

API Management

Sonu Sathyadas



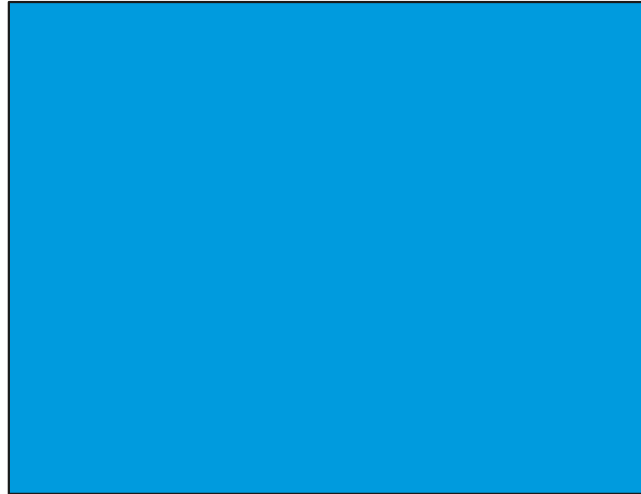
Agenda

- Understanding API Management
- Overview of Publisher and Developer portals
- Publish/Setup Web API in Azure (WADL, Swagger)
 - Use of Products, Analytics in Azure API Management
 - Configuring Policies at product or API or Operation level, API Caching
- Securing backend APIs
 - Implementing API securities using AAD, OAuth 2.0, OpenID Connect etc
 - Protecting and Optimizing APIs
- Working with Developer Portal
 - Subscribing to Products and APIs

Understanding API Management

The Old Way of Building Products

CoolProduct



The Old Way of Building Portfolios

CoolProduct



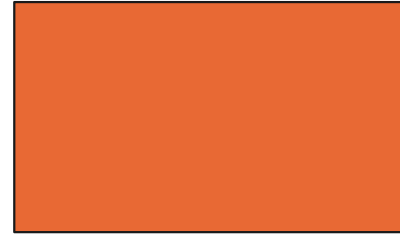
NewToy



WonderThing



SuperGizmo

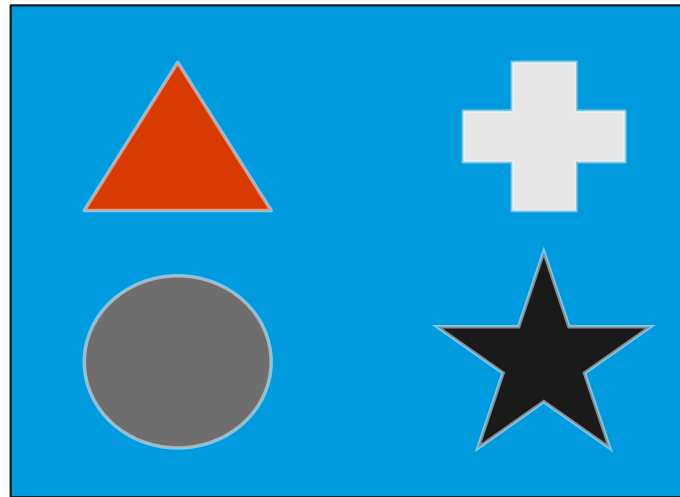


GameChanger



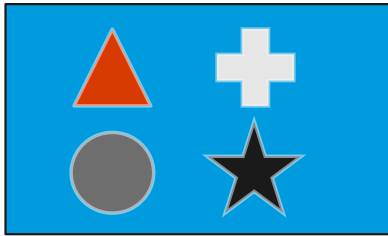
A Modern Product

CoolProduct

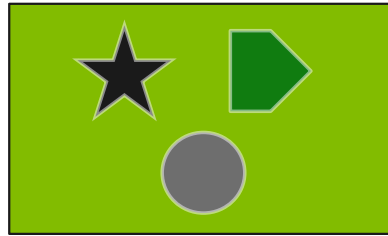


A Modern Portfolio

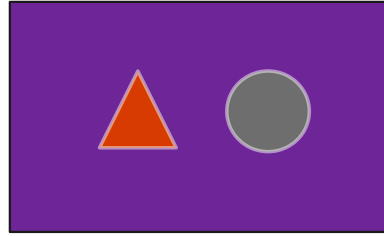
CoolProduct



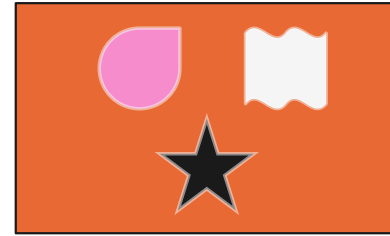
NewToy



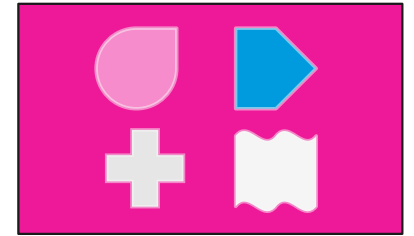
WonderThing



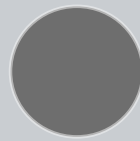
SuperGizmo



GameChange



API Store



Challenges with an API-based Approach

Discoverability. What APIs are available and what do they do?

Onboarding. How can consumers request access to APIs and learn how to use them?

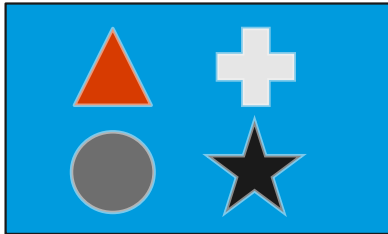
Security. How do I secure my vital data? How do I protect my backends from unauthorized access and overload?

Monitoring. Who is using my APIs and how much? Are my APIs online and working as expected?

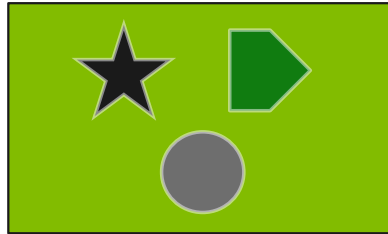
Lifecycle. How do I evolve my APIs without adversely impacting consumers?

A Modern Portfolio: Marketecture

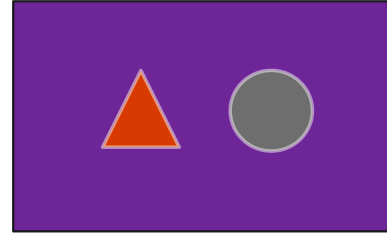
CoolProduct



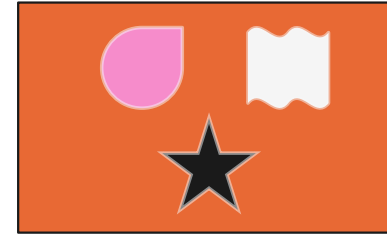
NewToy



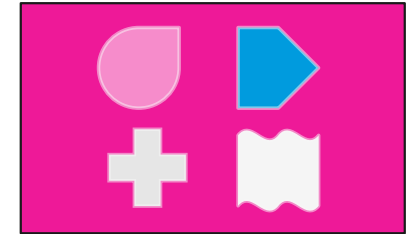
WonderThing



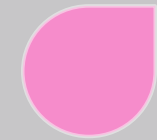
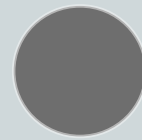
SuperGizmo



GameChanger



API Management



Backends on Azure

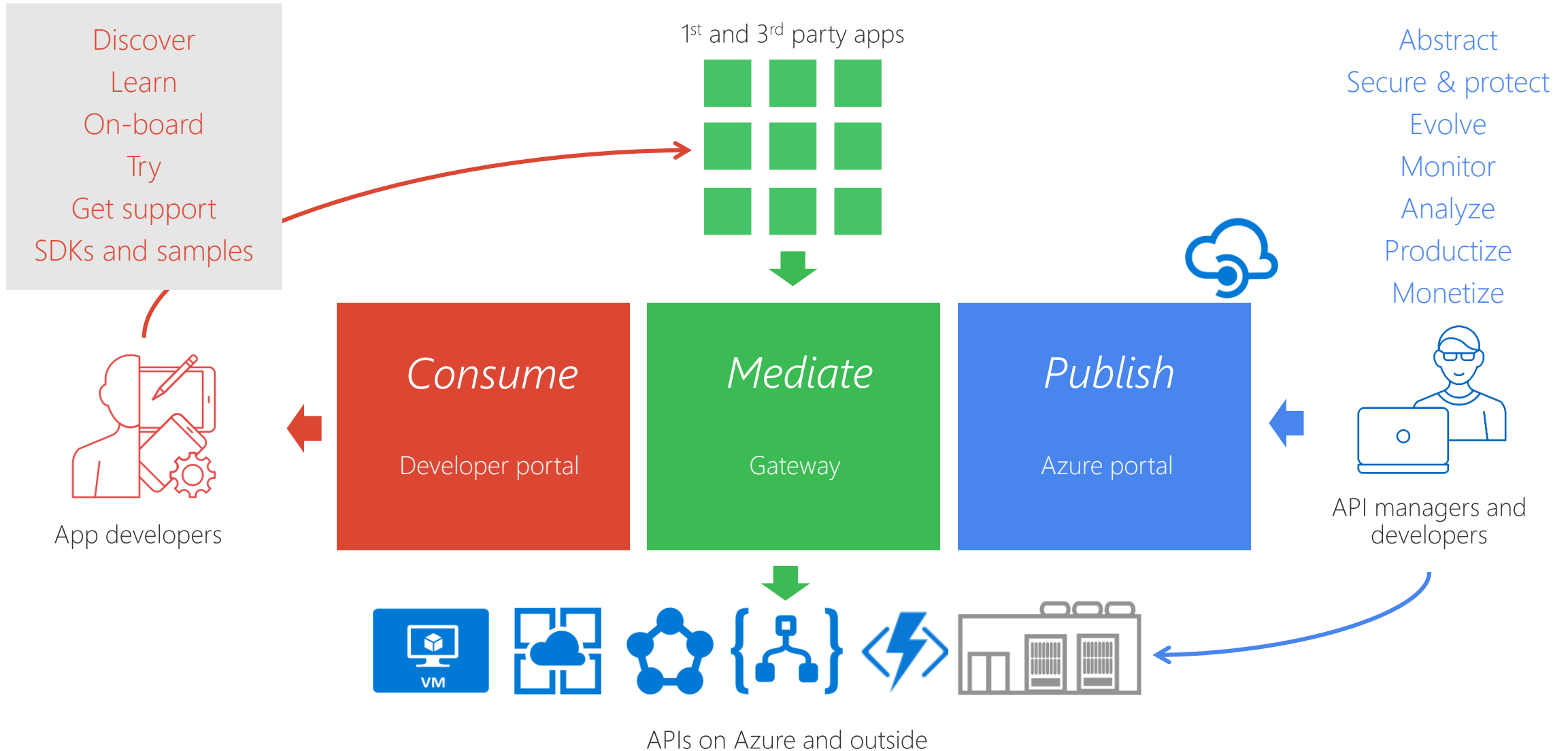


Backends on premises

Why API Management

- Establish a single API “front door”
- Build an API façade for existing backend services
- Add new capabilities to the APIs, such as response caching
- Reliably protect published APIs from misuse and abuse
- Package and publish APIs to developers and partners
- On-board developers via a self-service portal
- Ramp-up developers with docs, samples, and API console
- Gain insights into API usage and health from analytics reports

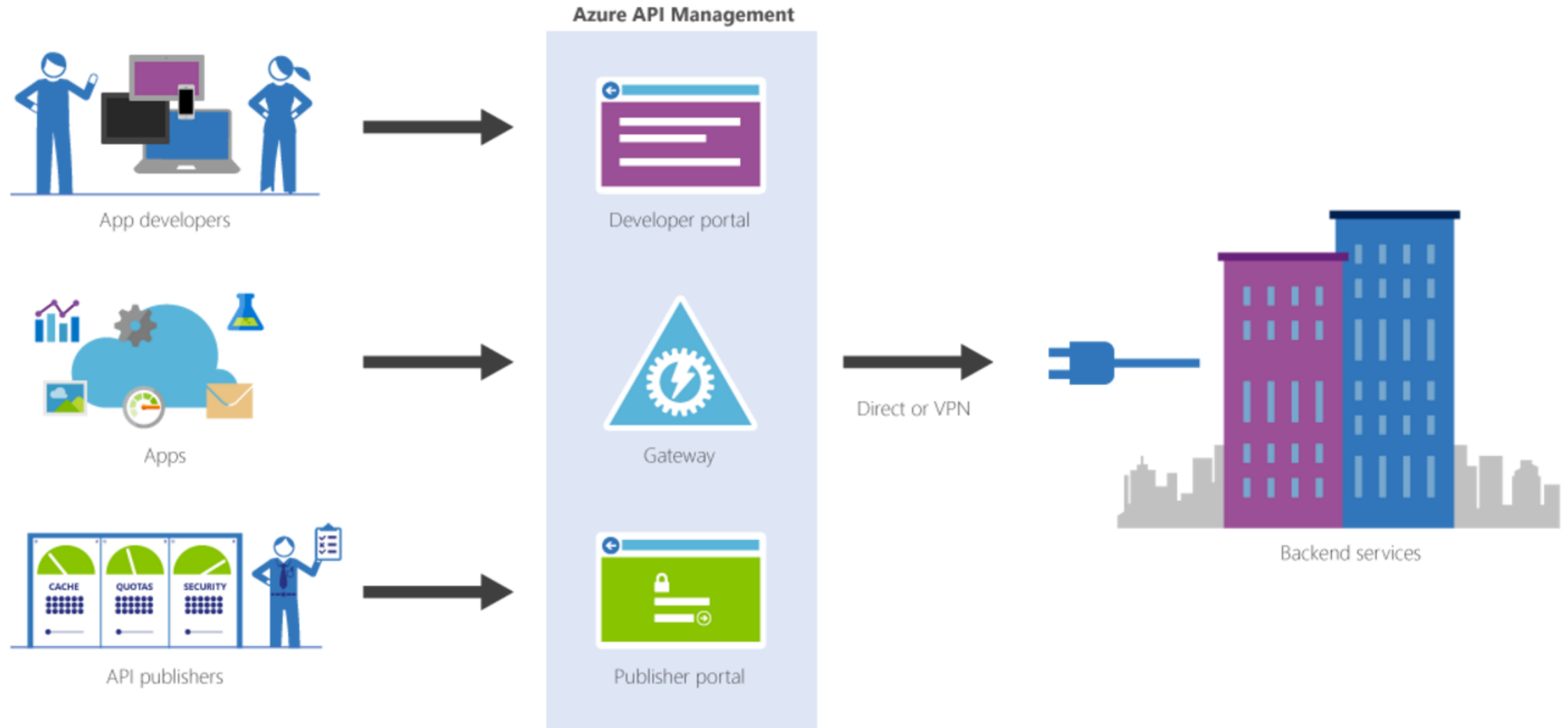
API Management - a hub for enterprise APIs



Why API Management

- Azure API Management can take any backend and launch a full-fledged API program based on it.
- Common scenarios include:
 - **Securing mobile infrastructure** - Gating access, preventing DOS attacks
 - **Enabling ISV partner ecosystems** - Developer Portal and building API facade
 - **Running an internal API program** – Centralized location and gating access

Azure API Management



Provide first-rate developer experience

- Self-service API key management
- Auto-generated API catalogue, documentation and code samples
- OAuth-enabled API console for exploring APIs without writing a line of code
- Sign in using popular Internet identity providers and Azure Active Directory

Protect and optimize your APIs

- Simplify and optimize requests and responses with transformation policies
- Secure APIs with key, JWT token validation and IP filtering
- Protect your APIs from overload and overuse with quotas and rate limits
- Use response caching for improved latency and scale

Manage all your APIs in one place

- Expose all APIs behind a single static IP and domain
- View near real-time usage, performance and health analytics
- Automate management and integrate using REST API, PowerShell and Git
- Provision API Management and scale it on demand in one or more geographical regions

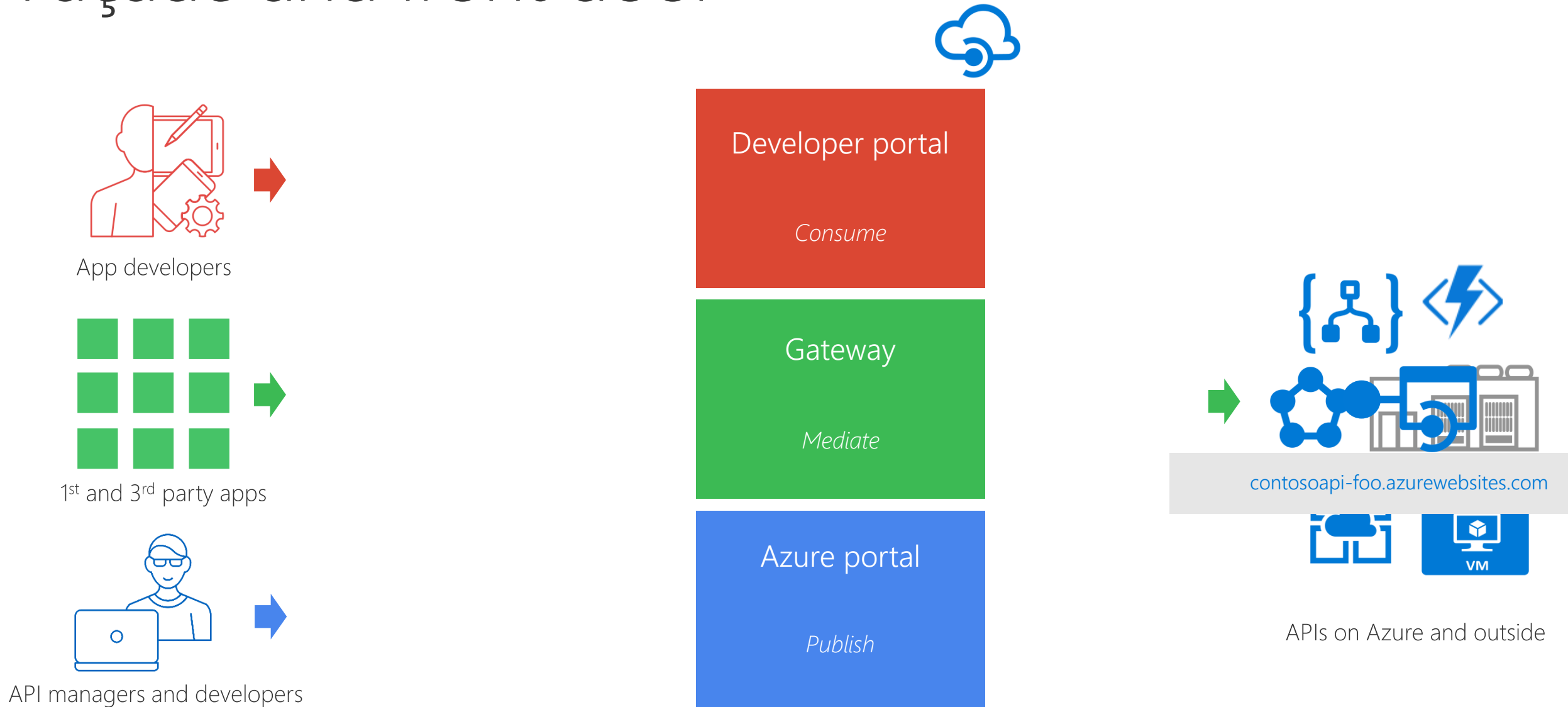
Azure API Management – Scenarios

- Securing mobile infrastructure
 - By gating access with API keys, preventing DOS attacks by using throttling, or using advanced security policies like JWT token validation.
- Enabling ISV partner ecosystems
 - By offering fast partner onboarding through the developer portal and building an API facade to decouple from internal implementations that are not ripe for partner consumption.
- Running an internal API program
 - By offering a centralized location for the organization to communicate about the availability and latest changes to APIs, gating access based on organizational accounts, all based on a secured channel between the API gateway and the backend.

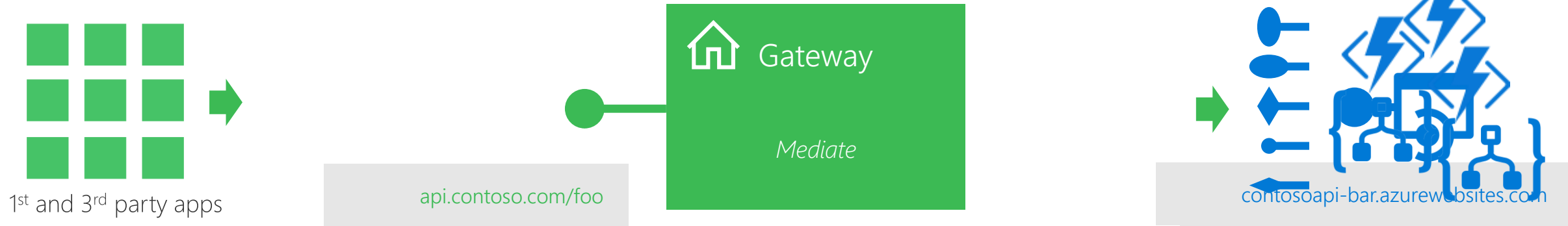
Complete Walkthrough Steps

- Develop ASP.Net Web API Project
- Host/Deploy to Azure
- Create instance of API Management
- Add Created API
- Add Operations to API
- Create subscribers
- Test APIs

Façade and front door



Façade and front door



Configuring Policies

Policies

- Encapsulate common API management functions
 - Access control, Protection, Transformation, Caching, ...
- Chained together into a pipeline
- Mutate request context or change API behavior
- Set in the inbound and outbound directions
- Can be triggered on error
- Applied at a variety of scopes

Access restriction policies

- + Check HTTP header
- + Limit call rate per key
- + Limit call rate per subscription
- + Restrict caller IPs
- + Set usage quota per key
- + Set usage quota per subscription
- + Validate JWT

Advanced policies

- + Control flow
- + Forward request to backend service
- + Log to EventHub
- + Output trace information
- + Retry
- + Return response
- + Send one way request

Calculate effective policy

Configuring Policies at product or API or Operation level

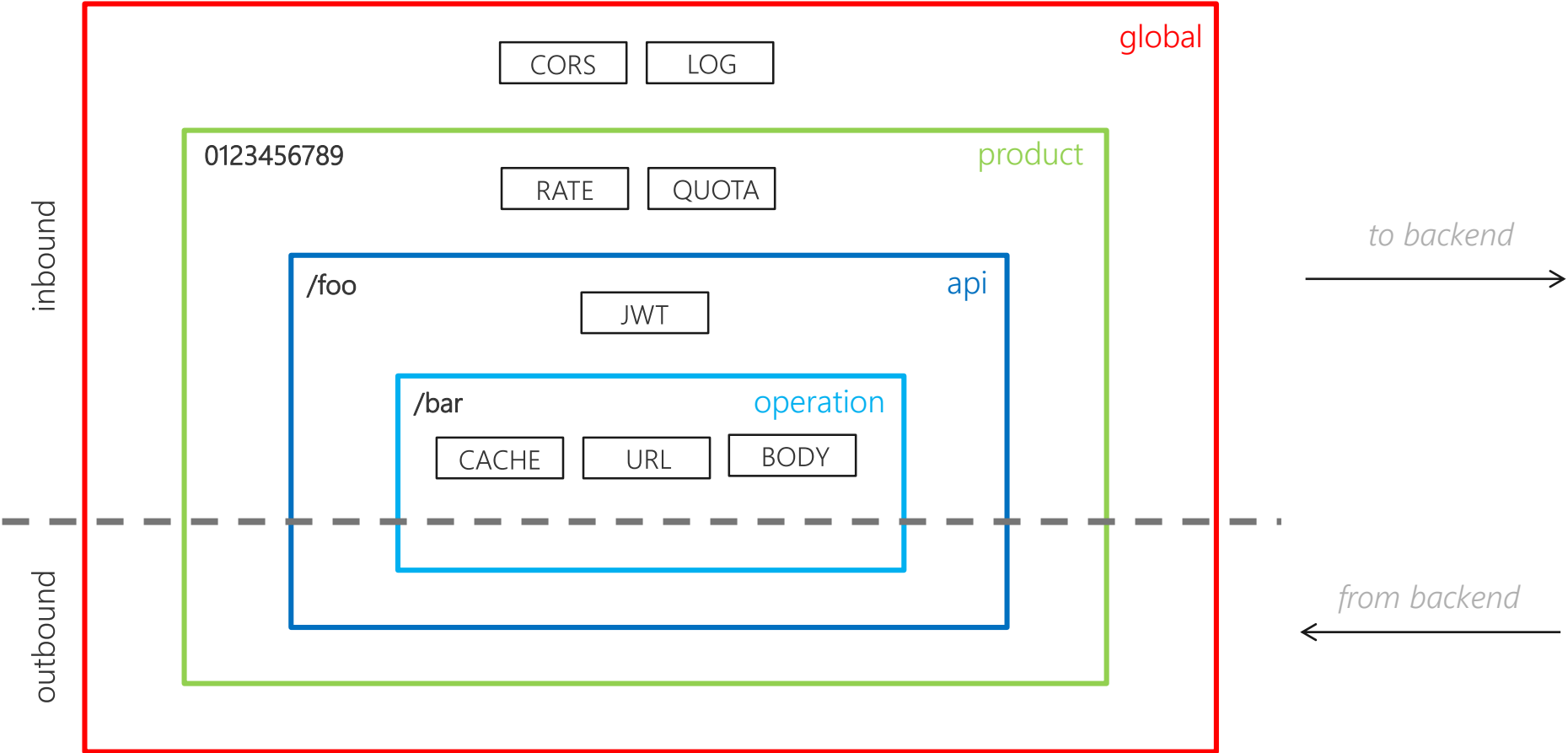
- Policies are applied inside the gateway which sits between the API consumer and the managed API.
- The gateway receives all requests and usually forwards them unaltered to the underlying API. However a policy can apply changes to both the inbound request and outbound response.
- Policy expressions can be used as attribute values or text values in any of the API Management policies, unless the policy specifies otherwise. Some policies such as the [Control flow](#) and [Set variable](#) policies are based on policy expressions.

Policy scopes

GET /foo/bar HTTP/1.1
Host: api.constoso.com
Key: 0123456789

from caller
→

←
to caller



Policy expressions

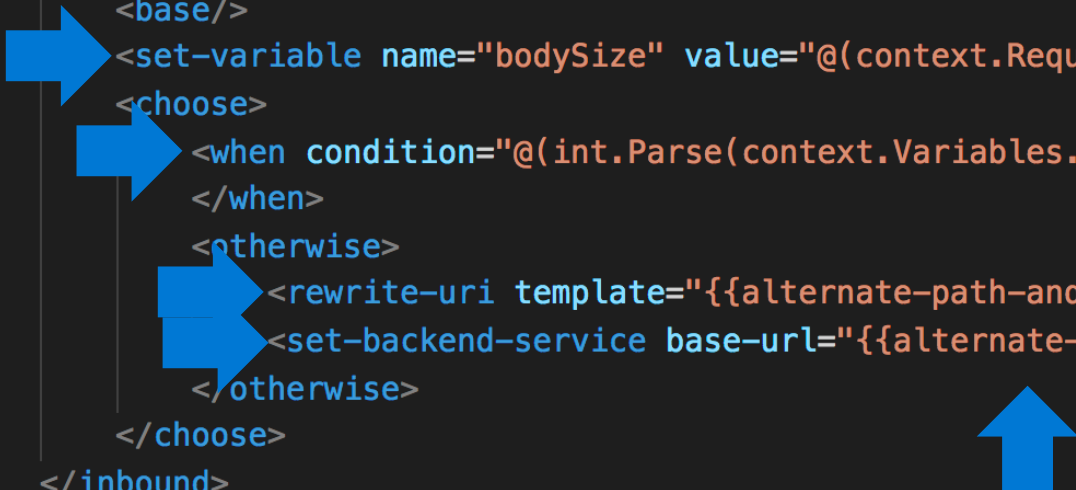
C# "snippets" embedded in policy documents

Have read-only access to the request context

Can only use whitelisted .NET types

Dynamically configure and conditionally execute policies

```
<policies>
  <inbound>
    <base/>
    <set-variable name="bodySize" value="@context.Request.Headers["Content-Length"][0]"/>
    <choose>
      <when condition="@int.Parse(context.Variables.GetValueOrDefault<string>("bodySize")) < 256*1024)">
        <rewrite-uri template="{{alternate-path-and-query}}"/>
        <set-backend-service base-url="{{alternate-host}}"/>
      </otherwise>
    </choose>
  </inbound>
  <outbound>
    <base/>
  </outbound>
</policies>
```



The diagram illustrates the structure of a policy document. It features a dark background with XML-like code. Blue arrows highlight specific parts of the code: one points to the `<set-variable>` element, another to the `<when>` element, a third to the `<rewrite-uri>` element, a fourth to the `<set-backend-service>` element, and a fifth points upwards towards the `<choose>` block.

Versioning APIs

Versioning is a highly debated subject

Version or not?

Semantic versioning?

What is a breaking change?

Where to place version information?

Path? Query? Header? Media type?

How to identify version?

Number? Date? Name?

Our approach to versioning

Natively understand versions at the system level

Versioning is an opt-in

Offer versioning scheme options

Inform developers about the changes

Control when the changes get adopted



Versions

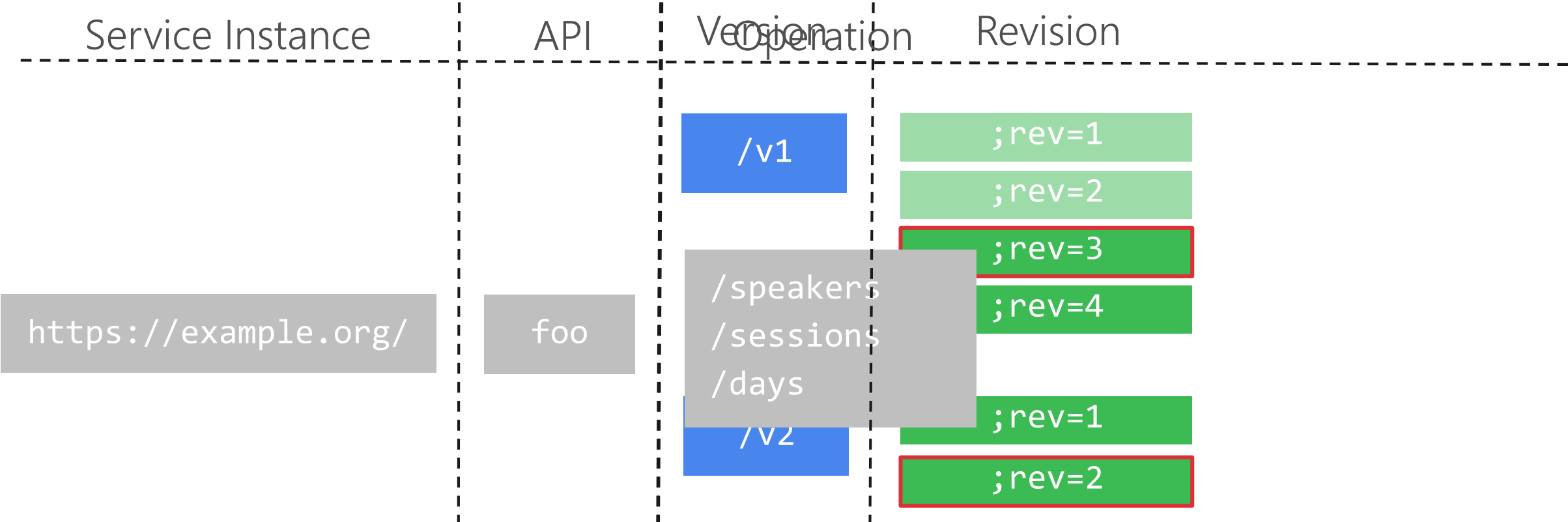
Consumers choose
when to adopt



Revisions

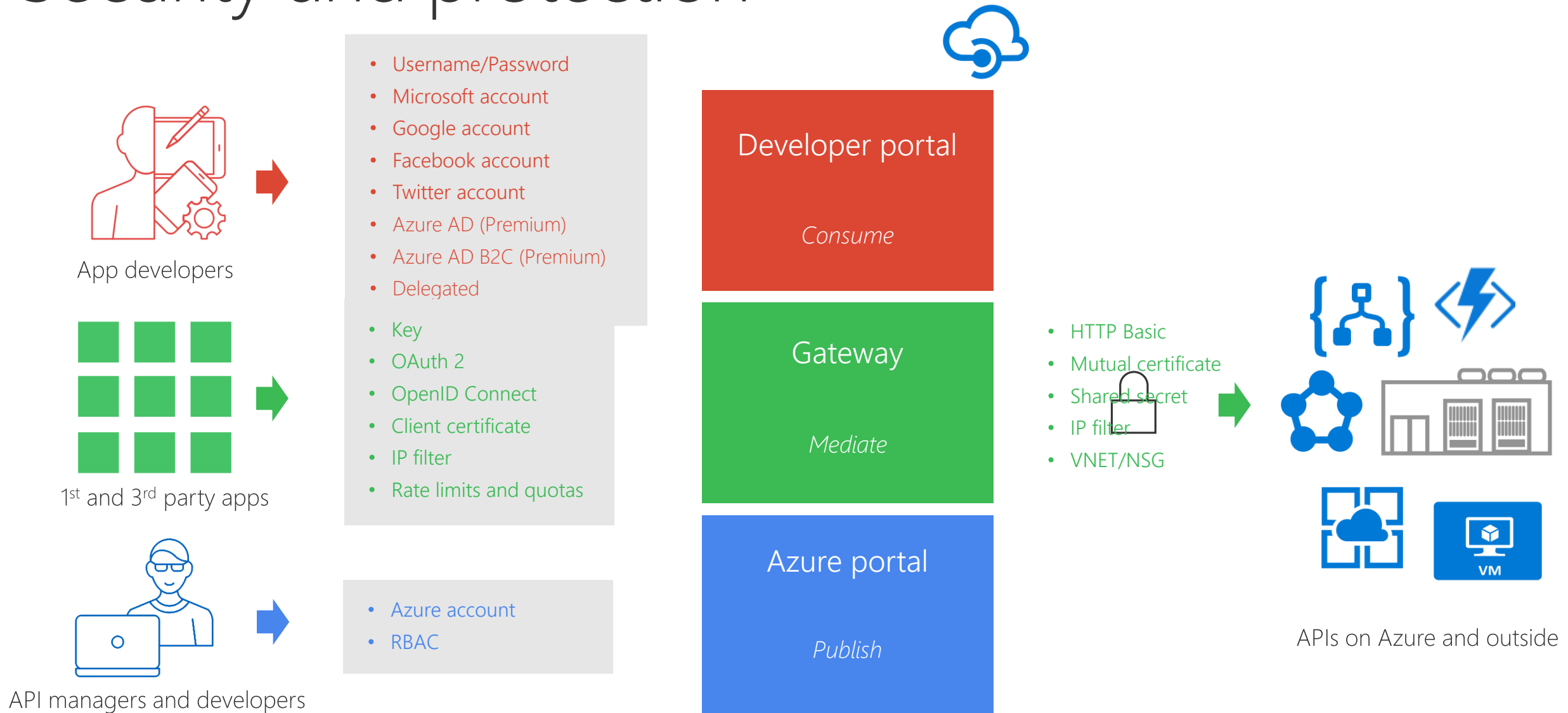
Providers choose
when to deploy

Versions and revisions in API Management

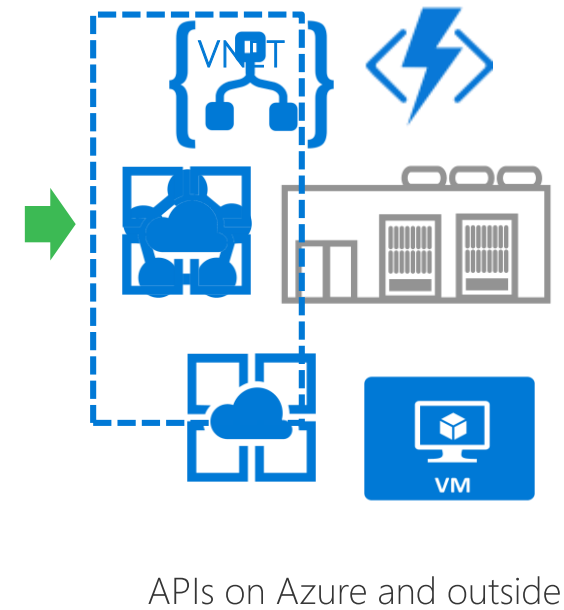
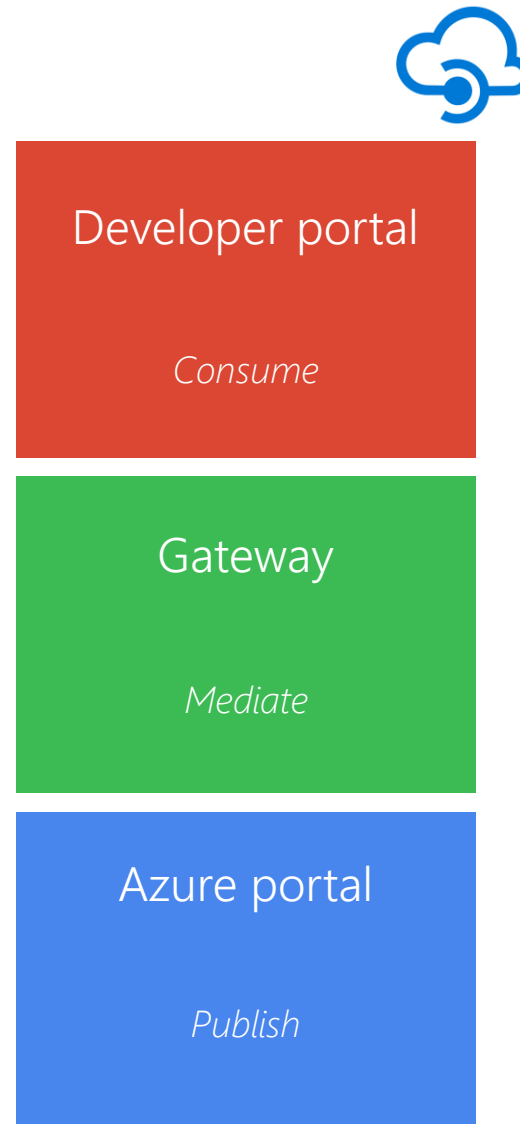
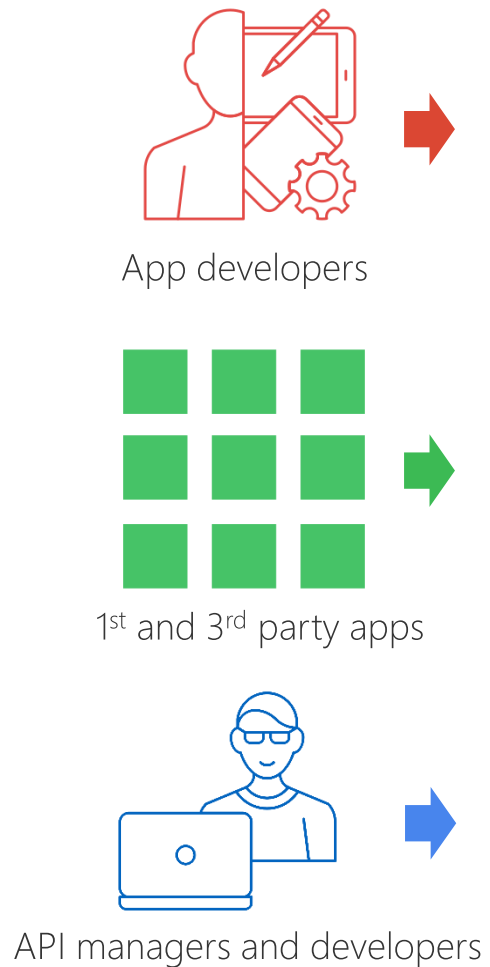


Security

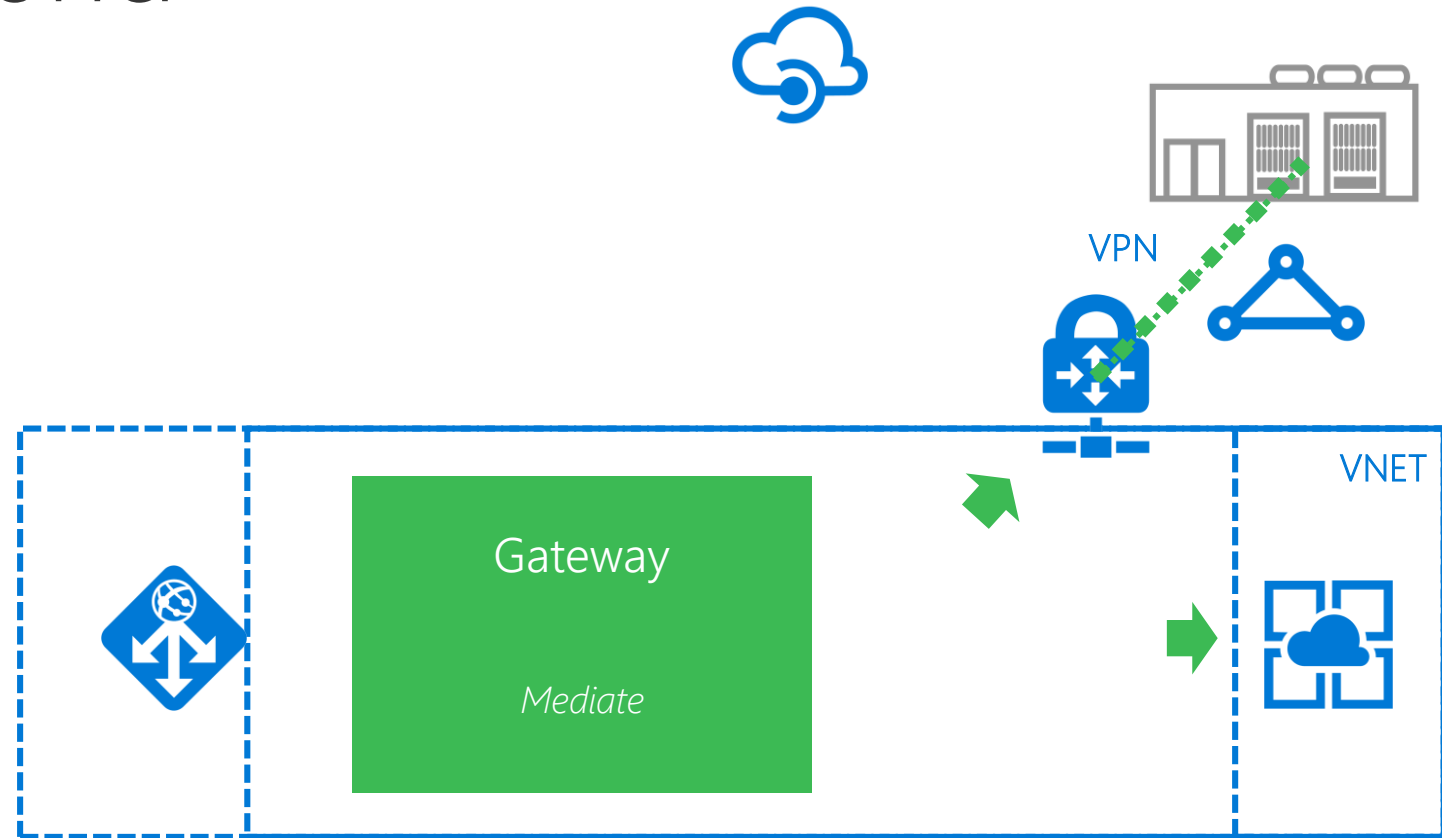
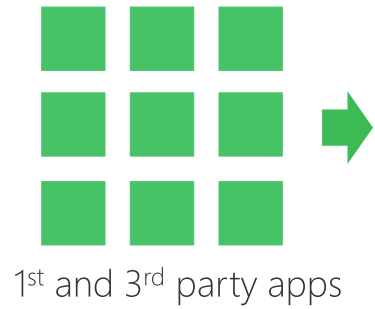
Security and protection



VNETs and Hybrid



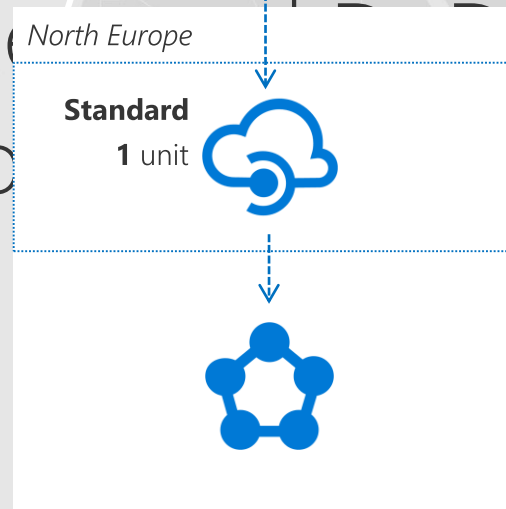
VNETs and Hybrid



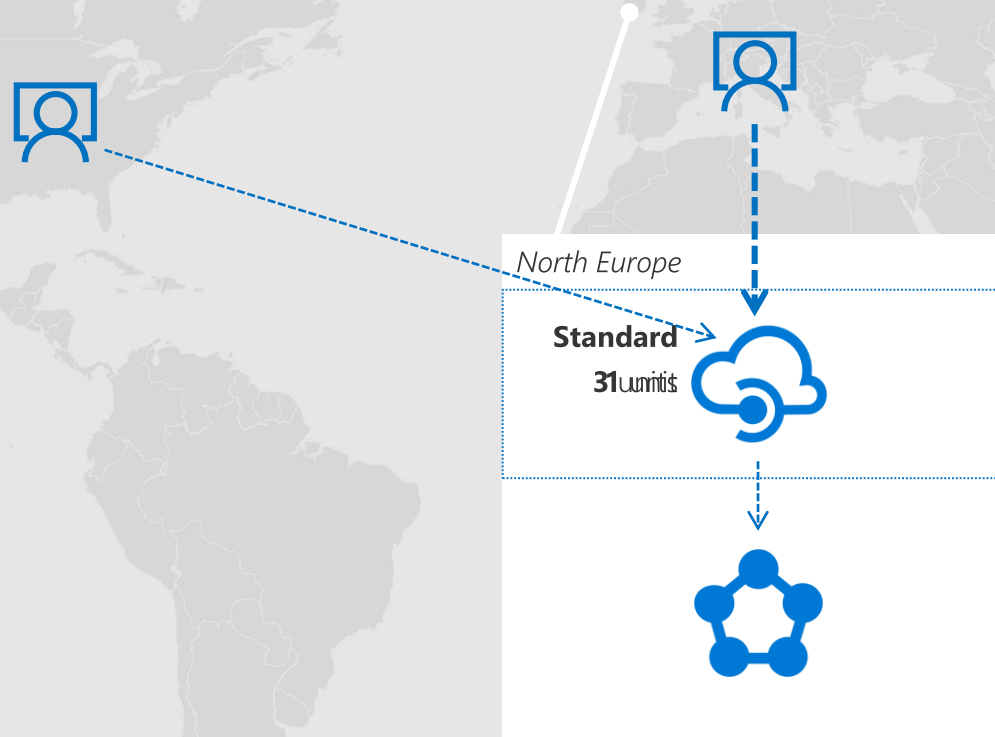
Scaling and Global access

Multi-region and scaling

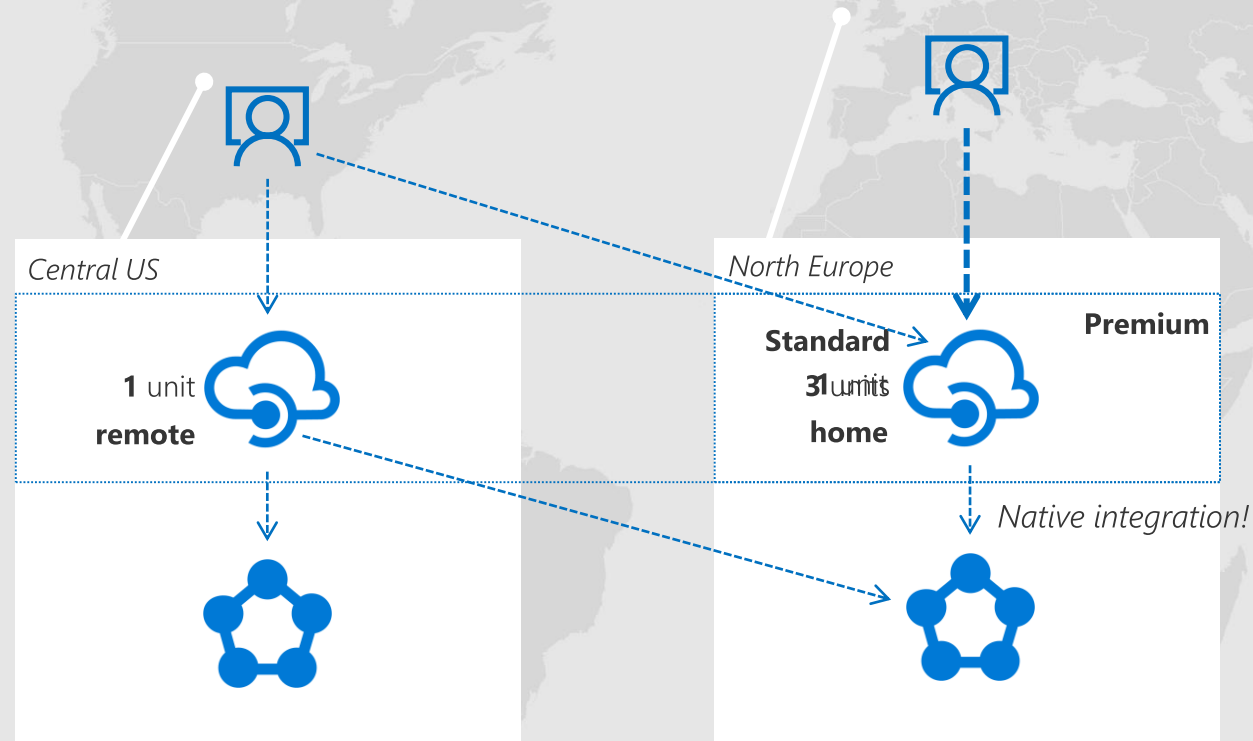
27 public regions in Americas, Europe, Asia and Australia
6 US government regions (preview)
Social Media regions (preview)
China!



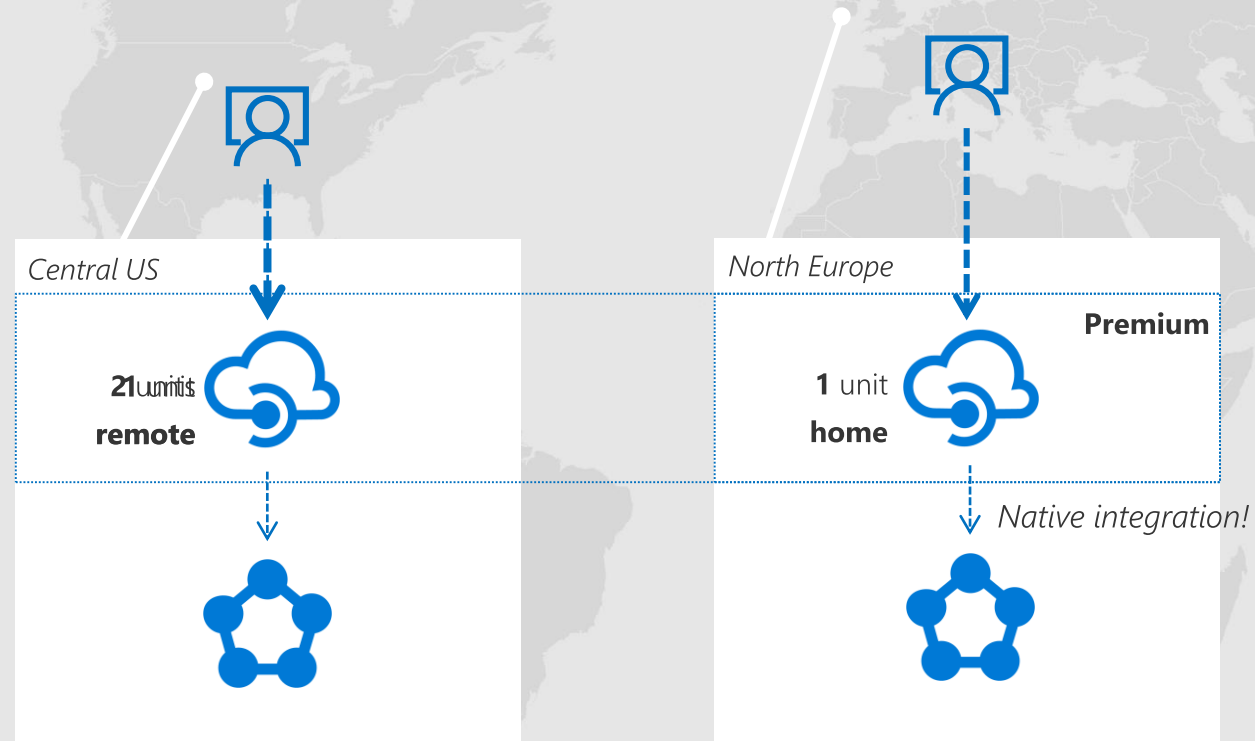
Multi-region and scaling



Multi-region and scaling



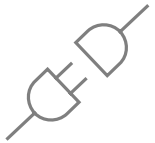
Multi-region and scaling



Azure API Management



Cloud hosted, turnkey, fully managed



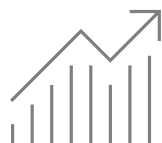
Works with APIs running in the cloud or on-prem



Abstracts, protects and optimizes APIs



Promotes and supports app developer engagement



Provides API governance, insights, and analytics

Thank you

sonusathyadas@synergetics-india.com