



How to secure your precious APIs with Azure API Management

Vladimir Vinogradsky

/in/vladvino

@vladvino

Agenda

API Management 101

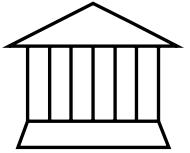
API security

Mitigations

Network security

Wrap-up and Q&A

API management distilled



Façade

[Hide backends from frontends](#)

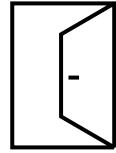
Aggregate or slice

Modernize or normalize

Re-architect

Move

Mock



Front door

[Single point of ingress](#)

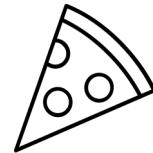
Route

Secure

Throttle

Transform

Observe



Frictionless consumption

[Self-service user onboarding](#)

Discover

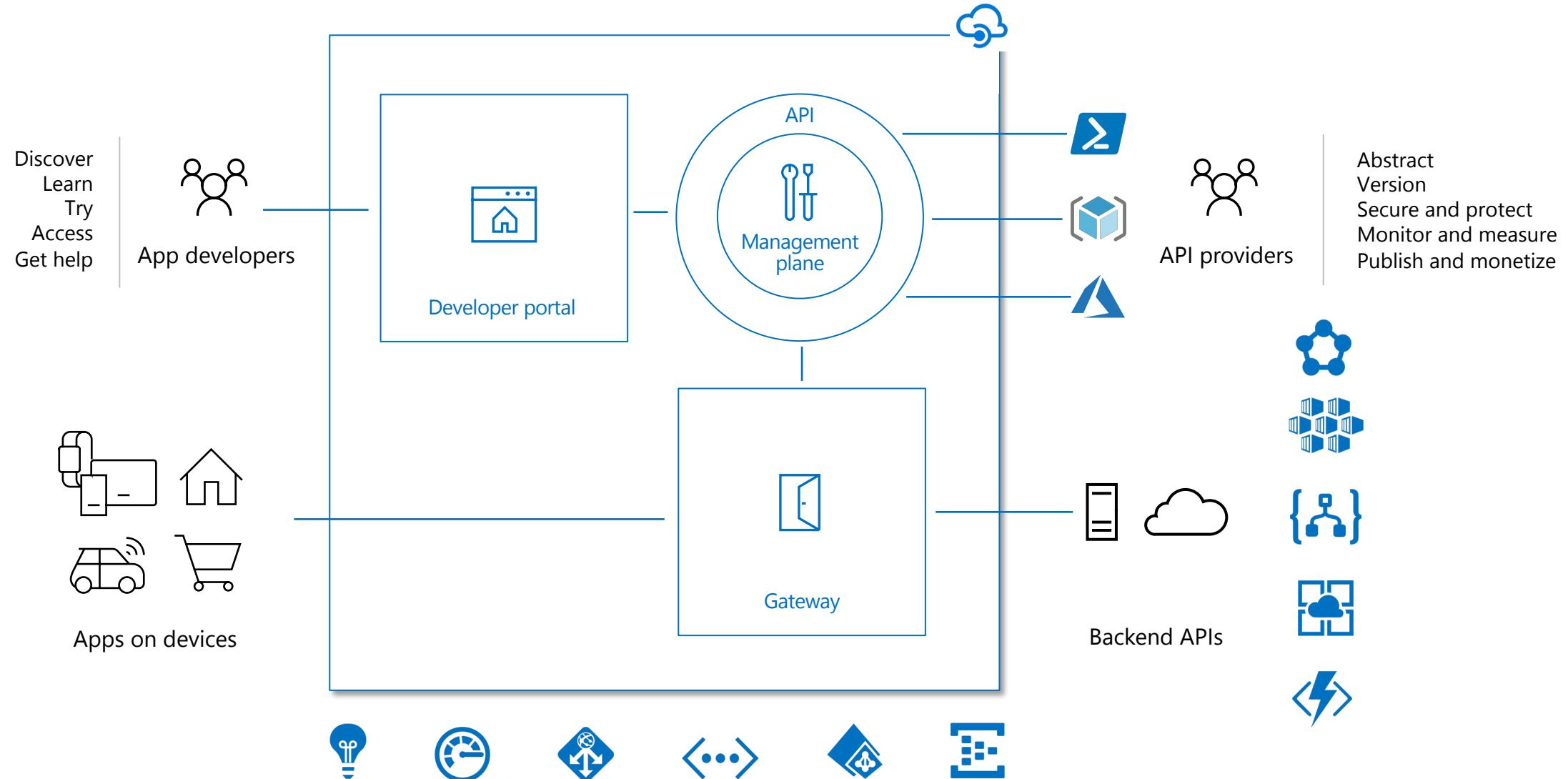
Learn

Try

Request access

Get started

Azure API Management



Cross domain policies

+ Allow cross domain calls

+ CORS

+ JSONP

There is a policy for that

Encapsulate common API management functions

Access control, Protection, Transformation, Caching, ...

Mutate request context or change API behavior

E.g. add a header or throttle

Set in the inbound and outbound directions

Applied at a variety of scopes or on error

Scope determines which APIs are affected

Can define custom scopes in addition to four available by default

Composed into a pipeline from effective scopes

Degree of control over inheritance of scopes, i.e. <base/> element

Don't delete <base/> inadvertently

Authentication policies

+ Authenticate with Basic

+ Authenticate with client certificate

Access restriction policies

+ Check HTTP header

+ Limit call rate per key

+ Limit call rate per subscription

+ Restrict caller IPs

+ Set usage quota per key

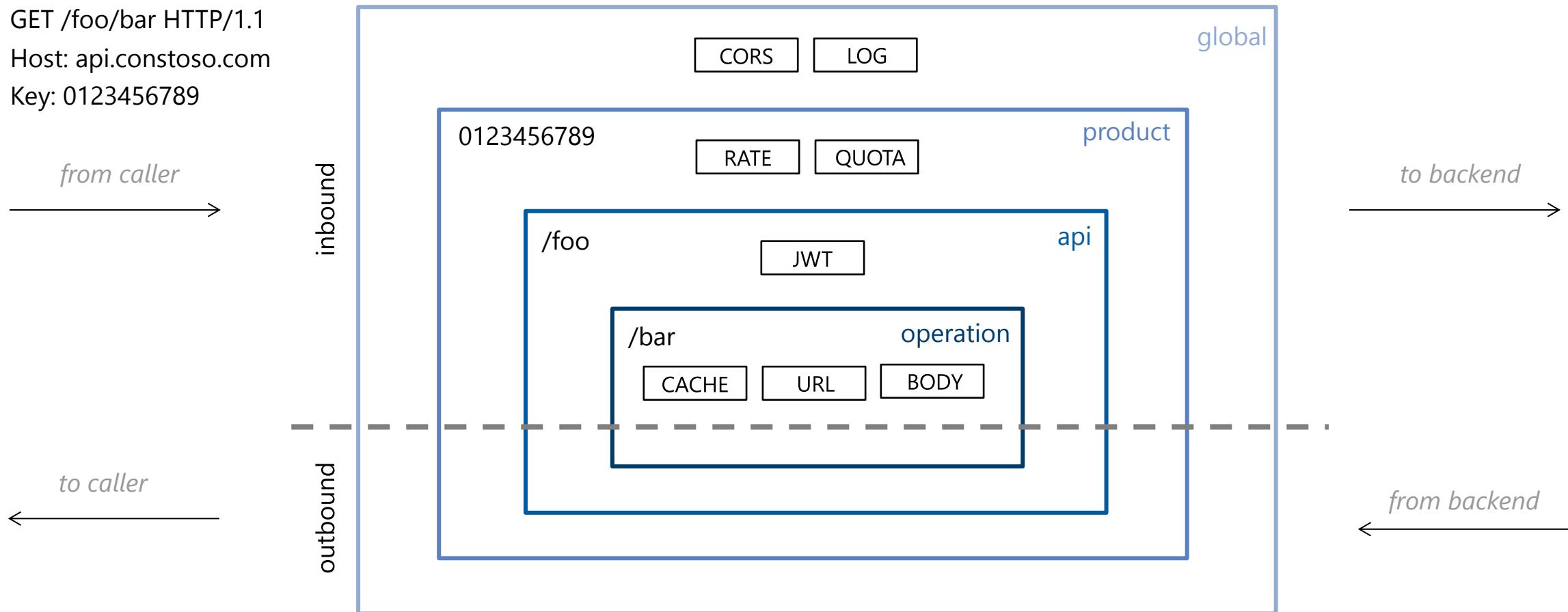
+ Set usage quota per subscription

+ Validate JWT

Calculate effective policy

Policy scopes

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



Policy expressions

- Are C# “snippets” used with policies
- Have read-only access to the request context
- Can use only whitelisted .NET types
- Are used to configure and conditionally execute policies

Named values

- Are scoped to an API service instance
- Keep secrets and “magic” strings out of policies
- Provide environment-specific values
- Add semantics, if named well
- Enable a single point of change

```
1   ...
2   <inbound>
3     <base/>
4     <set-variable name="content-length" value="@(context.Request.Headers["Content-Length"])[0]" />
5     <choose>
6       <when condition="@(int.Parse(context.Variables.GetValueOrDefault<string>("content-length")) > {{max-content-length}})">
7         <rewrite-uri template="{{alternate-path-and-query}}"/>
8         <set-backend-service base-url="{{alternate-host}}"/>
9       </when>
10      </choose>
11    </inbound>
12    ...
```

“By 2021, **90%** of web-enabled applications will have **more** surface area for attack in the form of exposed **APIs** rather than the UI, up from 40% in 2019.”

“By 2022, **API** abuses will move from an infrequent to the **most-frequent attack vector**, resulting in data breaches for enterprise web applications.”

*Source: Gartner, Inc., API Security: What You Need to Do to Protect Your APIs,
Mark O'Neill, Dionisio Zumerle, Jeremy D'Hoinne,
28 August 2019*

"The Open Web Application Security Project (OWASP) is an open community dedicated to enabling organizations to develop, purchase, and maintain applications and APIs that can be trusted"

Learn more at <https://www.owasp.org>



The OWASP logo is a circular icon featuring a stylized white fly or bee on a dark blue background. Below the logo, the word "tm" is visible.

OWASP™ Foundation
the free and open software security community

[Home](#) [About OWASP](#) [Acknowledgements](#) [Advertising](#) [Books](#) [Brand Resources](#) [Careers](#) [Chapters](#) [Donate to OWASP](#) [Downloads](#) [Events](#) [Funding](#) [Governance](#) [Initiatives](#) [Mailing Lists](#) [Membership](#) [Merchandise](#) [Presentations](#) [Press](#) [Projects](#) [Supporting Partners](#) [Video](#)

[Reference](#) [Activities](#) [Attacks](#) [Code Snippets](#) [Controls](#)

[Member Portal](#) [About](#) [Searching](#) [Editing](#) [New Article](#) [OWASP Categories](#) [Contact Us](#)

[DONATE](#) [OWASP DONATION PORTAL](#)

[Search](#)

[Log in](#) [Request account](#)

[Statistics](#) [Recent Changes](#)

Every vibrant technology marketplace needs an unbiased source of information on best practices as well as an active body advocating open standards. In the Application Security space, one of those groups is the Open Web Application Security Project™ (or OWASP for short).
The Open Web Application Security Project (OWASP) is a [501\(c\)\(3\)](#) worldwide not-for-profit charitable organization focused on improving the security of software. Our mission is to make software security [visible](#), so that [individuals and organizations](#) are able to make informed decisions. OWASP is in a unique position to provide impartial, practical information about AppSec to individuals, corporations, universities, government agencies, and other organizations worldwide. Operating as a community of like-minded professionals, OWASP issues software tools and knowledge-based documentation on application security.
Everyone is free to participate in OWASP and [all of our materials](#) are available under a free and open software license. You'll find everything [about OWASP](#) here on or linked from our wiki and current information on our [OWASP Blog](#). **OWASP does not endorse or recommend commercial products or services**, allowing our community to remain vendor neutral with the collective wisdom of the best minds in software security worldwide.
We ask that the community look out for [inappropriate](#) uses of the OWASP brand including use of our name, logos, project names, and other trademark issues.
There are thousands of [active wiki users](#) around the globe who review the changes to the site to help ensure quality. If you're new, you may want to check out our [getting started](#) page. As a global group of volunteers with over 45,000

 [Citations](#)

 [OCoC](#)

Who Trusts OWASP?
Citations of National & International Legislation, Standards, Guidelines, Committees and Industry Codes of Practice
[- Click Here](#)

How can OWASP help your org?
[Government Bodies](#) [Educational Institutions](#) [Standards Groups](#) [Trade Organizations](#) [Certifying Bodies](#) [Development Organizations](#)

Security101
Ask a software security question on our Slack channel - [open to all, experts to beginners](#)

API vs. web app security

#	OWASP Top 10 (2017)	OWASP API Top 10 (2019)
1	Injection	Broken Object Level Authorization
2	Broken Authentication	Broken Authentication
3	Sensitive data exposure	Excessive Data Exposure
4	XML External Entities (XXE)	Lack of Resources & Rate Limiting
5	Broken Access control	Broken Function Level Authorization
6	Security misconfigurations	Mass assignment
7	Cross Site Scripting (XSS)	Security misconfigurations
8	Insecure Deserialization	Injection
9	Using Components with known vulnerabilities	Improper Assets Management
10	Insufficient logging and monitoring	Insufficient logging and monitoring

API Management to the rescue

#	OWASP API Top 10 (2019)	Mitigations
1	Broken Object Level Authorization	<i>Area of investment</i>
2	Broken Authentication	Throttling; Request transformation; Token validation
3	Excessive Data Exposure	Filtering or masking sensitive data
4	Lack of Resources & Rate Limiting	Throttling; Limiting backend concurrency
5	Broken Function Level Authorization	Key/token/certificate-based authorization; Custom authorization
6	Mass assignment	<i>Area of investment</i>
7	Security misconfigurations	Restricting attack surface; TLS enforcement and configuration; CORS; Sanitization of response headers and error messages;
8	Injection	<i>Area of investment</i>
9	Improper Assets Management	Up-to-date API catalog; API lifecycle management
10	Insufficient logging and monitoring	Enforceable, customizable logging and tracing.

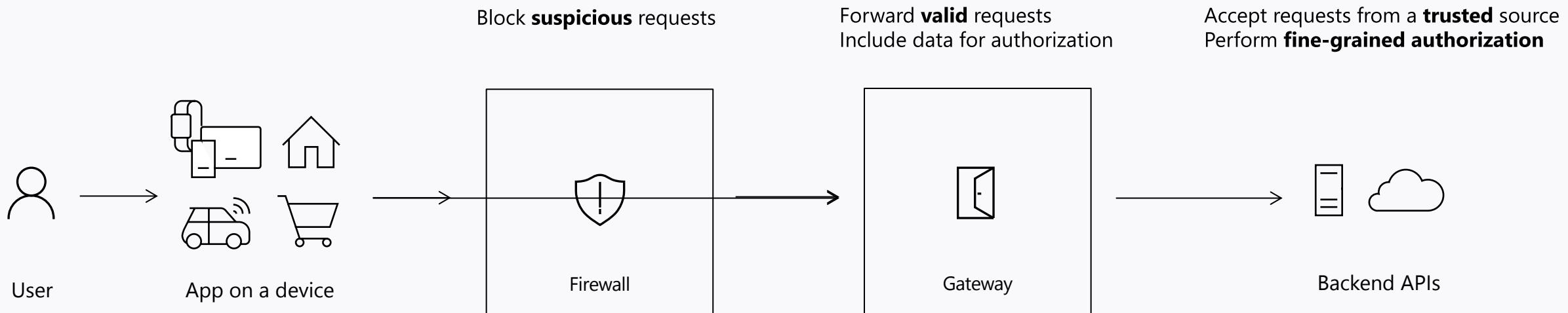
API Management to the rescue

#	OWASP API Top 10 (2019)	Mitigations
1	Broken Object Level Authorization	<i>Area of investment</i>
2	Broken Authentication	Throttling ; Request transformation; Token validation
3	Excessive Data Exposure	Filtering or masking sensitive data
4	Lack of Resources & Rate Limiting	Throttling ; Limiting backend concurrency
5	Broken Function Level Authorization	Key/token/certificate-based authorization; Custom authentication and authorization.
6	Mass assignment	<i>Area of investment</i>
7	Security misconfigurations	Restricting attack surface; TLS enforcement and configuration; CORS; Sanitization of response headers and error messages
8	Injection	<i>Area of investment</i>
9	Improper assets management	Up-to-date API catalog; API lifecycle management
10	Insufficient logging and monitoring	Enforceable, customizable logging and tracing

Layered defense

Defense in depth options

Comprehensive authN/authZ in every layer
Separation of concerns between the layers



Key
OAuth 2 & OpenID Connect
Client certificate
Custom authN/authZ
IP filter
Rate limits and quotas

HTTP Basic
Mutual certificate
Managed identity
IP filter

Disclaimers

Using API Management doesn't automatically guarantee security

I am not a security expert

Façade

- Expose specific backend resources
- Limit HTTP methods on exposed resources
- Enforce HTTPS and TLS configuration
- CORS

Response sanitization



- Filter or mask confidential data
- Standardize error messages
- Remove sensitive headers

Throttling

Rate limit

Approximate

Per region

Key expression defines throttling semantics

Can count requests with specific status code

Variable increment count

Quota

Calls and data transfer

Approximate

Per service

Key expression defines throttling semantics

Can count requests with specific status code

Variable increment count

Concurrency limit

Precise

Per node

Keys

On by default – prevent access to APIs
UUIDs by default – hard to guess
Used to identify and authorize apps
Security-wise roughly equivalent to Basic

JWT



Signed (JWS) and encrypted (JWE)
Validate via policy and expressions
Enforce claims
Require signatures and expiration time
Provide keys inline or via a discovery endpoint

Key vs. JWT

	Key	JWT
Credential type	Bearer	Bearer
Token type	Reference	Self-contained
Size	Immaterial	Material*
Protection	Transport	Transport/Signature/Encryption
Scope	All-or-nothing	Fine-grained
Expiration	External	Built in
Revocation	Yes	Refresh tokens only*
Subject	App	App or User

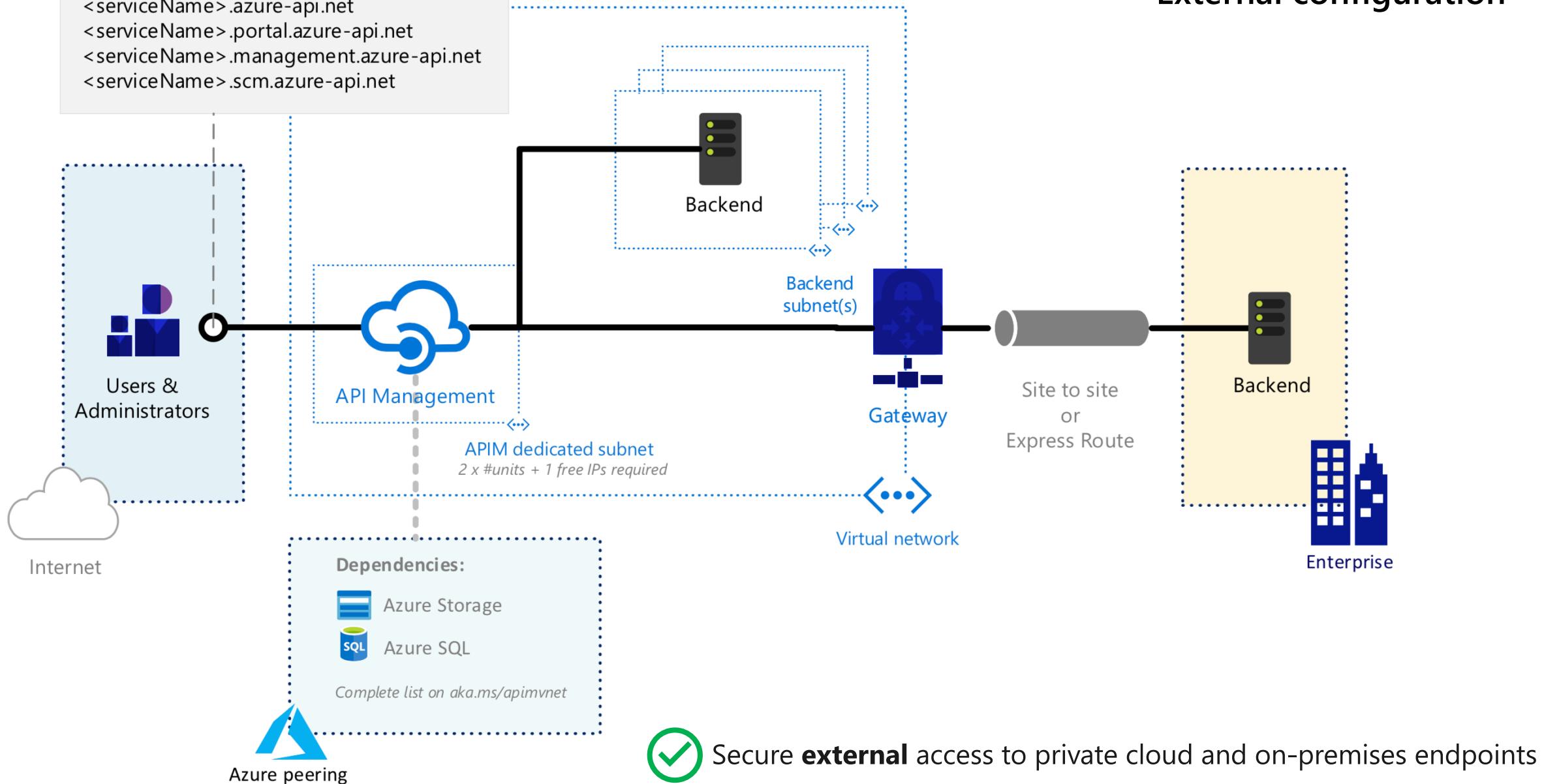
Client certificates

- Trusted and untrusted
- Validate via expressions
- Require certificate on per host basis
- Check or ignore revocation lists

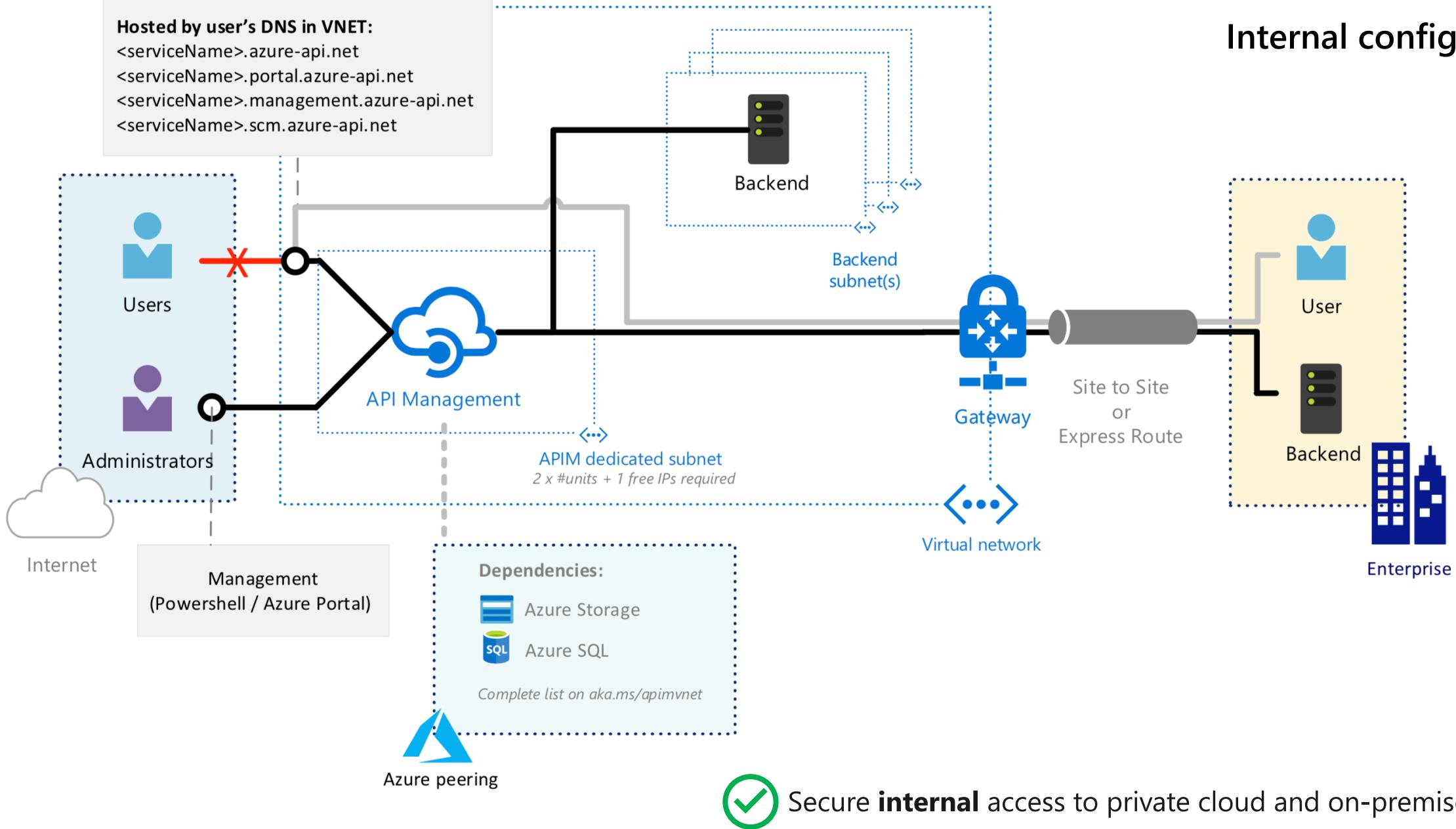
Custom authentication and authorization

- Integrate with a bespoke or not supported identity or authorization system
- Call out to an external HTTPS endpoint
- Cache the result for efficiency

External configuration



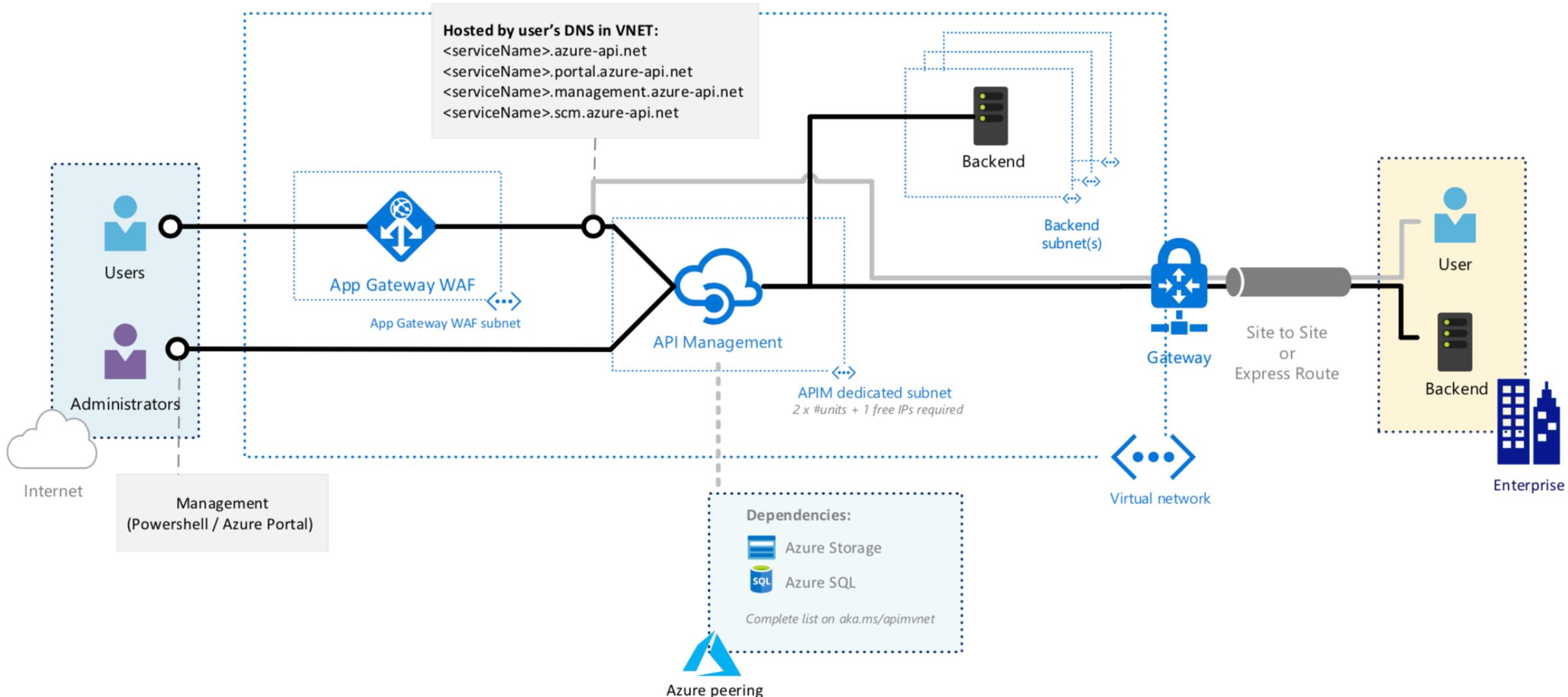
Internal configuration

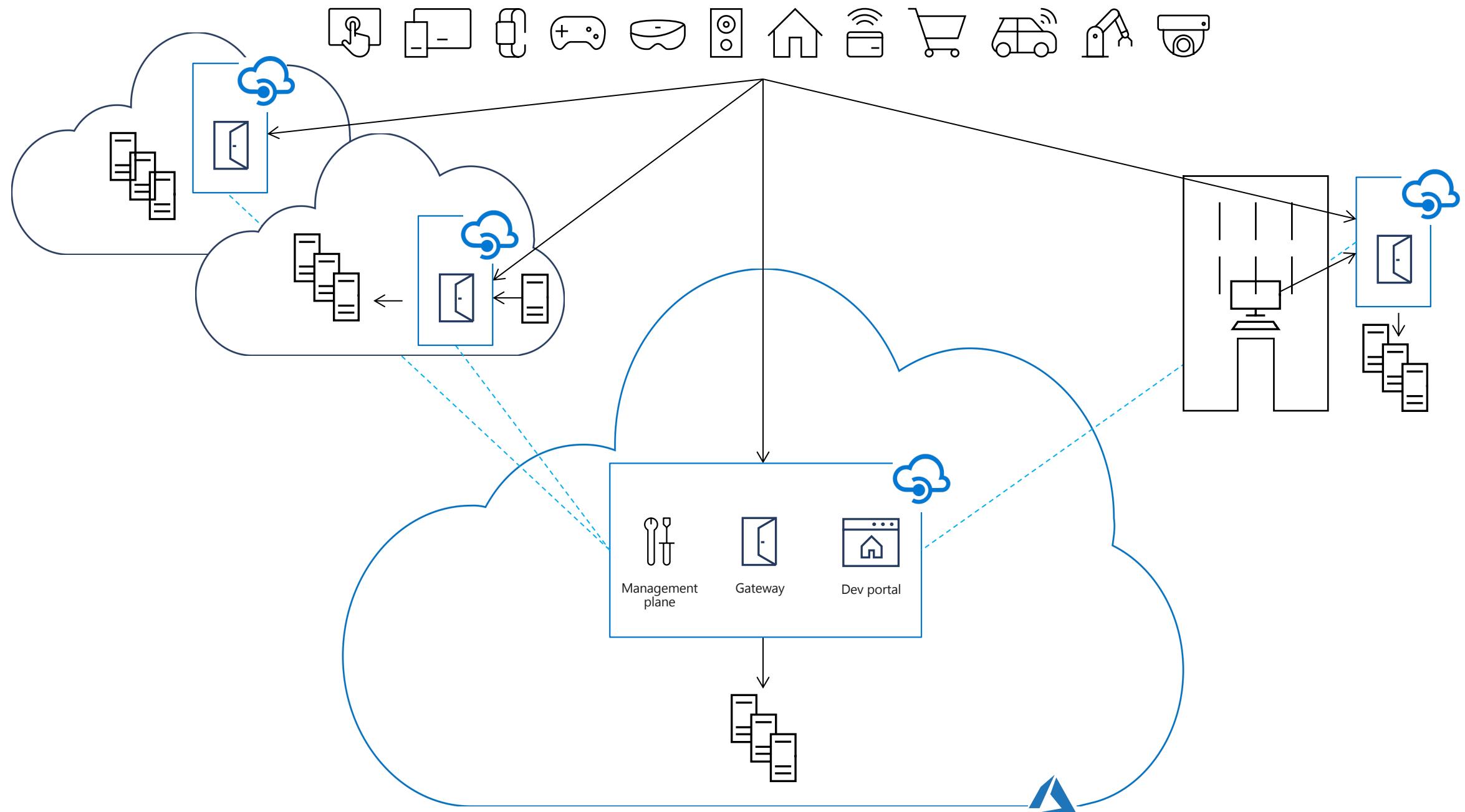




More secure **external** access to private and on-premises endpoints
Secure **internal** access to private cloud and on-premises endpoints

Internal configuration with WAF





Self-hosted API Management gateway PREVIEW



Hosted on premises or in a cloud

Packaged as a Linux-based Docker container image
Functionally equivalent to the managed gateway



Federated with an API Management service

Gateway pulls down configuration and pushes up telemetry
Gateway requires only outgoing connectivity to Azure



Deployed to Docker or Kubernetes

K8s simplifies deployment, scaling, updates, availability
Other container orchestrators can be used as well



Information

<https://aka.ms/apimlove>

