Aaron Delp (00:00.877)
Good morning, good evening wherever you are and welcome back to the Cloudcast. We're coming to you live from our massive Cloudcast studios here in Raleigh, North Carolina. And it is Aaron this week and we are going to be talking all about data pipelines and especially about Apache Airflow and some of the other solutions as well. And for that, we have Julian Leneve, CTO at Astronomer. So first of all, welcome to the show, Julian, and give everyone a quick introduction, please.

Julian LaNeve (00:28.774)
Hey, Aaron. Yeah, it's great to be here. Thanks for having me. My name's Julian. I'm the CTO of a company called Astronomer. We're a series C data infrastructure company that's built a business around this open source project called Apache Airflow. Airflow is the most popular way to write, run, deploy, manage data pipelines. It was built about 10 years ago, originally by Airbnb.

And today it's downloaded over 30 million times a month. There's about 80,000 companies out there using it. And historically, Airflow is used to power the very traditional ETL pipelines, internal dashboards. But we've seen a shift recently to where Airflow is powering more critical use cases. So they're feeding data back into the product itself.

It's power and compliance use cases, payroll, AI, ML, basically anything that has to do with moving data around, doing some sort of transformation on it. Airflow is kind of front and center there. And Airflow, because it's a distributed system, can be somewhat difficult to scale yourself. so astronomers built a business around helping companies manage and scale their Airflow deployment.

Aaron Delp (01:46.861)
Fantastic. And also too, I wanted to mention too, because I think it's always interesting when we talk about Apache projects on the podcast. We've had a number of Apache projects and it's super interesting to me this dichotomy between Apache and everything Apache has done over the years and full recognition out there. Dave McNally, kind of president of Apache, he's a good friend and going back to my cloud stack days.

Apache has done a fantastic job of staying relevant within the communities. But also to I think a lot of folks over the years have kind of struggled with that line of like, hey, what's the open source version? And then by the way, when does it cross over into the commercial version? And so tell everyone a little bit about like, I'll say compare and contrast, I do I was gonna say advantages and disadvantages, but I'll just simply say

compare and contrast like Apache versus the more commercial aspects of some of these.

Julian LaNeve (02:51.75)
Yeah, it's a great question. It's always a topic of very active discussion here at Astronomer. And I can imagine also places like Databricks, Confluent, similar business models. The way I view it,

kind of my philosophy here is that anything that's geared towards an individual developer, ease of use, like that type of functionality goes in the open source project. We want the open source project to be...

well adopted, want individual developers to love using it because that's ultimately how you continue growing the project. And we're very fortunate to have great relationships with basically everyone on the kind of Airflow project that's committing to it, that's helping maintain it. We have two engineering teams that are fully dedicated to the open source project.

We obviously invest quite a bit in airflow itself to make sure that it's continued growing, right? Because without airflow, there would be no astronomer. But to your point, we need to stand up a commercial product somehow. And the way I think about doing that is twofold. The first is, again, airflow is a distributed system. And unless you have in-house experience running distributed systems,

It's oftentimes easier just to delegate that responsibility to a third party like astronomers. This is classic managed service business. And then the second piece is anything that like helps with enterprise adoption. So for example, tying into your SSO system so that authentication and authorization is handled the same way, helping you scale out to many deployments at once.

giving you the tools that you need to observe your pipelines across all of your deployments. These things that are very helpful if you're running an organization or a full data team end up typically living in our commercial product.

Aaron Delp (04:50.189)
No, that's yeah, that's that's super helpful. Thank you very much for that. so let me also to kind of maybe set the stage for those that aren't familiar with either astronomer or airflow. When we say distributed systems and data pipelines, like what do we mean? And like, what are we typically?

I don't know if replacing is the right word, but what are we typically, typically supplementing or building? What does the stack look like? You know, kind of top to bottom when we're talking about building these distributed systems.

Julian LaNeve (05:22.618)
Yeah, I'll start with data pipelines because that's kind of the core use case of, of Airflow and some of these other orchestration tools. A data pipeline is essentially just a set of processes that you want to chain together to successfully deliver some sort of outcome. I think the most canonical example of a data pipeline is an ETL pipeline where you're extracting data from some source system, right? Salesforce here is very common. Your.

transforming that data. Oftentimes, the raw data in and of itself is not useful. You want to do some sort of grouping or calculation or inference on it. And then you're loading that cleaned

transformed data into some final destination, whether that's a data warehouse or some sort of dashboarding tool. I mean, nowadays, even applications that are user-facing.

I like working with data pipelines quite a bit because the types of use cases that data pipelines power vary so much. Right at the end of the day, it's just a set of processes that you want to chain together. And the reason you need a tool like Airflow to help you manage your data pipelines is these things oftentimes start very simple, where you have one pipeline, a handful of steps that kind of all depend on each other. But as you continue scaling out,

of your data pipelines, your use cases as you start to onboard more teams. The scheduling logic that you need gets a lot trickier. Maybe you want to wait for some data to arrive before actually running the pipeline. Or maybe whenever the first pipeline is done, you want to trigger a set of downstream pipelines. You need some level of observability into what's going on. If your pipeline fails, you want to get alerted. As your pipeline's running, you want to be able to see how it's progressing.

And you want these pipelines to be very reliable. Like I mentioned, we're seeing kind of a shift in use cases for Airflow from these internal analytics types of use cases to things that if the pipeline goes down, there's very real business implications. We've had several customers tell us that if Airflow goes down for them, they end up on the front page of Wall Street Journal and not in a...

Julian LaNeve (07:45.88)
in a good way because everybody today is building very data-driven businesses. Now, to actually run and scale those pipelines at massive scale, you need more than one component sitting there running those pipelines, helping you manage those pipelines. Airflow itself has about four or five different components. You need a database to help you track the state of all of your pipelines.

You need a scheduler that's sitting there looking at what pipelines need to run. You need a set of workers that actually do the execution work for the pipelines. You need a UI so that people don't have to like interact with the database itself. There's a, there's a handful of components that you need to run and keep up all the time for your data pipelines to continue running. And that's what I mean by distributed system, right? Is it's not just like one machine that you have someplace that

As long as that thing remains up, your pipelines are good. You end up scaling it out to many different machines so that, for example, as you start to introduce ML pipelines that require GPUs or a lot more compute, you can go scale that independently of the rest of the components. And these distributed systems can be hard to run and manage, especially at scale, which is why oftentimes folks that run Airflow

will turn to someone like Astronomer to help them do it. There's no business value in just keeping airflow running. The value comes from what you're actually able to build on top of it. So

if you come delegate that responsibility to us, we have the most reliable and efficient airflow service in the market. That means that you get to focus on actually building and deploying those pipelines that power new use cases.

Aaron Delp (09:38.061)
Yeah, actually you you

Kind of went ahead to where my next question was going to be was a little bit talking about managed services and the advantages of managed services, because I feel like that's something that's really become very universal, especially in the open source side of the house of, yeah, you could operate all of this yourself, but a couple of different things. A, do you have the expertise in house or, you know, do you have to train up those people? And then also just kind of the overhead of something like that. And does that actually provide

business value. Right. And and so I think that that's a really good thing for everyone out there when we're talking about some of these managed services in particular, to consider. Now, maybe if I could double click into that one step further, though, because I kind of see, you know, let's let's break up DevOps for a second, right? There's there's advantage to the developer side of the house, which I think we've talked about a little bit. And then there's, know,

operations and speed and velocity kind of things. But is there more to it than that of like, okay, the developers get, you know, very specific things that they no longer have to do on a day to day basis. So the operations folks have things they just don't have to think about anymore. Like, let's double click on that just a little bit more with the managed services piece of all of that.

Julian LaNeve (10:57.818)
Yeah, absolutely. mean, that's the goal of any managed service, right, is take away as much of that operational responsibility as possible. Things like keeping the infrastructure running, making sure that it scales as it needs to, making sure that as things go wrong, like you have someone sitting there troubleshooting, ensuring that like airflow itself is running well, that if you run into issues with like things that are airflow specific, you're able to go resolve them.

It's not like very undifferentiated toil, I like to call it because like, again, no, it's the type of work that that it's the type of work that like no one wants to do. That delivers no value. And you only think about it when things are actively going wrong. And that's exactly why, you know, we we had astronomer exist, we've run airflow for now thousands of companies across the world.

Aaron Delp (11:32.439)
I love that word. I agree. I love it.

Julian LaNeve (11:54.296)
We obviously like to think we're quite good at it. Again, we have the most reliable Airflow service there is. And so if you want to go delegate that responsibility of operating Airflow to someone else so that you can free up your data engineers time, your software engineers time to actually

build, deploy and monitor these pipelines, like you want to be able to give that operational responsibility to someone else.

Now, not everybody starts with a managed service, right? Like, they can sometimes be expensive and only worth paying for when you need them. And that's the beauty of having this kind of open source and commercial model, is anybody can go try the project for free. They can run it on their own for free. And then once the use cases become very critical, because they've had a great experience with it, or as soon as you start to scale to multiple teams,

Or as soon as you see your team doing a lot of work, keeping Airflow or any open source projects up and running, that's oftentimes when it makes sense to turn to a managed service. Again, there's like 80,000 companies out there already using Airflow, the open source project. And we've, we've captured a very small percentage of that today. And we want that 80,000 to continue growing, right? We're

In the process now of releasing Airflow 3.0, which is the next major version of the project, 2.0 was released about four years ago now. Just to give you maybe some context on growth, the 2.0 download numbers were around a million a month four years ago. And because we've continued investing in the open source project and releasing

a great set of features that the market seems to really like. It's now downloaded over 30 million times a month in four years. So we've seen some explosive growth there and I've obviously felt the tailwinds that in our commercial business.

Aaron Delp (14:04.719)
That's fantastic. And let's explore that a little bit. What does a typical implementation look like? Because the way I think of it is like, your typical, like, yeah, hey, I could go download the open source and play with it myself. But when do the growing pains kick in? When you're trying to introduce these pipelining tools, do you get to a certain point and you're like, oh gosh, this is getting to be a mess and I now need to...

do standardization, it you hit scale issues or is it like, we talk about a lot, know, like with Kubernetes and orchestration, like the bigger it gets, the nastier it gets. So complexity becomes the root issue. Like tell everyone a little bit about what a typical journey looks like from small to large.

Julian LaNeve (14:55.674)
Yeah, it's a great question. obviously it starts with someone needing orchestration. Once you have or are starting to build out a data platform, you have multiple tools that all need to talk to each other, right? You might have a sales tool, a marketing tool, you might have a data warehouse, a BI tool. As soon as you need those things talking to each other, you need orchestration sitting on top of it. That is

exactly why orchestration exists. Then folks typically decide they want to use Airflow for a number of reasons, but the two in particular are Airflow itself is the most popular orchestration tool by a very large order of magnitude. And that means that it's already very well integrated with the most common tools you'd see in a data platform.

And that means that like you, your engineering team doesn't have to spend time building out those integrations. They can just go use the integrations that already exist. And the other, the other big reason that folks turn to Airflow is orchestration becomes very difficult to replace once it's kind of rolled out to an organization. And the reason for that is, is because it talks to every single data source and system. So if you want to go replace that,

You have to go reintegrate the new system with basically everything at your company. And so you oftentimes don't want to take a risk when choosing an orchestration tool. That's why I actually love that Airflow itself is in the Apache Software Foundation, as opposed to something that's like fully astronomer owned. Because if we were to ever go out of business, Airflow, the open source project would continue existing and would still be very well adopted. So there's very little risk.

in choosing Airflow. There are many ways of running Airflow. You can turn to one of the cloud providers. You can run it on Kubernetes yourself. You can use some of the tooling that we've put out there. And oftentimes, that's a great way to get started by using the Cloud Providers Managed Service or running on Kubernetes yourself if your team has the experience to do so.

Aaron Delp (16:47.129)
Yeah.

Julian LaNeve (17:13.114)
And then we see folks turn to us typically when one of a couple of things happens. The first is scaling issues, right? We can run the airflow at a much larger scale than the cloud providers, than typically folks running Kubernetes on their own, because we've seen airflow scale to thousands, tens of thousands of pipelines. And so we know exactly what's going to break, how to get ahead of it so that as you continue scaling your usage, it continues going well.

The second is complexity. So as you start to have more complex use cases and more complex pipelines and different teams working on the platform, it's oftentimes helpful to have someone that knows the project incredibly well, right? We have office hours. We're very close to all of our customers to the point where like they'll come to us and just bounce ideas off of us, right? Hey, we want to get our ML team using Airflow to deploy ML pipelines.

Like, can you make some recommendations for how it's been done before? That's the type of question that we get all the time because we've seen every way of running Airflow. So oftentimes outside of just like helping with the infrastructure, it's nice to have a trusted partner in building out your data platform. And we very much see ourselves that way. And the third is when

you're scaling to multiple teams and you need something that sits on top of all of your Airflow deployments.

to help you manage them, make sure they're consistent, make sure they're all upgraded, make sure that the right people have access to the right things, right? The typical kind of enterprise scaling challenges. There are a number of other reasons that customers come to us, but I'd say those are typically the three big ones. You run into scaling issues with the open source project and want to delegate that to us. You need a trusted partner to help build out your data platform outside of just managing infrastructure.

Aaron Delp (18:52.173)
Yeah.

Julian LaNeve (19:06.79)
or Airflow works great for you with one team, you want to go roll it out to your entire organization, and you need something to help you manage many Airflow deployments at once.

Aaron Delp (19:15.809)
No, I love that. And excuse me, one aspect I want to kind of close out on is anytime I think about automation in general, you know, whether it's in data pipelines context or otherwise is security, especially at scale. And so...

What recommendations, because like you were mentioning, like, hey, the organization wants to expand and there's multiple areas or multiple groups or whatever, right? What recommendations do you have for developers around best practices regarding security or multi-tenancy, for instance, for data pipelines as well?

Julian LaNeve (19:57.766)
Yeah, data pipelines become interesting to think about from a security perspective because there can be misalignment between what a user has access to and what the data pipeline has access to. You want to make sure that the data pipeline, which is oftentimes acting with a service account, only has access to what it needs. Airflow was built with security and best practices in mind.

You can actually declare for every pipeline what connection it should use, and that contains the credentials that are necessary to go access any of your data sources. You can go store those credentials in a managed secrets backend like HashiCorp's Vault product or Google Secrets Manager, any of the others, so that all of your credentials live in one place. You have full control over them if something were to ever go wrong.

you can shut off access immediately. The astronomer product itself, I think, helps a ton with authorization and authentication. So we're actually able to tie into the customer's SSO system so that as new folks are onboarded or off-boarded, those changes get propagated through the data platform through our commercial platform.

And then of course, like you want audit logs, you want to make sure that you understand who's doing what. And that's oftentimes where it's also helpful to turn to a managed service because again, I'd say that the one thing that we for sure will always do very well is build for the enterprise because going back to that maturity curve, folks only come to us when they're running into scaling issues or multiple teams.

And so that's very much how we've built the product to go meet the security needs of even the most secure organizations in the world.

Aaron Delp (22:00.377)
Fantastic. I love it. Well, Julian, I think that's a great place for us to close out. Thank you very much for your time this week. If anyone out there is interested, what's the best way to get started?

Julian LaNeve (22:13.008)
Yeah, so we actually just released Airflow 3. I would definitely recommend getting started with that because there's a ton of new exciting things in there. You can do that by going to the Apache Airflow GitHub project or the Airflow site. Super easy when you just Google Airflow. You can run that locally for free. It's very easy to get started with. And then if you want to check out

kind of our cloud products and how we think about things. The astronomer.io website has a link in the top right to start a free trial. We give folks a couple hundred dollars in credits to go kick the tires and see how it works.

Aaron Delp (22:57.423)
Fantastic. Well, Julian, thank you very much for your time this week. And on behalf of everyone out there, thank you very much for listening this week. And if you enjoyed the podcast, certainly tell a friend, leave us a review wherever you get your podcast, if you are able to do so. And of course, we're always looking for feedback. Show at thecloudcast.net. On behalf of Brian and myself, thank you everyone for listening and we will talk to everyone next week.