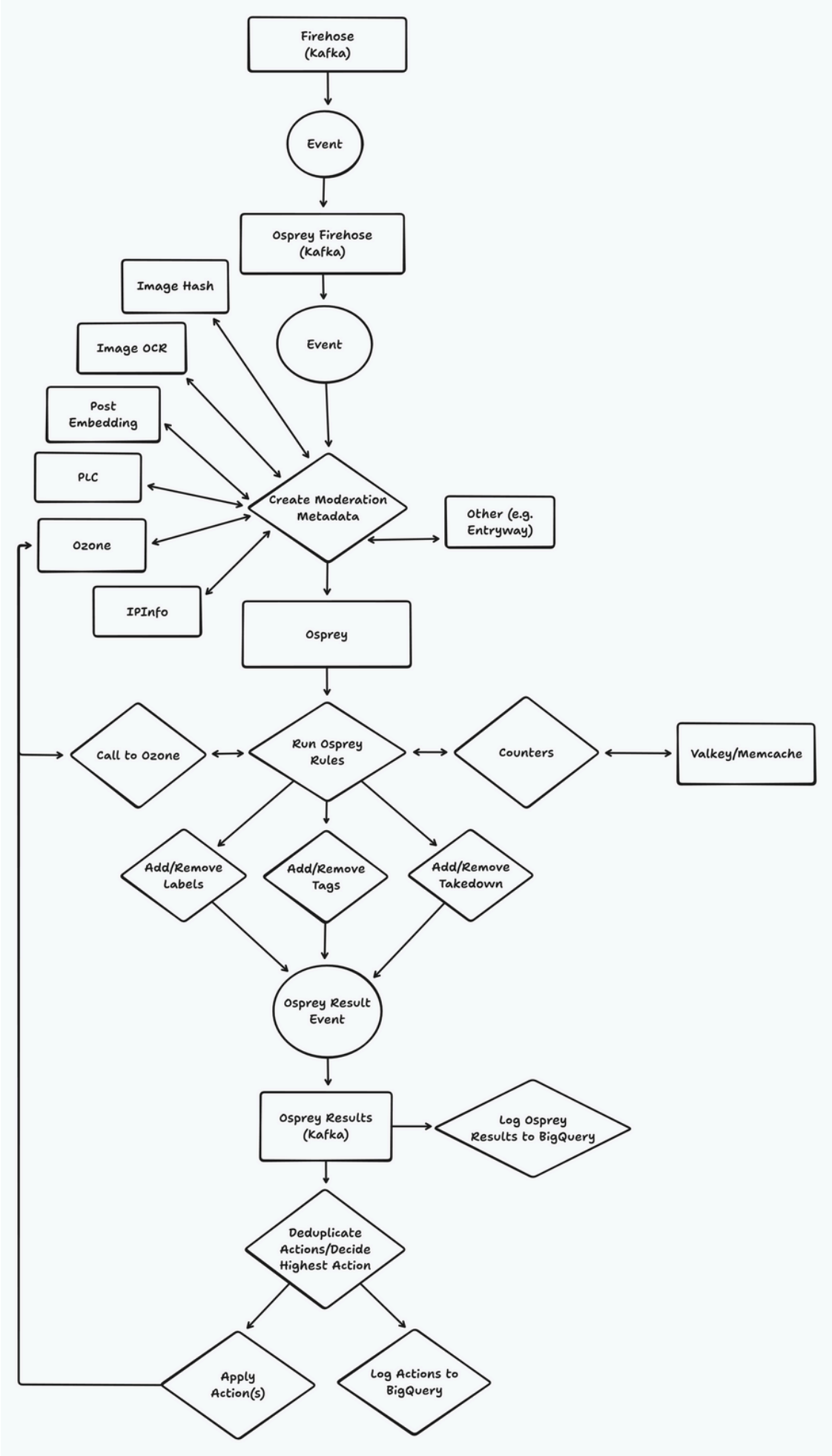

Figure 4: An overview of how Osprey fits into Bluesky's moderation architecture

Case Study Spotlight: Writing Rules to Stop T-Shirt Spam

To understand how Osprey works in practice, it's helpful to examine a rule that Bluesky has currently deployed, detailed below. Bluesky had identified that spam accounts selling t-shirts were a problem long before adopting Osprey. Because they had previously triaged it through a mix of internal systems and third party vendors, they knew it was one of the first rules they wanted to implement.

```
HRCountryTshirtSpamRule = Rule(
  when_all=[
    ListContains(
      list='high_risk_countries',
      phrases=[SignupCountry, LastSigninCountry],
    ) != None or AccountAge <= 3 * Day,
    ListContains(
      list='tshirt_spam_domains',
      phrases=ExtractDomains(s=PostText),
    ) != None,
  ],
  description='User with high risk country posting with
known tshirt spam domain',
)

WhenRules(
  rules_any=[HRCountryTshirtSpamRule],
  then=[
    AddAtprotoLabel(
      entity=Did,
      label='spam',
      comment ='User with high risk country posting with
known tshirt spam domain',
      email=None,
      expiration_in_hours=None,
    ),
  ],
)
```

Figure 5: An example rule written in Osprey to block t-shirt spam campaigns

The actual process of building and viewing this rule is straightforward thanks to SML's easy learning curve. For example, in Osprey, SML enables clear high-level variable definitions – such as "high_risk_countries." These variables can be used across all rules in Bluesky's database, reducing duplication and enabling clear syntax visibility for any given rule. This specific rule, which labels new accounts from high risk countries using a known spam domain as a spammer, is only ~20 lines long and can be easily understood by colleagues with no prior programming experience.

Case Study Spotlight: Using the Analysis UI to Identify IP Threats

For safety teams, Osprey's analysis UI can accelerate threat investigations by unlocking new ways to proactively detect threats. For example, in Bluesky's previous system, a safety analyst would not have been able to directly search for specific IP addresses that might be suspicious. Instead, they would have to search for specific event types, filter on those events for the suspicious behavior, and only then view the IP addresses associated with those filtered events.

In contrast, Osprey allows for flexible, element-agnostic querying. That means that an analyst can immediately search for a specific user account or a specific last sign in IP. From there, it is much easier to resolve individual threats (based on specific results) or identify what rules need to be written.

What's Next

Osprey is open source and available today with public working group meetings every two weeks. You can read more about Osprey's V1.0 features, and join our Discord community for questions and discussion. We're excited to see how Osprey adapts and evolves to your different needs!