Feathercast Script

Ideal time is 5-10 minutes.

Interview will take place on Google Meet, and include video as well as audio.

Resulting interview will be posted in both formats - audio on <http://feathercast.org/> and video on <https://www.youtube.com/theapachefoundation>

Hi, and welcome to Feathercast. My name is \_\_\_\_\_\_\_\_\_

Today I’m speaking with $Person from the Apache $Project project. $Project is a \_\_\_\_*description from the website*\_\_\_\_\_, and $Person is going to tell us some more about what that means.

Welcome, $Person!

Q: I read the description of your project from the website, but what does that actually mean? What does your project do? Who would use it?

The concept of an API Gateway has been there for several years. So an API Gateway acts as the single entry point into the system. In addition to request routing, it can perform additional duties such as request transformations, authentication, monitoring, and load balancing. Hence, in a nutshell, all requests from clients will first go through the API Gateway, and then will be routed to the appropriate service. Therefore this can be used either to expose your applications to the internet or inside the intranet for your microservices to communicate with each other.

I got introduced to APISIX while we were evaluating some existing API Gateways for a project. And to my surprise, there were quite a lot of opensource API gateways available and some were popular and been there for several years. Hence this intrigued me on why would Apache incubate another API Gateway.

The reason turns out to be due to the current landscape of the internet. From a historic standpoint API gateways didn’t have such high- performance requirements more than a decade ago. At that time Internet traffic mainly runs from the browser to the server, and there was less traffic from microservices or the intranet. And most of the API Gateways built at the time were limited to the technologies at that time. However, there is a lot of traffic now, with mobile phones, IoT devices. This will keep on increasing with the introduction of 5G technologies. At the same time, the internal traffic is also getting increased with the hype microservice architecture. Therefore in this new business situation, there are more new requirements for the API gateway.

One of the main requirements is the high performance and ability to scale and reduce at will. The load testing of APISIX shows it outperforms some of the popular API gateways by 5-10 times. Its implementation is based on Nginx and etcd. Compared with traditional API gateways, APISIX has functions such as dynamic routing, plugins hot-reloading, and gRPC protocol transcoding, which is especially suitable for API management under the microservices system.

Another goal of APISIX is to be cloud-native friendly with its light architecture and a low memory footprint. Due to the cloud-native architecture, it can run in multi-cloud and hybrid cloud environments. APISIX tends to keep the core small as possible and adds support for additional features via using the concept of plugins. Therefore most of the general functionalities such as authentication, open tracing, and support for serverless computing are developed as plugins. This also allows users to develop custom plugins for their custom use case, instead of relying on the community.

So although relatively new to the industry APISIX addresses the requirements of the era of the cloud-native and microservices landscape.

Q: Tell us some user stories: Give us an example of people who are using $Project in the real world, and what kinds of problems they are solving with it.

APISIX is gradually getting adopted by many organizations. For us at Salzburg Research we needed an API Gateway to secure the API calls in a European Union project called eFactory. Over the years the European Union has funded many projects in the industry 4.0, and eFactory is an effort to integrate such platforms and to provide a federated platform with seamless access.

As we are dealing with large platforms latency was one of our major concerns and APISIX seems to be a perfect candidate for the job. Also as this is a multi-platform ecosystem with multiple authentication boundaries we needed an elegant and simple approach to develop custom authentication and authorization plugins. And due to the modular approach, the effort to develop plugins was less. Also, as we deal with a lot of Factory data MQTT support for the API gateway was a good addition. Currently, we use the APISIX as an API security gateway to authenticate/ authorization and enforce policies to all the API calls in the eFactory ecosystem.

Q: Name origin story? (If weird/interesting name.)

The project was originally donated by a startup company in China. And number 6 is considered a lucky number there.

Q: Recent releases/development/activity?

We’ve been pretty active throughout the year. Apache APISIX has released 6 Apache releases by 6 different release managers in the seven months since it entered the Apache incubator, almost one release per month. As this is an incubating project and a lot of features are down the road we would love to get new contributors. The project has two main parts, the core, and the dashboard. The dashboard is being redesigned with React and ant design at the moment. The next release mainly adds support for Apache Skywalking, by increasing the traceability of the system. Later releases will also add support for open source projects such as Apache Dubbo, Apache Pulsar, and others

Q: Where do you hang out? Where should I come to connect with you?

We do usually use our developer mailing list for communication. We also discuss features in the Github and we do also have a slack account.

Q: Where do I get more info? (This is where you advertise your website, mailing lists, and other online resources. Also promote upcoming events, if any.)

* The website
* Docs in the repo
* Getting started guide
* We also have a twitter account where we announce major activities in the project.

----------------

Resources for projects:

* Update your data on projects.apache.org - <https://projects.apache.org/about.html>