What's Kafka and what does Confluent do?

Help with solving Kafka-esque data problems

Justin
Jun 15, 2021

avoid bugs like Gregor by using Kafka

The TL;DR

Apache Kafka is a framework for **streaming data** between internal systems, and **Confluent** offers Kafka as a managed service.

- We're dealing with a lot of data these days – Big Data™ – and recording, storing, and moving it around is **hard and expensive**
- Kafka helps **stream that data** throughout your company and **distribute it** to the systems that want to use it
- The Kafka architecture works through a **publish-subscribe** pattern
- Kafka 101 **terminology**: producers, consumers, messages, and topics

Kafka is new, but it's getting pretty popular: managed service provider Confluent, founded by the original creators of Kafka, filed their S-1 last week.

I've got two problems, that's it

Kafka exists to solve two fundamental problems facing almost every data infrastructure team at every company.

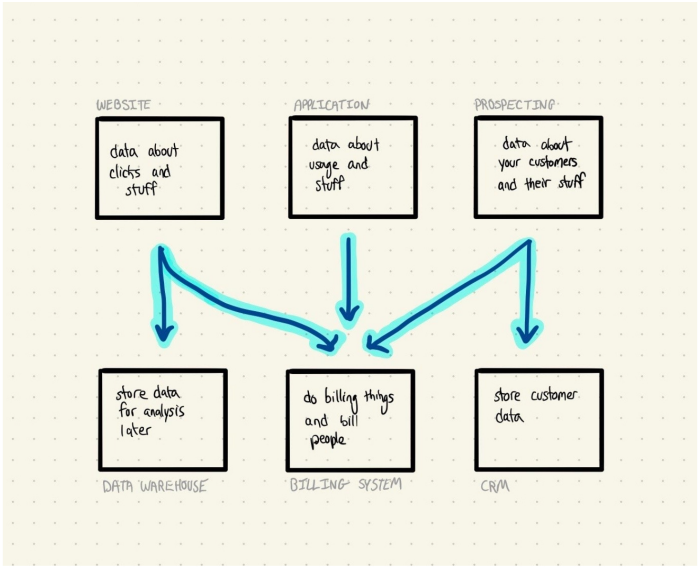
1. **There's a lot of data, and it's all happening very quickly**

As storage has gotten cheaper, we've been collecting more and more data. Most software companies record every single website visit and click, and some go even deeper. Once you have more than few users interacting with your product, you're talking about millions of different events per day. Storing and managing that **size and velocity** of data is hard.

2. **Data needs to move around to be valuable**

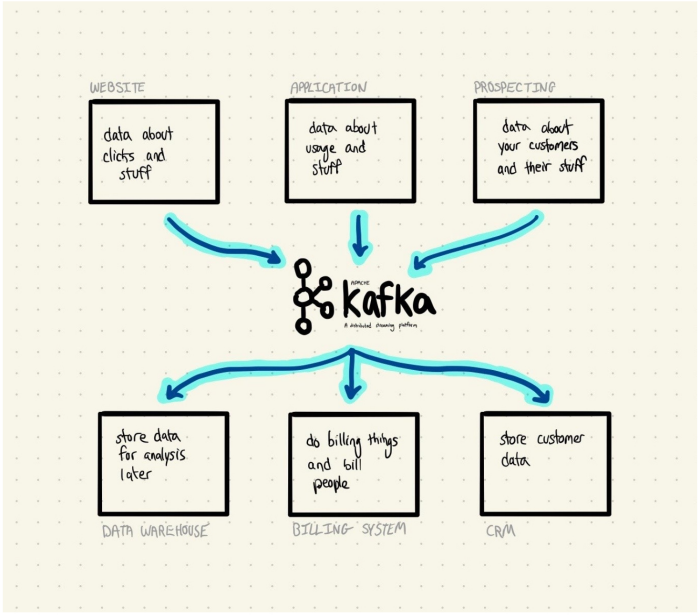
Even if you're a wiz at collecting and storing your data, there's a problem: you're going to need to move it around for it to be valuable. Where data gets initially collected and stored is rarely where it's going to be useful.

Let's use the internal systems we had at DigitalOcean as an example. When users interacted with our product, that would generate an event: `resource_created`, `resource_destroyed`, etc. Those events needed to be routed to multiple systems: our analytics team needed them in a warehouse for dashboarding, and our billing team needed them in the billing system to track and generate invoices.



Moving this data on a consistent basis and schedule is actually really hard, especially after the fact.

A few engineers were dealing with these two problems at LinkedIn back in 2011, and developed Kafka to solve them (among other things). Kafka is a **distributed messaging system** – a bunch of jargon that means that it makes it easy to collect and move data around your organization. Kafka acts as a central bus station for your data, ingesting it from different sources (your site, your app, Salesforce, etc.) and distributing it to all of the destinations it needs to get to.



Let's dive into how Kafka works. Because that's what you're here for.

Pub-sub: like a newsletter, not a sandwich

Kafka works through what engineers call a **publish-subscribe model,** often shortened to pub-sub (to save 10 milliseconds pronouncing it). It's an architecture for sharing data and tasks across an organization through messages. To understand what that means, think about starting a newsletter. But in reality, definitely don't think about starting a newsletter.

If you want to get people to read your newsletter, there are basically two ways you can go about distributing it.

you can go about distributing it:

1. **Send it when people ask for it (request-response)**

You can wait for someone to ask you for a particular issue of your newsletter, and then send it over via email. You wait for their request ("I want to read this") and then send back your response ("here's the newsletter"). In software, this is called the **request-response model**.

2. **Publish it for everyone to see (publish-subscribe)**

Waiting for people to ask for your content is silly and inefficient. Instead, what everyone does is publish it widely on the web, and then anyone who wants can sign up to get it. You *publish* ("here's the newsletter") and then others *subscribe* ("I want to read this"), hence **pub-sub**. The order of operations is reversed.

> 😇 **Don't sweat the details** 😇
>
> There's a lot more to the distinction between request-response and pub-sub, and it can be hard to understand the first time around. To keep it simple, just keep your eye on the goal of pub-sub: to make sure data gets to where it needs to go easily.
>
> 😇 **Don't sweat the details** 😇

Kafka is a system for implementing pattern #2: you publish your data to a central location (Kafka), and then destinations (a data warehouse, another app, your billing system) can *subscribe* and *consume* that data as it comes in. Data only gets *retained* for a limited period of time: those destinations need to take that data and use it, or it gets thrown out to make room for new data coming in.

One thing worth noting: Kafka is *not* the only piece of software that you can use to build pub-sub systems, and it's definitely not the first or most popular. RabbitMQ, for example, has been around for almost a decade. What makes Kafka unique is how powerful it is and how it helps shepherd data to where it needs to go.

You could also make a reasonable case that Segment is a Kafka competitor. They orient themselves towards different target customers, but the underlying tech is very similar; in fact, the Segment engineering team wrote a technical blog post about how they built what basically appears to be Kafka-lite.

> 🚨 **Confusion Alert** 🚨
>
> Kafka is a piece of software, but it's not something you just plop into your codebase and it solves your problem. You need to *orient your architecture around it* – it's a helpful tool for you to implement a specific type of data infrastructure, but it's only a piece of the puzzle.
>
> 🚨 **Confusion Alert** 🚨

Kafka 101 and terminology

Now that we've got the basics down, let's get into how people actually use Kafka. There's special terminology for the different types of system elements that we've spoken about. Because it wouldn't be software if it wasn't confusing.

- **Producer** – an application that acts as a **source** of data in Kafka.

This might be your website producing click data, or your app producing product data. Producers do the publishing, like you producing your newsletter(s).

- **Consumer** – an application that acts as a **destination** for data in Kafka.

This might be your data warehouse storing that click data, or your CRM storing customer details. Consumers subscribe, like the homies subscribing to and reading your newsletter(s).

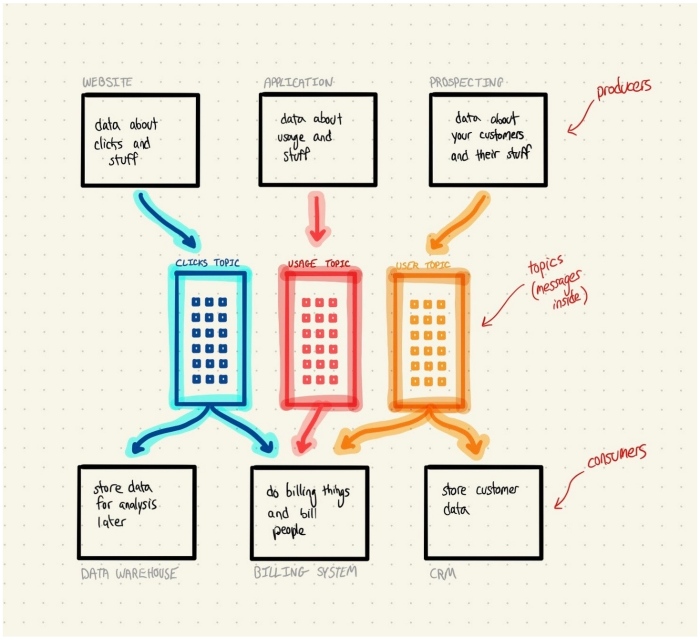
- **Topic** – an individual stream of organized data.

If you produce multiple newsletters (one called Dank Memes Weekly and another called Shitposting Quarterly), this corresponds to an individual one of them. Producers and consumers can produce and subscribe to a topic.

- **Message** – an individual unit of data.

This is the lowest level of data, and corresponds to an individual issue of an individual newsletter in our (mediocre) example.

To put it all together in a sentence: **producers** write **messages** to **topics** that **consumers** subscribe to. Phew. Part of why pub-sub is important is that multiple consumers can subscribe to a topic; that means that you only need to *send* data once, even if multiple places need access to it.



The last important thing to know about Kafka – and part of why it can be complex to set up – is that it's built as a distributed system by default. In order to handle crazy throughput - data coming in very, very quickly – Kafka is usually deployed across multiple servers in a data center. Networking these servers together and making sure they're in sync is pretty difficult, and it's part of why managed service providers like Confluent are printing money.

(bonus) what does Confluent do?

(this section is usually reserved for paid subscribers, but what the hell, have fun)

Confluent provides Apache Kafka as a managed service. What does that mean, you ask? If you're a developer, using Confluent means that you get to use Kafka without having to set it up on your own servers. Confluent *hosts* Kafka for you, and you pay on a usage basis. The company was founded by the team of engineers at LinkedIn who originally built and open-sourced Kafka, so they know what they're doing.

> 🔍 **Deeper Look** 🔍
>
> Jay Kreps, one of the Confluent founders, wrote this lengthy post explaining logs and why he thinks software is moving in an event-driven direction. It's technical, but highly worth a read if you can.
>
> 🔍 **Deeper Look** 🔍

Confluent is a classic open source story, and fits into the same category as Elastic, Redis Labs, MongoDB, etc. It goes like this:

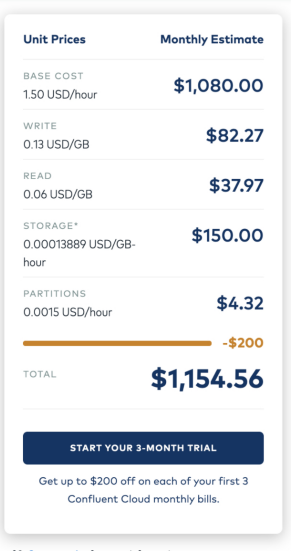
- Team of engineers builds open source framework at company
- Software is very hard to run on your own infrastructure
- They leave to found a company that commercializes said software by hosting it for you

When I say *hosting*, that doesn't necessarily mean that you're abdicating all control of your environment to Confluent – you can also self host.

In addition to managing your Kafka deployment for you, Confluent has also built a nice series of plugins (they call it a "hub") for getting data in and out of Kafka. They've also been putting muscle behind ksqlDB, a new product built on a top of Kafka that helps companies build applications that require real-time streaming (i.e. real time fraud detection).

Confluent's pricing is predictably confusing. If you're using their cloud product,

there are 5 line items you need to calculate to understand what your monthly bill is going to be. Here's a screenshot of their pricing calculator that I configured to use a standard multi-zone setup on AWS:



The absolute number here isn't that important (managed services are expensive, and that's fine) – it's more about this pricing model optimizes for flexibility over simplicity. The first time I checked out Confluent – maybe about 2 years ago or so – they were charging simply for data in and data out. Easy. But as companies move upmarket, pricing models are destined to follow. And that's why they need a calculator now.

Terms and concepts covered

`Request-response`

`Publish-subscribe (pub-sub)`

`Producers`

`Consumers`

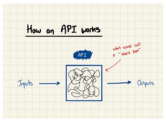
`Topics`

`Messages`

`Distributed`

Further reading

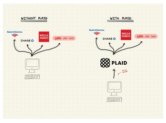
- You can get a glimpse of how popular Kafka is and which companies are using it on Stackshare
- Confluent's site has a great video explaining the thinking behind Kafka and what exactly it is
- My friend Muji wrote a great teardown of Kafka and Confluent (really in depth) for his paid subscribers here



What's an API?

What McDonalds and Lyft have in common

Justin Jan 9, 2020 187



What does Plaid do?

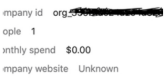
Technically begrudgingly tackles Fintech

Justin Jan 14, 2021 34 3

What does New Relic do?

Keeping an eye on your servers and apps

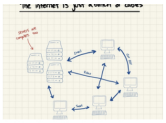
Justin Jan 11 23 2



What's Reverse ETL?

Getting your data OUT of your warehouse?

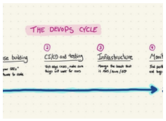
Justin Feb 1 19



What happened to Facebook?

A basic explainer of what that outage was all about

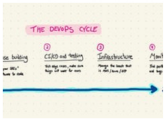
Justin Oct 5, 2021 29 4



What does GitLab do?

The TL;DR GitLab is a somewhat contrarian take on DevOps: it's basically one giant tool for literally anything you'd want to do relating to building and...

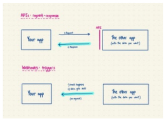
Justin Jan 4 8



What's DevOps?

IT has a cool new name

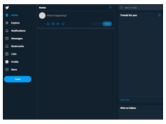
Justin Jan 5, 2021 34



What are webhooks?

Triggered

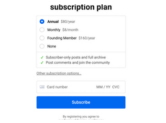
Justin Sep 13, 2021 28 5



What's Headless E-Commerce?

We may be running out of names

Justin Nov 2, 2021 20 7



I built a (basic) Substack clone in a month

This was probably a waste of my time

Justin Sep 2, 2020 50 4

See all ›