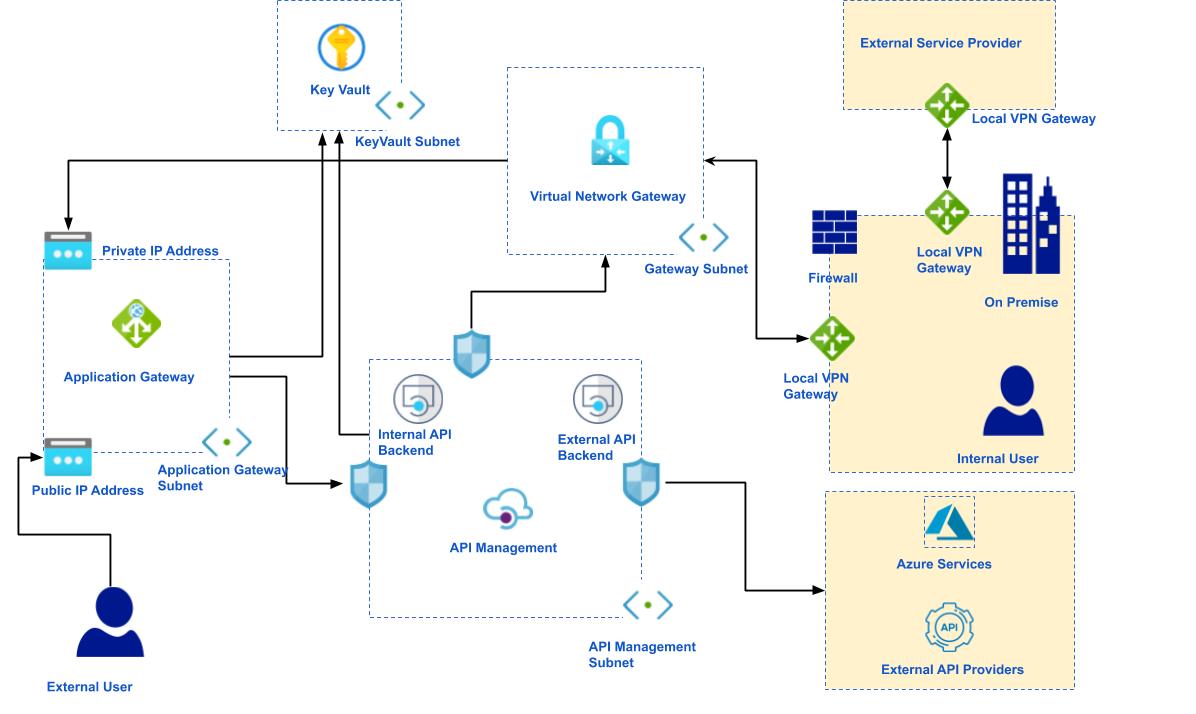
Task 1: Team A wants to communicate with API's owned by Teams B, C, D, E and F. All teams have separate Azure subscriptions. All API's need to communicate with a SQL server currently hosted on an Openstack cluster in the on-premise data center. Present to us a solution or multiple solutions for this scenario.

Solution: If there is a shared service spoke available, create a single API Gateway to publish the team's B-F APIs. Integrate the API gateway into a VNET that has access to the on-premises data center (via Express Route). For security purposes, publish the API gateway through Application Gateway and configure the WAF policies. Give team A access to the published APIs using subscription-based authentication, Azure AD (OAuth 2.0) or certificates.

If there is no such thing as a shared spoke, in each of the team's spokes, create an API gateway to publish the backend APIs. Integrate those API Gateways within a VNET for access to the SQL server. Create an API Gateway for Team A in their spoke where you will publish the APIs from teams B-F. This will give a central point of access for Team A to all the APIs that they are using, and they can further do transformations as needed at a gateway level.

Another option is to use the self-hosted API gateway option and deploy it in the Openstack cluster and use the Azure portal to manage the APIs. This option is feasible only if the APIs are hosted on-premises. Otherwise, it will introduce additional latency because of the roundtrip.

This is a design I prepared for on of my clients. We can use it to discuss the solution for the scenario



Task 2: CCoE would like to roll-out various policies accross 100 Azure subscriptions. These subscriptions are split into a number of distinct categories. Some policies would need to be applied to all subscriptions, and some only to subscriptions belonging to a specific category or categories. Present to us a solution or multiple solutions for this scenario.

Solution: Organize the subscriptions using Management Groups. Since the Root management group cannot contain policies and blueprints, create a single or multiple management groups below the root management group, depending on the preference. I prefer at least two, just to separate the dev/test from production root management groups to be able to test the root policies or blueprints on a top root level firstly in dev/test before applying it to the production tree. Create the rest of the tree, depending on category requirements.

Organize the policies in Initiatives or Blueprints and apply those on the management group level. You need to be careful when applying the policies at a management group level because for the resources impacted the most restrictive policy apply. For this reason, the most restrictive policies will need to be applied at lower leaves in the tree.

I took the following picture from the Cloud Adoption Framework, just to save time. We can go through it when I am describing my solution.

