A Broadcom Company
technologies

White Paper

CA Brightside - Enterprise Support for Zowe

Business and Infrastructure APIs on the Mainframe

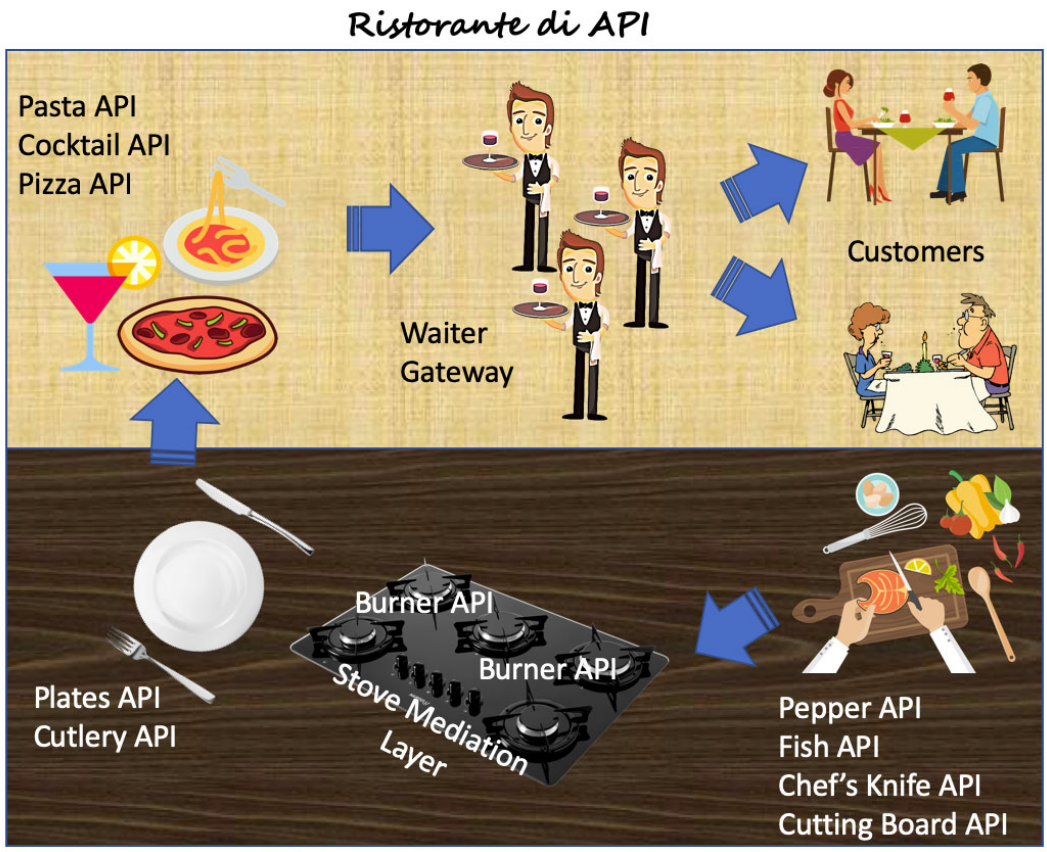
Summary

API management tools such as those in the Layer7® API Management platform are used by application delivery teams to manage business APIs in order to meet service-level agreements (SLAs). For infrastructure services like Kubernetes, IT teams manage their APIs using a tightly coupled lightweight mediation layer. In the same vein, the Zowe API mediation layer manages infrastructure APIs for the mainframe platform.

IBM z/OS Connect is a tool that builds business APIs on the mainframe. These mainframe business APIs should be managed by application delivery teams using a business API management tool. These business APIs should not be managed by the Zowe API mediation layer because the capabilities of that tool are better suited for the technical needs and IT teams who manage infrastructure.

The Zowe API mediation layer allows mainframe IT departments to provide gated, performant access to infrastructure APIs for their internal consumers such as development teams, DevOps automation, and mainframe IT staff.

Figure 1: Variety of APIs in the Restaurant of APIs



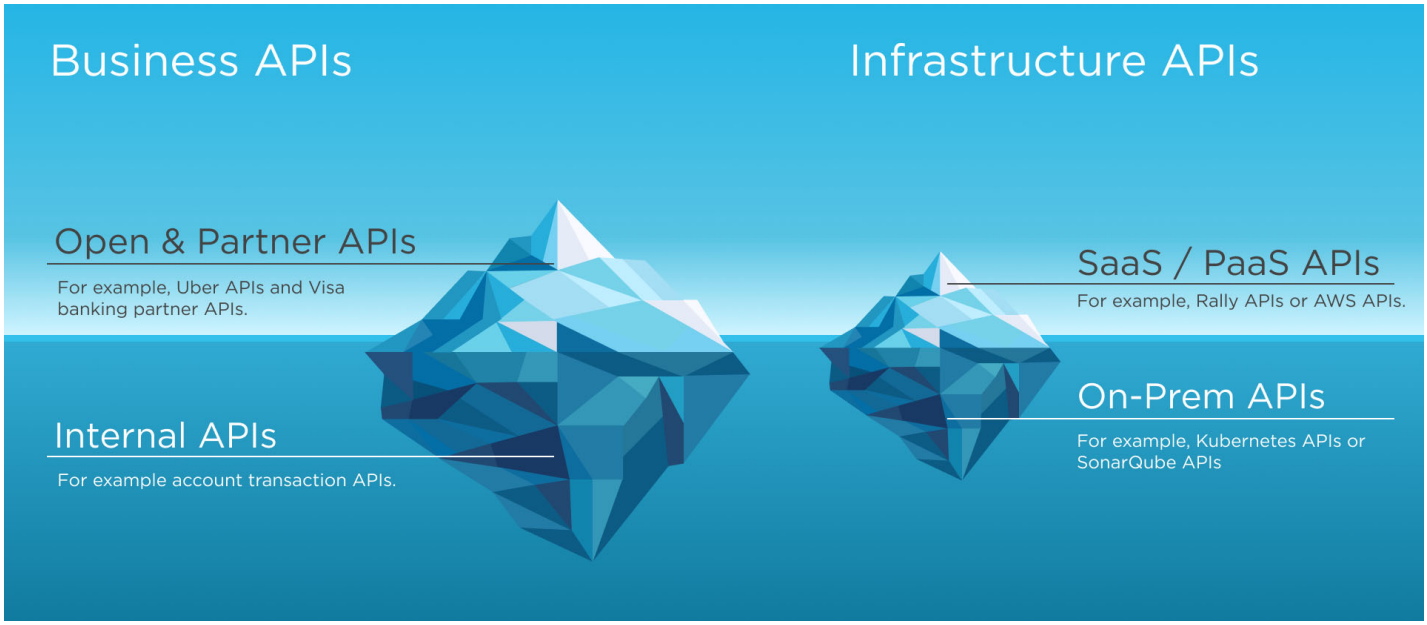
Introduction

APIs are application programming interfaces. At its core, an API allows one computer program to communicate with another program using an agreed upon protocol. APIs power the modern app driven world. Ever wonder how your Uber app uses Google Maps? The answer is APIs.

When we refer to APIs, we'll be speaking strictly about web APIs. Within web APIs, there are a few standard protocols such as HTTP/HTTPS (Representational State Transfer (REST) and webhook) and SOAP. Our focus will be on the most popular form of web APIs in use today known as REST.

Before we dive into the world of API management and how the Zowe API mediation layer fits into that space, we need to first understand two distinct purposes of REST APIs in organizations. In the following diagram, there are two icebergs. The iceberg on the left represents APIs that are used for business activities. The iceberg on the right represents APIs that are used to access internal IT infrastructure services.

Figure 2: The Business and Infrastructure API Paradigm



Business APIs can be defined as APIs that offer a service to your organization's end-users. Business APIs are a highly valued asset in today's world of API economy and they add a new dimension to your product offerings. Business APIs can be used by an end user to get data or extend an application.

IT infrastructure APIs are not meant for your organization's customer. They are APIs for tools like GitHub that your organization uses internally to optimize, automate, and run your business. Infrastructure APIs are used by administrators to manage and administer core systems. For example, IT infrastructure APIs can be used to deliver status data to a management dashboard to monitor critical systems.

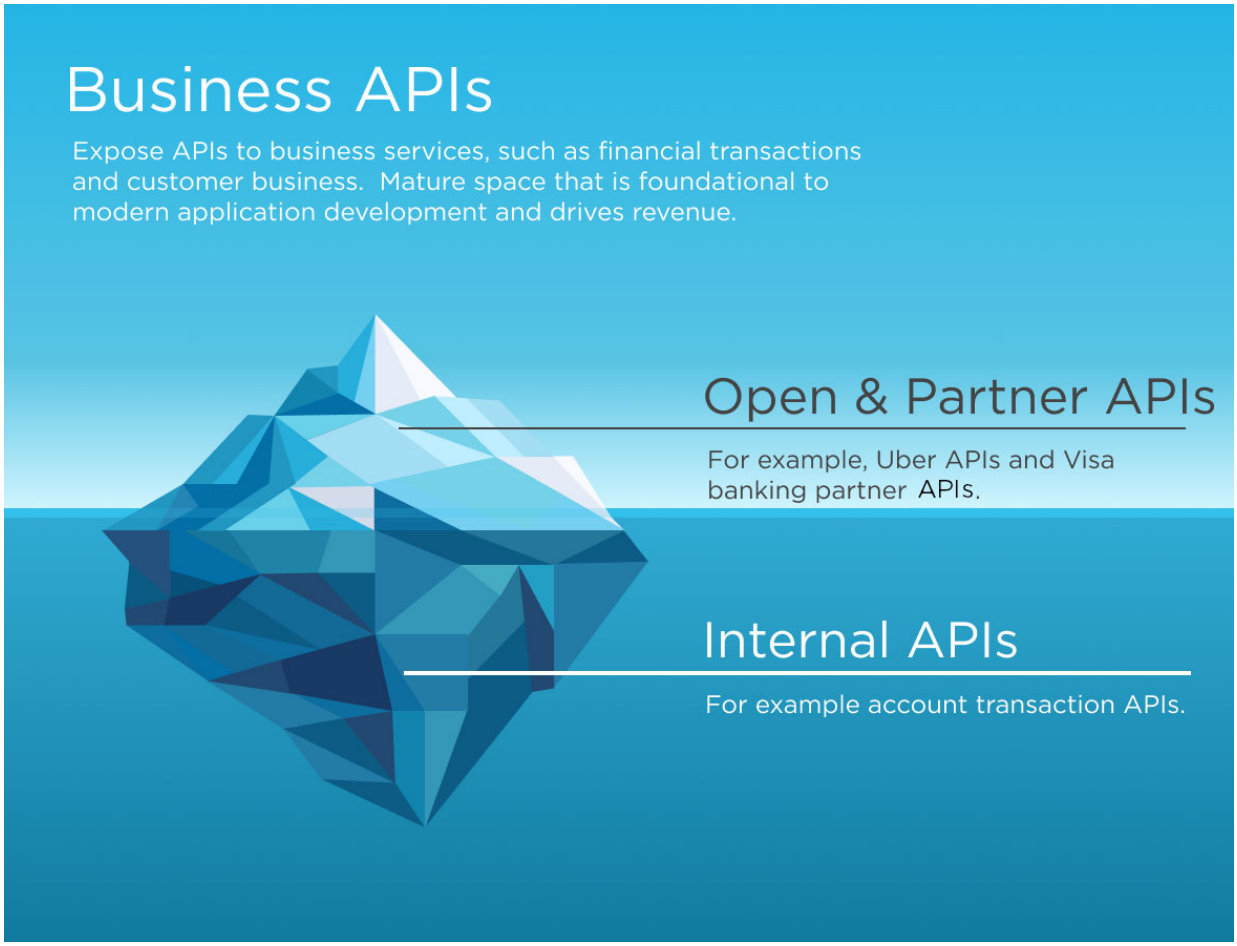
Business APIs

Business APIs are used to access data and transactions related to your core business activities. They must be secure, performant, and managed to meet stringent SLAs as your business relies on them to offer services to your customers.

Classification of Business APIs

Business APIs can be broken down into open APIs, partner APIs, and internal APIs. The following figure shows how the types of business APIs relate to each other within the business API iceberg analogy.

Figure 3: Types of APIs



Open APIs are business services exposed to anyone on the Internet for public consumption. Open APIs are also commonly referred to as public APIs. For example, Uber documents and exposes their ride request services through REST APIs that anyone can use to integrate with their apps.

Partner APIs are services provided by or used by third-party organizations with special access. For example, Visa allows participating pilot banks to develop a B2B specific payments solution using their Visa B2B Connect APIs.

Internal APIs are building-block APIs for applications or other APIs that ultimately serve external business purposes. Internal APIs are not exposed for external consumption, but serve as a cog in servicing many other external facing UIs or APIs. For example, an API to look up a bank account holder's pending transactions for deposits or withdrawals.

Mainframe Business APIs

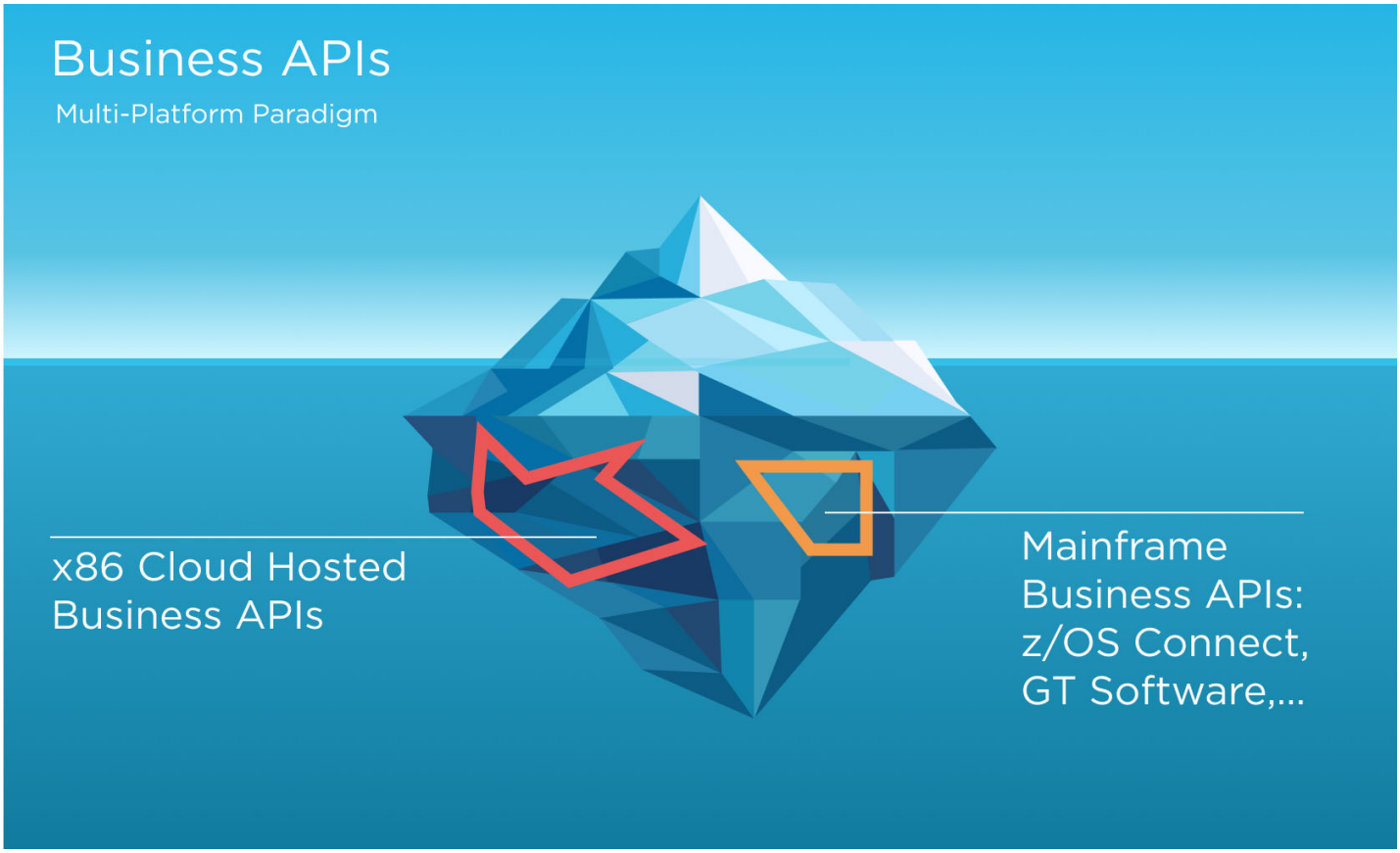
Many mission-critical applications run on the IBM mainframe platform. These applications rely on infrastructure services like IBM CICS, IBM IMS, IBM Db2, CA IDMS™, and CA Datacom® to manage heavy loads of transactional processing. These mainframe business transactions can be exposed through REST APIs.

IBM z/OS Connect EE is a product that allows you to layer in REST APIs on top of applications with transactional logic such as IBM CICS and IBM IMS. In the past, transactions built into these subsystems had to go through several layers such as IBM MQ or expose the CICS Transaction Gateway through an Enterprise Service Bus and then undergo heavyweight transformation to be exposed as a SOAP or REST API. IBM z/OS Connect EE changes this paradigm by allowing business transactional logic that lives on the mainframe to be exposed easily as REST APIs. There are other products such as GT Software Ivory Suite that also allow API enabling of mainframe business logic.

The APIs created with these business transactional logic tools fall into the business API category, specifically into internal APIs. They are internal business APIs because other external facing GUIs, apps, or APIs may use these APIs to expose business data or services to the end-user. **These APIs are often exposed in an API management tool, such as CA Technologies, A Broadcom Company, Layer7 API Management platform, Google Apigee, or Mulesoft Anypoint API Manager.**

The following figure shows the source of business APIs within the iceberg analogy.

Figure 4: Source of Business APIs

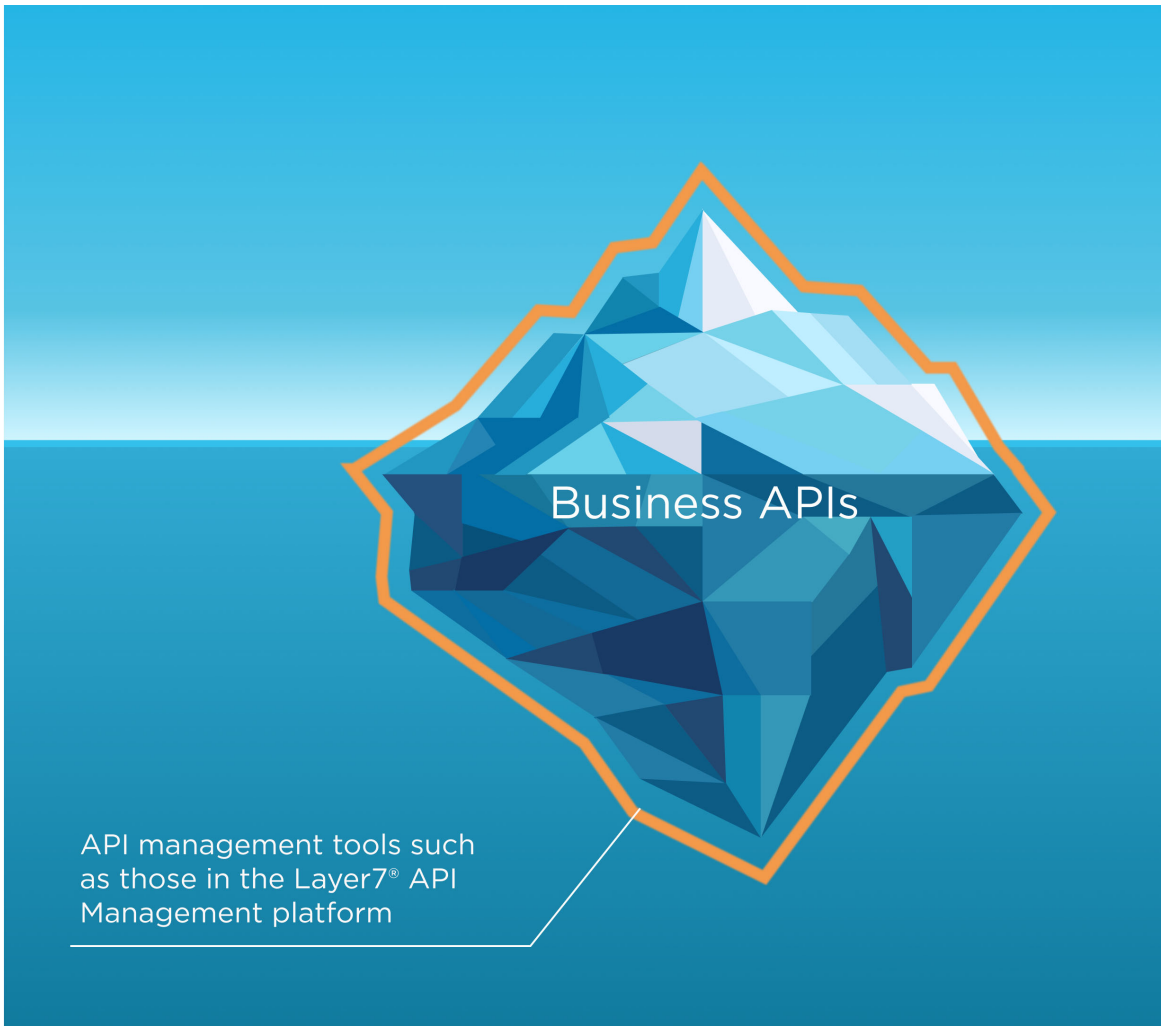


API Management Solutions

With all these APIs being offered and used, there comes a need to establish some best practices and management around them. Organizations have concluded that they need tools to manage their APIs for a variety of reasons including security, authentication, load balancing, usage metrics, analytics, monitoring performance, regulating traffic, and documenting in a consistent and ongoing manner.

Given all these reasons, API management tools have established themselves as a vital cog in the IT architecture of most organizations that build business APIs—open, partner, and internal. As shown in the following figure, these tools need to manage all types of business APIs.

Figure 5: API Management Around Business APIs



Developer portals similar to what we saw with Visa and Salesforce can be externalized using API management tools like the Layer7 API Management platform, Mulesoft Anypoint API manager, or IBM's API Connect. Internal accessibility to these business APIs in a centralized API catalog is one of the reasons for many recent app innovations such as "voice payments" from Barclay's. Fortune 500 companies often have more than10,000 line-of-business application developers. The possibilities for innovation are endless when all these developers have access to the full catalog of open, partner, and internal business APIs.

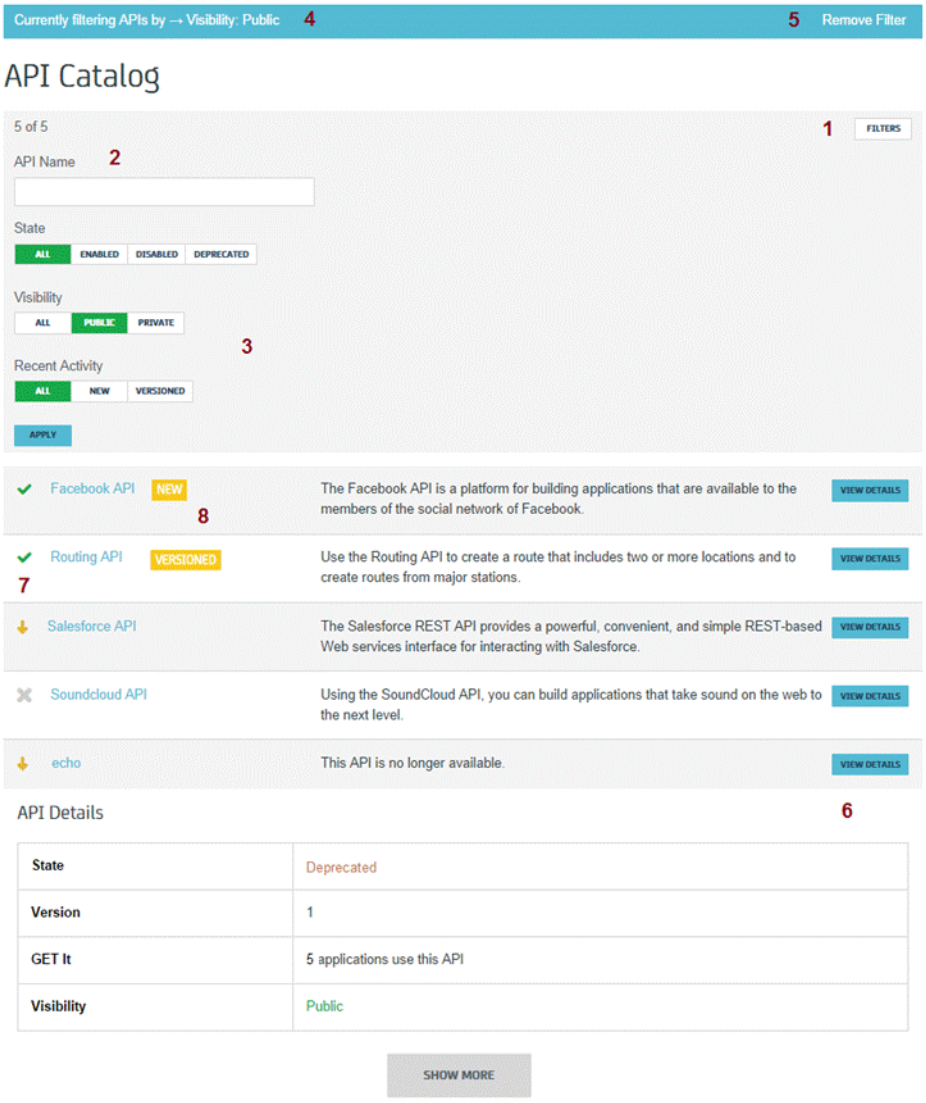
API Routing

API's are routed through a combination of API gateways or service meshes. The primary difference between the two is that API gateways manage north-south traffic (traffic from outside the network to inside the network), while service meshes manage east-west traffic (traffic between services running inside the network). However, the terms gateway and service mesh are sometimes used interchangeably since there is not a lot of difference in terms of how they manage APIs.

API Catalog

Within the context of business API management, API catalogs organize and document APIs in a meaningful format through a web GUI that promotes wider internal usage of these APIs. The value of being able to provide access to APIs from 1000s of microservices and services running on many different platforms —including mainframe—is a powerful prospect in fueling innovation. Many organizations use API catalogs to expose business APIs from the mainframe to their entire staff of application, and front-end developers resulting in a never-before-seen influx of innovation. The following figure shows an example API catalog.

Figure 6: The API Catalog in the Layer7 API Management Platform



Application Delivery Management Role

In addition to the technology involved in managing business APIs, it is important to understand the ownership of responsibility for the business APIs SLA. This responsibility generally lies in the hands of the application delivery management teams (ADMs). Each ADM team may be responsible for managing the delivery of one or more of your business' product lines. The ADMs constantly monitor, adjust, and optimize the API management controls to keep the flow of both north-south and east-west traffic running as smoothly as possible.

Infrastructure APIs

Infrastructure APIs provide access to the tools, services, and platforms that your organization uses to create valuable products for your customer such as GUIs, business APIs, apps, or other products.

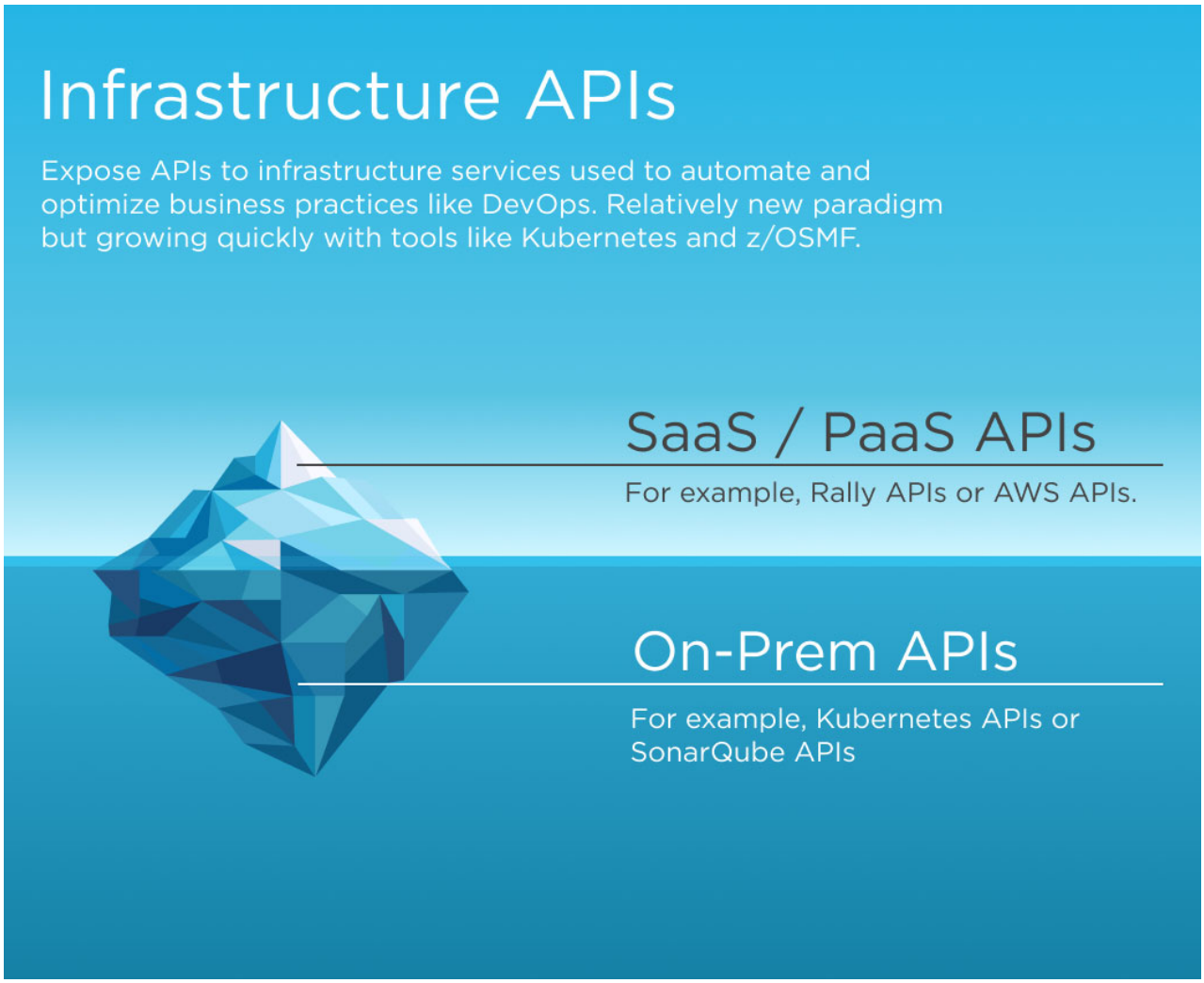
Some examples of these infrastructure tools include Kubernetes, SonarQube, Jenkins, AWS, IBM CICS, IBM IMS, CA Datacom, and CA Endevor®. Why are infrastructure APIs important? These APIs are what allow your development teams, DevOps teams and IT teams to automate repetitive processes within your organization. If it wasn't for infrastructure APIs, we wouldn't have any build automation, automated provisioning of test or prod environments, or deployment automation.

Classification of Infrastructure APIs

Infrastructure APIs can be broken down into on-premises services and SaaS/PaaS services. On-premises services are hosted on the organization's internal network. An example of an on-premises service with REST APIs is Kubernetes which offers APIs for various container management actions (Note: Kubernetes could be deployed on an off-prem cloud host, but for the purposes of this discussion, let's assume it's deployed on an on-premises host). SaaS/PaaS services are hosted by third-party organizations outside your organization's network. The advantage of SaaS/PaaS services is that your organization doesn't need to run anything on their infrastructure, but the caveat is that you'll need to access this service through the public Internet. A popular example of a SaaS service used by many organizations is Salesforce. In addition to the browser based GUI, they also offer the Salesforce Lightning Platform REST APIs. Cloud platforms like AWS, Google Cloud and Microsoft Azure are examples of off-prem PaaS services which offer CLIs as the primary interface. The CLI interface utilizes REST APIs to offer scriptable access to the PaaS services.

The following figure shows how the types of infrastructure APIs relate to each other within the API iceberg analogy.

Figure 7: Classification of Infrastructure APIs



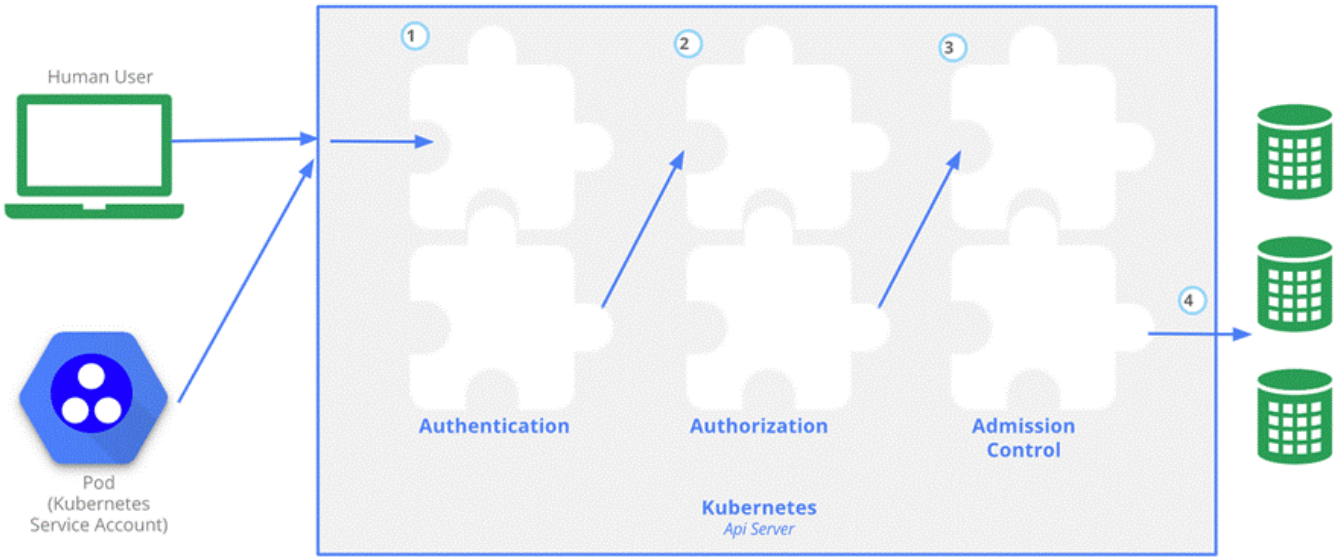
Simple On-Premises Service

Simple on-premises services like SonarQube come with REST APIs that are fairly basic. They accept API calls, authenticate the user, and provide a response. The API user is aware of the URL/port of the internally hosted service and just calls them directly with basic authentication credentials. Other than the need for basic authentication, there isn't a need for management of these APIs.

Complex On-Premises Service

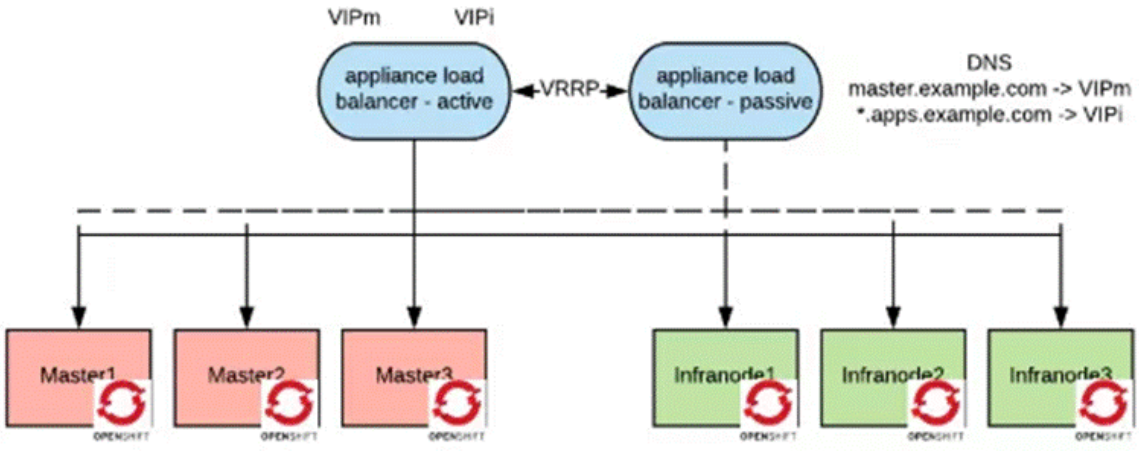
An example of a fairly complex on-premises service is Kubernetes, which is a “platform” for container management. It comes coupled with more advanced API mediation. The additional level of API mediation is required as the services offered by Kubernetes have a mission-critical impact on the business. This results in more complex security management, queuing of requests, and queuing of authentication. In addition to this functionality, the rate of API requests for deployment actions made from DevOps tools like CI/CD orchestrators may push the ability of the Kubernetes API server to service API requests in a timely fashion. The following figure shows a model of Kubernetes API requests.

Figure 8: The Kubernetes API Server (Source: kubernetes.io/docs)



To address this need for scalability and high availability with Kubernetes' infrastructure APIs, enterprise versions of Kubernetes such as Red Hat's OpenShift advise using load balancers. The following figure shows an example configuration with load balancers.

Figure 9: Load Balancing of the OpenShift Container Platform (Source: openshift.com)



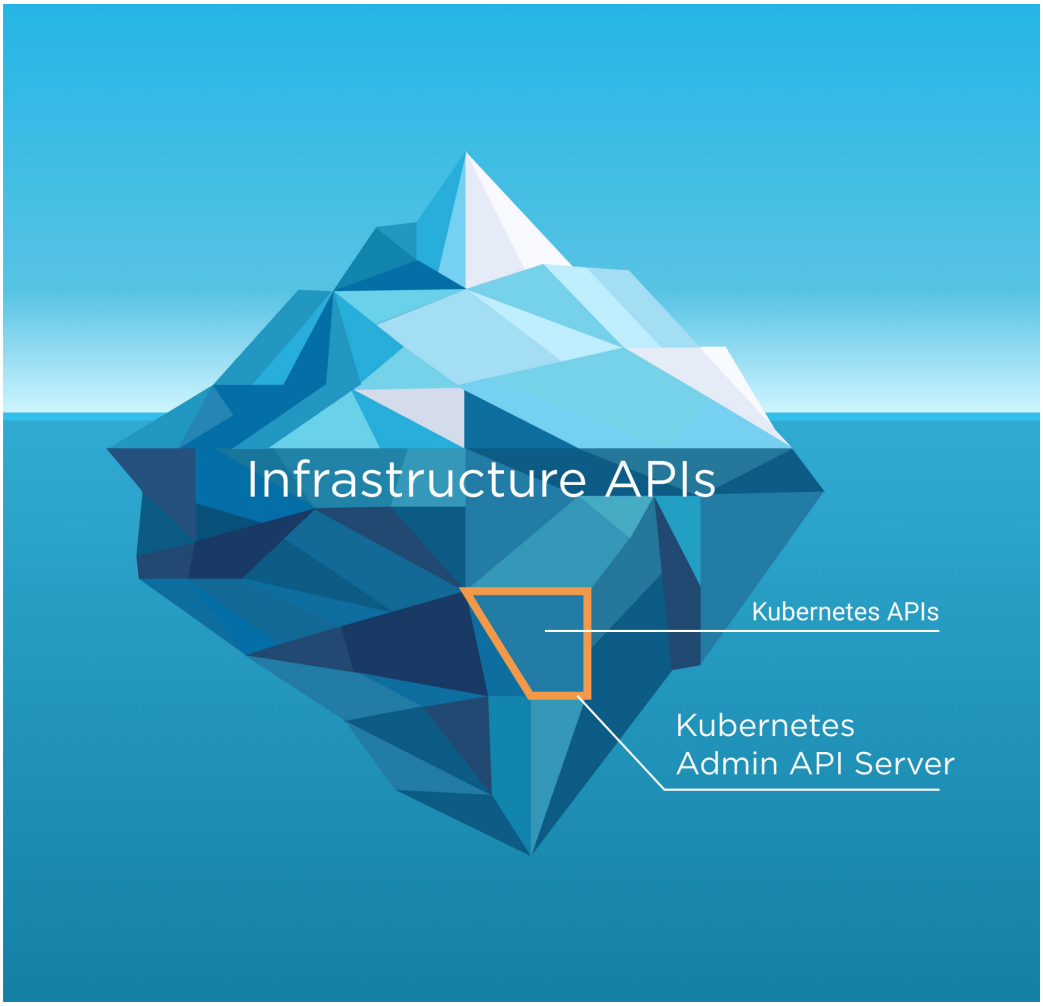
Other PaaS services such as cloud platforms (AWS, Google Cloud) likely use similar API mediation within their infrastructure to manage incoming API requests for their services.

Infrastructure API SLA Ownership

There are dedicated application delivery management (ADM) teams that manage the performance of the business APIs. Infrastructure APIs are not under the management realm of the ADMs. Instead, infrastructure API SLA is managed by internal IT teams. Unlike business APIs, infrastructure APIs are generally not centrally managed using API management tools or made available on a single, comprehensive API catalog. Whether it should be best practice for IT teams to start centrally managing infrastructure APIs just like business APIs is a question for another day. Today's reality is that they are not managed centrally.

As shown in the following figure, it is common to package a lightweight mediation layer for the infrastructure APIs of more complex on-premises services.

Figure 10: Lightweight API Mediation Around Kubernetes



IT teams that manage these on-premises services rely on these pre-packaged mediation layers to manage the SLAs for these infrastructure APIs. A subset of the benefits of commercial API management tools like security, authentication, load balancing, regulating traffic, and documentation can be achieved using the tightly coupled mediation layers of complex on-premises services.

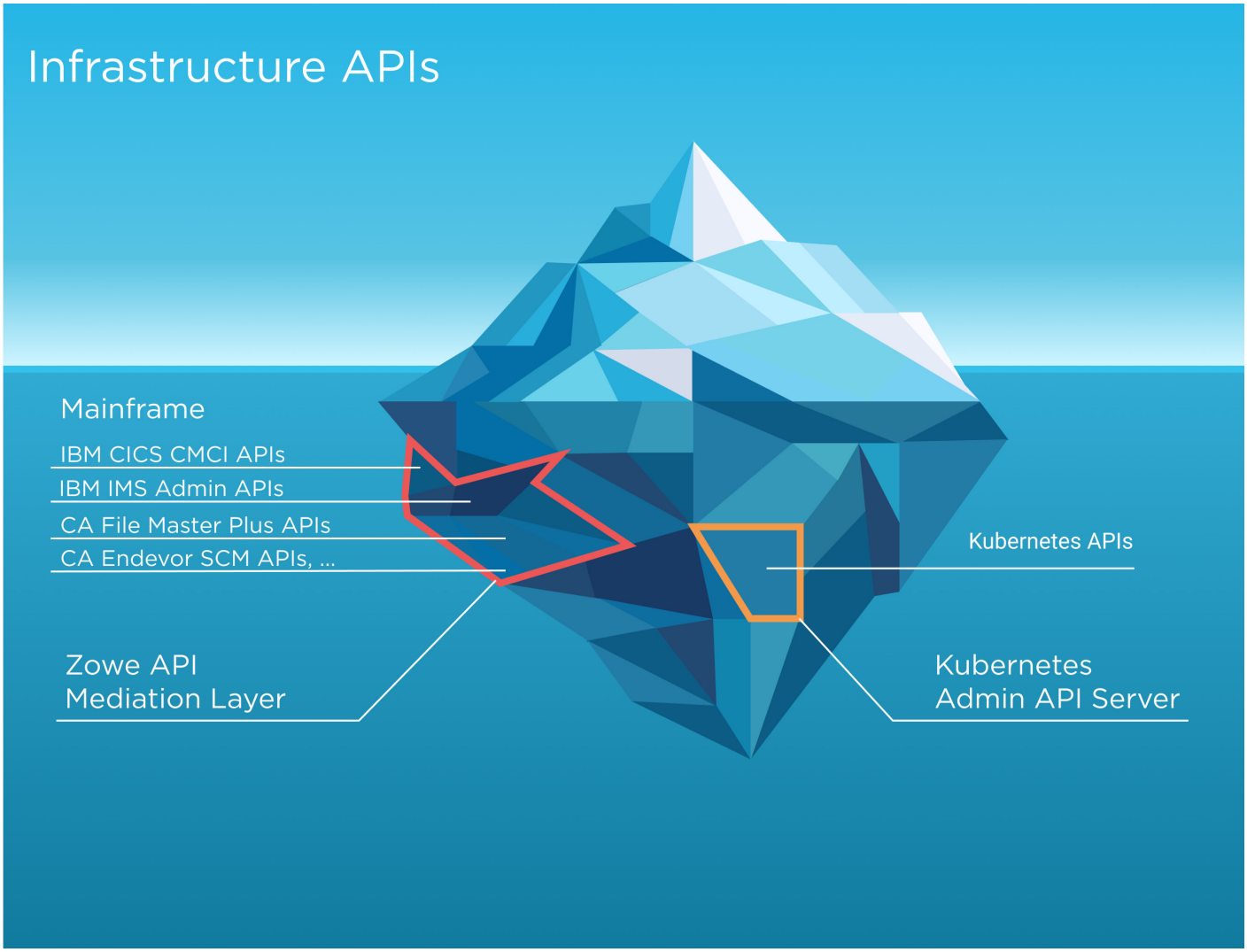
At this point, it is critical to understand:

- The difference between business and infrastructure APIs
- The difference between commercial API management tools used to manage business APIs and pre-packaged mediation layers that come with complex on-premises infrastructure services
- The division of responsibility with ADMs managing business APIs and IT teams managing infrastructure APIs

Zowe API Mediation Layer

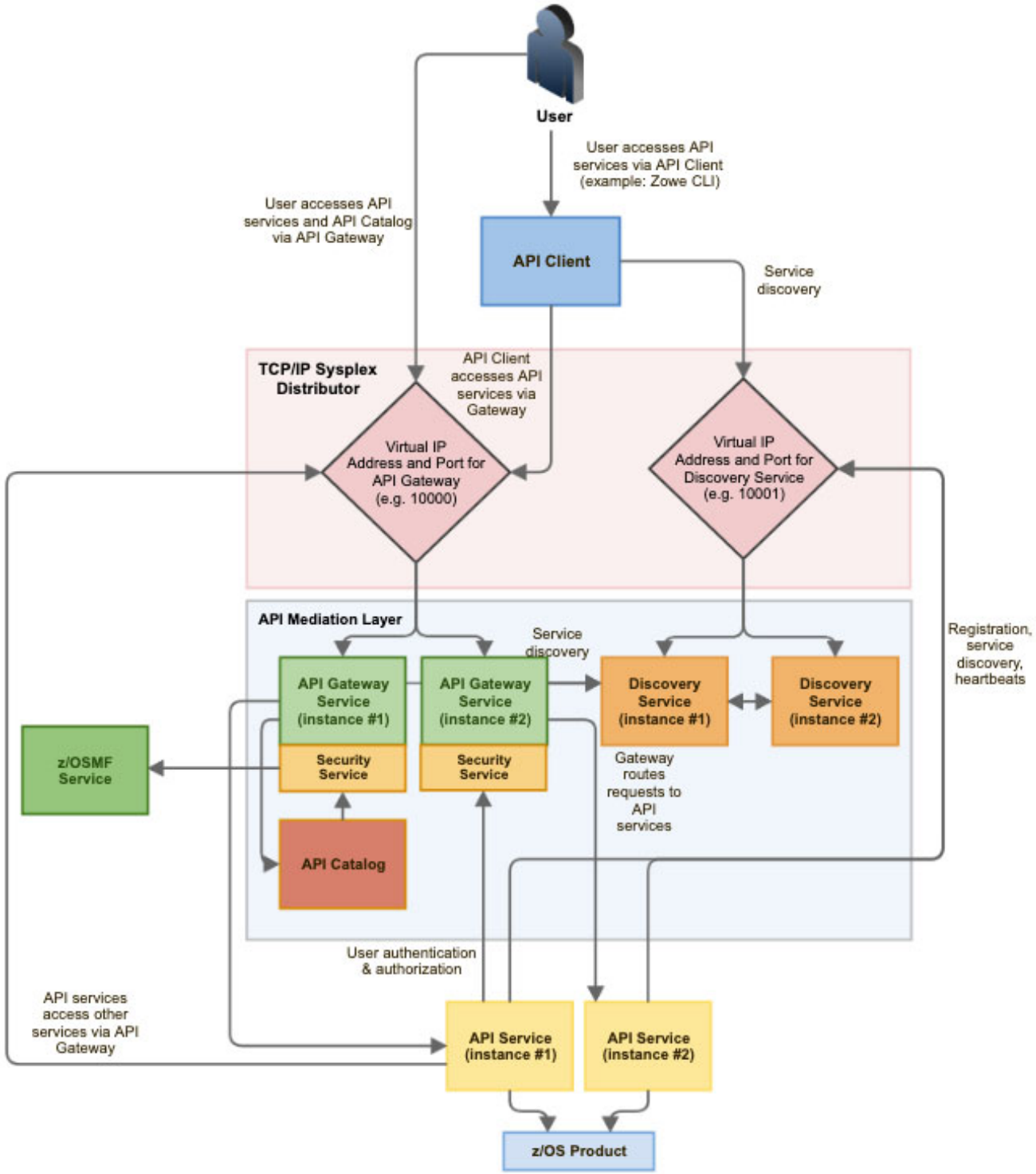
Now comes the big question. How does all this relate to the Zowe API mediation layer? Let's draw a parallel with the other infrastructure services we've discussed so far—OpenShift/Kubernetes, AWS, and Google Cloud. All of them offer REST APIs for administrative and operational purposes that are widely used by development teams for DevOps automation and IT automation. All of them have built-in API mediation to address routing, security, and SLA needs. In the same manner, the mainframe —specifically the IBM z/OS operating system, along with the variety of batch, transactional, and database services —act as an on-premises deployment platform and needs the same type of API mediation. The Zowe API mediation layer is a solution to this need as shown in the following figure.

Figure 11: Lightweight Zowe API Mediation Around Mainframe as a Part of Infrastructure APIs



Zowe has an objective to simplify the architecture of the mainframe infrastructure services. As part of this objective, Zowe is helping software vendors build REST APIs on top of traditional mainframe services to enable programmatic access to infrastructure on the platform. The following figure shows how a user accesses mainframe APIs with Zowe.

Figure 12: Zowe API Mediation Layer



Some examples of infrastructure APIs on the mainframe include:

1. IBM z/OS Management Facility (z/OSMF)
2. CA Endevor® Software Change Manager
3. IBM CICS Transaction Server for z/OS
4. CA OPS/MVS® Event Management and Automation
5. IBM MQ for z/OS
6. CA Workload Automation ESP Edition
7. CA File Master™ Plus
8. IBM z/OS Connect EE

These APIs can be used for a variety of SDLC automation such as automated testing, code builds and scans, DevOps CI/CD pipelines, help desk self-service, and chatbot integration. This is the same way APIs from cloud platforms are used for development, testing, help desk, and continuous deployment. Given that mainframe's role as a platform is not so different from the ones we discussed earlier (OpenShift, Kubernetes, and AWS) there is a need to impart some control over the growing list of infrastructure APIs. **The Zowe API mediation layer is a lightweight layer of centralization, developer portal, and load-balancing for the collection of REST APIs from various software vendors that make up the mainframe platform.**

Now that you know about all the different types of APIs, try to classify the APIs in Figure 1 (Ristorante di APIs) as business APIs, infrastructure APIs, and other parallels that you can draw with the world of APIs.

Common Questions

Can I use the API mediation layer in Zowe to manage my IBM z/OS Connect enabled APIs?

It is not recommended to use the Zowe API mediation layer to manage IBM z/OS Connect enabled business APIs from IBM CICS, IMS, or Db2. As discussed, the Zowe API mediation layer's domain is infrastructure APIs on the mainframe platform. API management tools such as those in the Layer7 API Management platform are generally established as the manager of business APIs which ultimately serve your end-user. z/OS Connect enabled APIs are traditionally business APIs meant to be used by external facing APIs or GUIs and thus belong on the management domain of a business API management layer. Ultimately, the decision on where to manage your business APIs powered by z/OS Connect should be made by your application delivery management team—not by the mainframe IT team.

Why do I need the Zowe API mediation layer if I already use a commercial API management tool?

For infrastructure services that expose APIs, a lightweight API mediation layer is generally tightly coupled with the service platform that relies on it—we saw this with the previous OpenShift and Kubernetes example and with cloud platforms such as AWS which likely run their own mediation layers to service incoming API requests. With the influx of infrastructure APIs on the mainframe platform, it is imperative that we establish control over the SLA and security needs of these APIs. Attempting to build a tightly coupled API mediation layer for the mainframe platform using your API management tooling, while feasible, would be like trying to build your own cloud platform from scratch. It would be highly unconventional and break established domain standards in API management. In addition to the effort, IT staff may not have the licensing and budget to use a commercial API management tool to manage infrastructure APIs.

Conclusion

Business APIs from the mainframe are becoming the norm for organizations that are going through a digital transformation. To fuel innovation within their application development teams, mainframe API enablement tools like IBM z/OS Connect are used to create business APIs. These APIs should be managed by application delivery teams using API management tools such as those in the Layer7 API Management platform.

Infrastructure APIs on the mainframe are growing in numbers rapidly, driven by the need to automate, optimize, and integrate mainframe operations with the rest of the IT infrastructure. The Open Mainframe Project's subproject Zowe has taken an aggressive role in accelerating the growth of Infrastructure API enablement on the mainframe by working with various software vendors like CA Technologies, Rocket Software, and IBM. To support the proliferation of infrastructure APIs on the mainframe, Zowe has introduced the API mediation layer which should be used by mainframe IT teams to manage these APIs.

CA Brightside is an offering from CA Technologies which provides enterprise-grade support, release cadence, and features that elevates open-source Zowe components—including the Zowe API mediation layer, for mission-critical, and production-grade usage.

ca technologies
A Broadcom Company