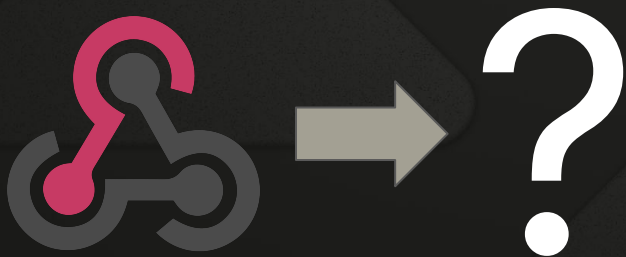


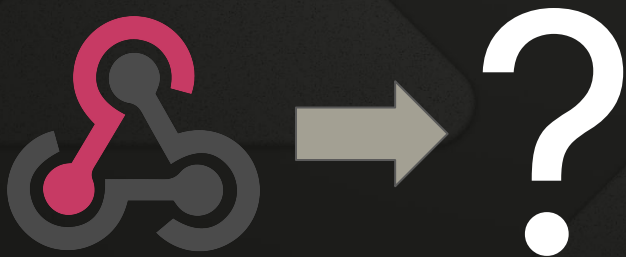
# Beyond Webhooks: The Future of Scalable API Event Delivery



**Phil @leggetter**  
**Hookdeck**

API Days / New York / 14 May, 2025

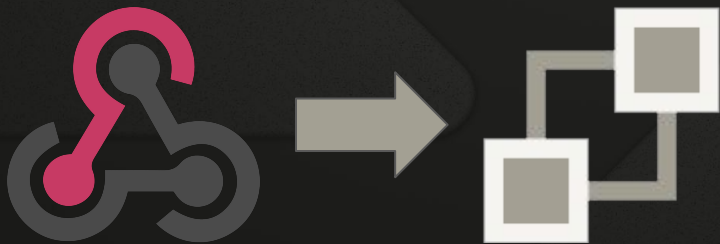
# Beyond Webhooks: The Future of Scalable API Event Delivery



**Phil @leggetter**  
**Hookdeck**

API Days / New York / 14 May, 2025

# What are Event Destinations and Why It's Time to Move Beyond Webhooks



**Phil @leggetter**  
**Hookdeck**

API Days / New York / 14 May, 2025

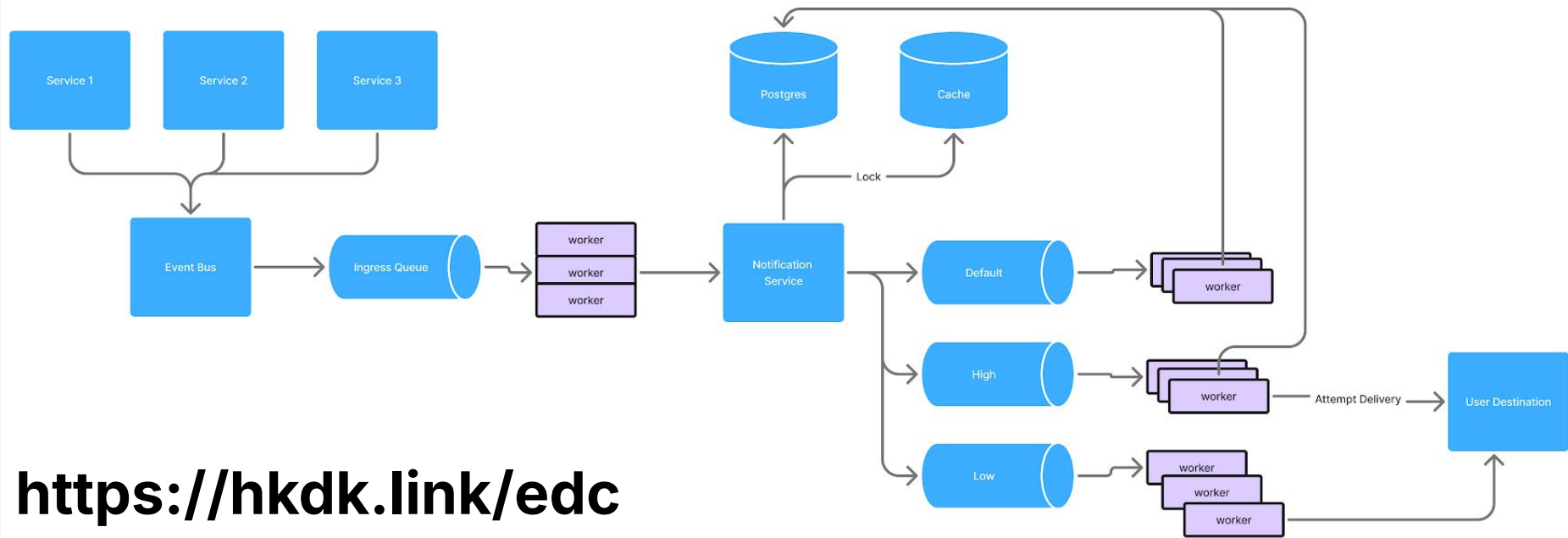
# Brief History of Webhooks



- 2007: Jeff Lindsay creates "web hooks"
- Simple HTTP callbacks for real-time notifications
- Examples: GitHub, Stripe, Shopify, Twilio

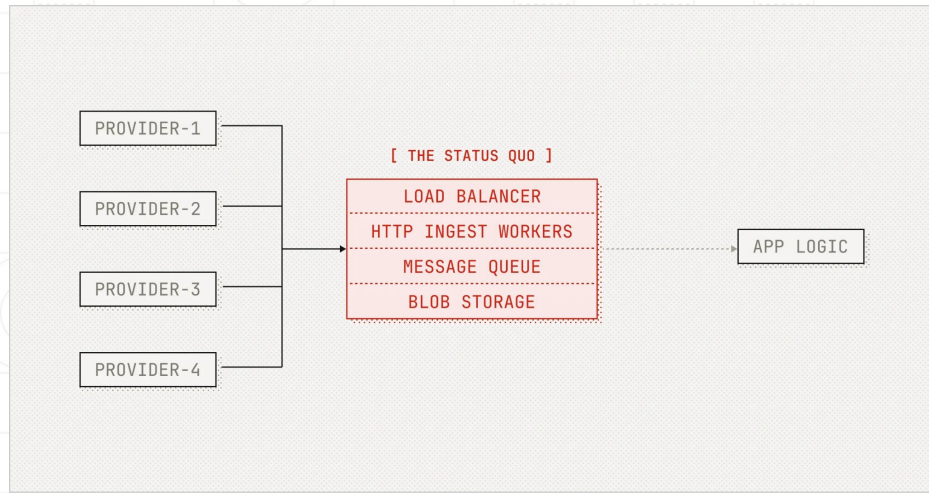
# Webhook Producer Challenges

- Scalability of Delivery
- Reliability & Retry Complexity
- Monitoring & Observability
- Customer Support Load
- Risk to Customer Infrastructure



# Webhook Consumer Challenges

- Reliable Inbound Infrastructure
- Opt-In and varied Webhook Endpoint Security
- Inbound Processing Logic: Handling Retries & Idempotency
- Debugging & Observability
- Testing & Development Workflow



# Event-Driven Architectures - Internal & External

- EDA = microservices = AWS, Azure, or GCP.
- Critical events also come from external services (e.g. Stripe, Shopify, Twilio) via inbound webhooks.
- Your architecture must account for both internal and external events.
- Why haven't we applied the same rigor, tools, and patterns to external event delivery?



The background features a light gray grid with various geometric shapes scattered throughout, including circles, squares, hexagons, and wavy lines, some of which are filled with a fine dot pattern.

# **How Leading Platforms Are Rethinking Event Delivery**



Event Streams

&gt; Getting Started

&gt; Event Types

Event Delivery Retries  
and Duplication

▼ API Reference

▼ Sink Resource

Overview

Sink Test Resource

&gt; Subscription Resource

Event Type Resource

&gt; Schema Resource

GA Migration Guide

Messaging

Voice

Serverless

Flex

Studio

All docs...

On this page ▾

**sinkType** enum<string>[Not PII](#)

The Sink type. Can only be "kinesis" or "webhook" currently.

Possible values: `kinesis` `webhook` `segment` `email`**status** enum<string>[Not PII](#)The Status of this Sink. One of `initialized`, `validating`, `active` or `failed`.Possible values: `initialized` `validating` `active` `failed`**url** string<uri>[Not PII](#)

The URL of this resource.

**links** object<uri-map>[Not PII](#)

Contains a dictionary of URL links to nested resources of this Sink.

## Create a Sink resource

POST `https://events.twilio.com/v1/Sinks`

## USE CASES

- > Discounts
- > Product merchandising
- > Purchase options
- > Global markets
- > Orders and fulfillment
- > Payments
- > Sales channels
- > B2B
- > Shopify Collective
- > Blockchain

## BEST PRACTICES

- > Performance
- Accessibility
- Localize your app
- Integrating with Shopify
- Mobile support
- Non-deceptive code
- > Compliance
- > Security

## BUILDING BLOCKS

- > GraphQL
- > Extensions
- > Shopify Functions
- ▼ Webhooks

About webhooks

## Subscribe to a webhook topic

Subscribe your app to Shopify webhook topics so that it will be alerted when an event occurs on a merchant store.

Suppose you are building a warranty pricing app that determines which warranty options a customer can add to their cart, based on the cost of an order.

When a customer is checking out, the total order cost is used to determine which warranty options a customer can select from.

In this tutorial, you'll subscribe your app to a webhook topic to be alerted whenever a new order is created.

### What you'll learn

In this tutorial, you'll learn how to do the following tasks:

- Use your [app configuration file](#) to set up a webhook subscription.
- Use cloud-based delivery methods like [Google Cloud's Pub/Sub event bus](#) to receive webhooks.
- Test your subscription is configured correctly and you are receiving webhooks.

#### Info

Shopify recommends using [Google Pub/Sub](#) as a cloud-based solution for delivering webhooks. You can also use [Amazon EventBridge](#).

In instances where you want to hand-roll your own webhooks infrastructure, you may prefer your webhooks be delivered through [HTTPS](#).

During development, you may choose to use your app's URL or external mock server sites like [webhook.site](#) and [Beeceptor](#). These are not recommended for production.

### Requirements

#### Use the latest version of Shopify CLI

You must use [Shopify CLI version 3.63.0 or higher](#) to configure app-specific webhook subscriptions.

#### Scaffold an app

Scaffold an app that uses the [Remix template](#).

#### Set up a Google Cloud console project

Set up your Google Cloud account to use Pub/Sub.

### Project

Framework: [Remix](#)

Delivery method:

✓ Pub/Sub  
EventBridge  
HTTPS

[View on GitHub](#)

✓ Pub/Sub  
EventBridge  
HTTPS

### Set your app up to receive webhooks from Google

Shopify uses cookies to provide necessary site functionality and improve your experience. By using our website, you agree to our [privacy policy](#) and our [cookie policy](#).

OK

Shopify "Webhooks"

## VERSIONING

## &gt; Changelog

Upgrade your API version

## &gt; Upgrade your SDK version

## DEVELOPER TOOLS

## &gt; SDKs

## &gt; API

## &gt; Testing

## &gt; Workbench

## ▼ Event Destinations

[Integrate with events](#)

Amazon EventBridge

## &gt; Webhook endpoint

## &gt; Stripe CLI

## &gt; Stripe Shell

## &gt; Developers Dashboard

## &gt; Agent toolkit

Stripe health alerts

Building with LLMs

Stripe for Visual Studio Code

File uploads

## SECURITY

## &gt; Security

Home / Developer tools / Event Destinations

## Integrate with events

Copy page

Send events from Stripe to webhook endpoints and cloud services.

Set up an event destination to receive events from Stripe across multiple destination types, including webhook endpoints, and [Amazon EventBridge](#).

Additionally, you can use thin events created by [API v2](#) endpoints. The SDKs for API v2 includes helpers that allow your application to fetch the latest object state from Stripe.

### Use cases

## ▼ Event Destinations

[Integrate with events](#)

Amazon EventBridge

## &gt; Webhook endpoint

You might want your applications to receive events in real-time from your Stripe account to respond and perform actions accordingly.

Receive real-time event data from your account, enabling you to run back-end

Trigger a customer confirms a payment

Initiate a refund process when a customer disputes a charge

Ensure your customers make successful recurring subscription payments

### Supported destination types

Send events to an AWS account using [Amazon EventBridge](#), or deliver them to an HTTPS endpoint through [webhook endpoints](#).

### Events overview

Stripe generates event data to keep you informed about the activity in your account.

When an event occurs, Stripe generates a new `Event` object. After your destination receives the `Event`, your app can run back-end actions (for example, calling your shipping provider's APIs to schedule a shipment after you receive a `payment_intent.succeeded` event). We provide two different types of `Event` objects:

## ON THIS PAGE

[Use cases](#)

Supported destination types

Events overview

Thin events

Prevent application errors

Example thin event payload

Example snapshot  
event payloadRetrieve additional information  
associated with a thin event

Event permissions

Event retention

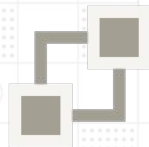
Event destination limits

Manage an event destination

Disable an event destination

# Introducing Event Destinations

- A modern event delivery model, inspired by Stripe, Shopify, and Twilio
- Supports delivery to many destination types — HTTP, queues, streams, and event buses
- Focused on reliability, scalability, and developer experience
- We've created a set of open-source guidelines have a conversation and assist adoption



[eventdestinations.org](https://eventdestinations.org)

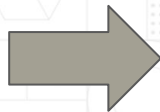
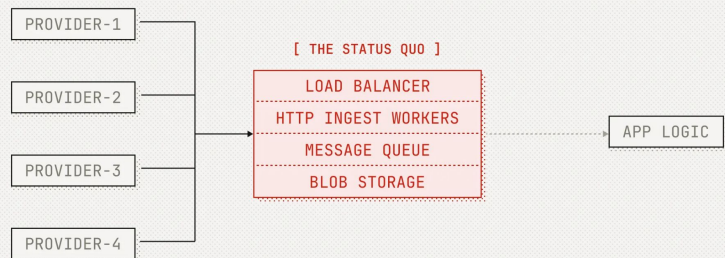
# Event Destinations: For API Platforms (producers)

- **Improved reliability:** Delivery to durable, fault-tolerant services (e.g., EventBridge, Hookdeck, Pub/Sub, Kinesis)
- **Increased efficiency:** Offloading retries, buffering, and failure handling to destination infrastructure
- **Required security:** Event destination types that require auth are an improvement on webhooks



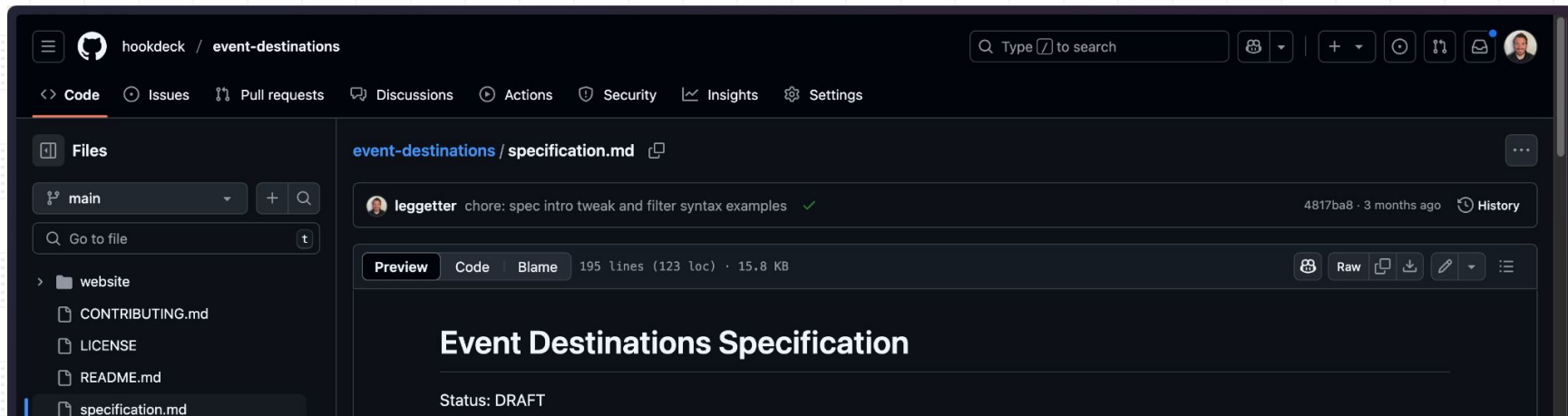
# Event Destinations: For Developers (consumers)

- **Integration flexibility:** Choose how and where to consume events — webhooks, queues, or event buses
- **Simplified infrastructure:** Remove the HTTP ingestion layer
- **Improved DX:** Better observability, fewer ingestion issues, and improved integration with existing tools



# Event Destinations: A Model Inspired by Practice

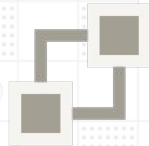
- Real-world patterns from Stripe, Shopify, and Twilio
- Delivery to Amazon EventBridge, GCP Pub/Sub, AWS Kinesis, and more
- Community-led effort to formalize and share modern event delivery guidance
- [eventdestinations.org](https://eventdestinations.org) / [github.com/hookdeck/event-destinations](https://github.com/hookdeck/event-destinations)





# Event Destinations Guidelines: Required

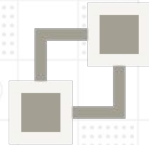
- At least two event destination types, including webhooks
- Automatic delivery retries with exponential backoff
- APIs to create, update, and delete destinations
- Alerts for destination failures



[eventdestinations.org](https://eventdestinations.org)

# Event Destinations Guidelines: Recommended

- At-least-once event delivery guarantee
- Event topics and topics-based subscriptions
- Auto-disabling of failing destinations after too many failures
- Developer UI to configure destinations & inspect events
- Manual retries via UI or API
- Filtering based on payload content



[eventdestinations.org](https://eventdestinations.org)

# When to Consider Event Destinations

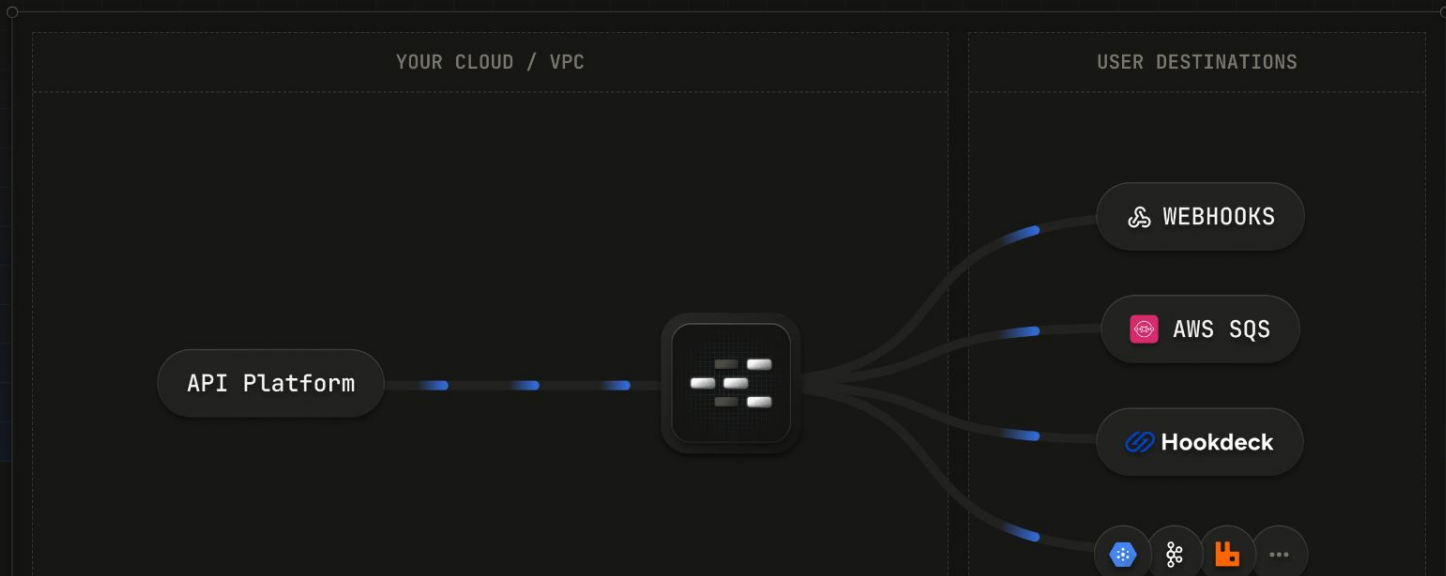
- Handling increasing scale and subscriber volume
- Experiencing frequent webhook failures or retries
- Reducing operational burden from delivery management
- Supporting customer integration needs

# Open source outbound webhooks infrastructure

Manage and deliver platform events directly to your users' preferred **event destinations**: Webhooks, Hookdeck, AWS SQS, RabbitMQ, GCP Pub/Sub, Amazon EventBridge, and Kafka.

[DOCS](#) →

[GITHUB](#) →



Outpost: Event Destinations Reference Implementation - [outpost.hookdeck.com](https://outpost.hookdeck.com)

# The Future of API Event Delivery

- Event Destinations are shaping the future of API event delivery
- Move beyond webhooks to scalable, resilient infrastructure
- Supports modern architectures and developer expectations
- Join the Event Destinations Initiative and help define what comes next
- Outpost: open-source Event Destinations implementation



[eventdestinations.org](https://eventdestinations.org)



[outpost.hookdeck.com](https://outpost.hookdeck.com)

# Thanks & Questions



**Phil @leggetter**  
**Hookdeck**

API Days / New York  
14 May, 2025

