Microsoft

Azure

Azure Event Hubs

The Architect's Cut

Clemens Vasters

Chief Messenger, **Twitter: @clemensv**

Patterns

Features

Architecture

Internals

Agenda

A *very* brief overview: “What is Azure Event Hubs?”

The Event Hubs Deep Dive

Clusters and Namespaces

Networking Capabilities

Authorization Models

Partitions and Log Storage

Message Broker Compute Model

Federation and Replication

Throughput and Latency

Observed Availability and Reliability

Q&A

What is Azure Event Hubs?

Platform-as-a-Service Event Stream Broker

Use the Apache Kafka® API, but with far lower cost and better performance.

Fully managed: You use the features, Azure deals with everything else

AMQP 1.0 standards compliant, Apache Kafka® wire-compatible

Polyglot Azure SDK and cross-platform client support

Industry-leading reliability and availability

Fast.

Azure Event Hubs vs. Apache Kafka®

Similar yet very different

	Azure Event Hubs	Apache Kafka
User Model	Partitioned event stream broker with high-availability replication	Partitioned event stream broker with high-availability replication
Architecture	Multi-tenant, 3-Tier Gateway/Broker/Storage cluster model, with tenant-isolation, all tiers independently scalable	Single-tenant monolith. Need to increase broker instances in a cluster to scale any dimension.
Implementation Language	C# and Native (C/C++)	Java
Cluster Manager	Azure Service Fabric (inline)	Apache Zookeeper (external); KRaft (inline, experimental)
Partition Mapping	Key hashing, client or server-side mapping of events	Key hashing, client-side mapping of events
Consumer Partition Ownership Coordination	Server-coordinated partition ownership (Kafka), client-coordinated ownership with external leader election. Parallel, direct partition reads.	Server-coordinated partition ownership
Server Workload Balancing	Dynamic and fully automated (100% hands-off). Broker resource allocation independent of partition count or ownership, flexible scaling.	Static assignment of partitions to broker instances requiring operator intervention for rebalancing.
Storage Model	Replicated log store, synchronous per-message flush-to-disk on all replicas	Replicated log store, asynchronous flush-to-disk controlled by host file system write cache settings.
Networking	Single endpoint access to all partitions, Public IP/DNS or Virtual Networking, Firewall.	Endpoint per broker instance. Multiple IPs required. Complex network management required.
Access Control	Token-based access policy model with unlimited publisher policies, Azure Active Directory role-based access control	Local accounts, federation extensibility.
Protocols	AMQP 1.0 (optional: WebSockets) HTTPS 1.1 Apache Kafka RPC	Apache Kafka RPC
Batching / Archives	Avro-packaged batch-packaging and archival to blob store	
Schema Registry	Schema Registry based on open CNCF Schema Registry API	(Proprietary from commercial vendors)

SKUs and Pricing

Azure Event Hubs *Standard*:

Shared-resources model

Blended capacity and consumption-based pricing.

Up to 40 MB/sec ingress:

- Throughput Units: 1TU = min(1000 msg/s, 1MB/s in), min(2000 msg/s, 2MB/s out)
- Capture
- First 12M events free, 12-25M $x/1M, 25-50M $x/1M, 50+ $x/1M

Azure Event Hubs *Premium*:

Isolated-resources model

Capacity-based pricing model. No consumption charges.

Up to 120 MB/sec ingress:

- Premium Units: 1PU = ~5-10MB/sec throughput, no per-event charge, includes Capture

Azure Event Hubs *Dedicated*:

Dedicated-resources model

Capacity-based pricing model. No consumption charges. Full isolated cluster.

Beyond 120MB/sec ingress (>4 GB/sec in prod)

- Capacity Unit: 1CU $4995/mo.
- 8CU required for availbility zones.

We will discuss all these versions today and what the units mean

You're running Apache Kafka® in your own datacenter?

Azure Event Hubs

Azure Event Hubs is

more reliable and dependable with higher 24/365 uptime
(no service windows or weekend downtime)

far less costly to own
(no initial/recurring licensing fees as with commercial Kafka distributions)

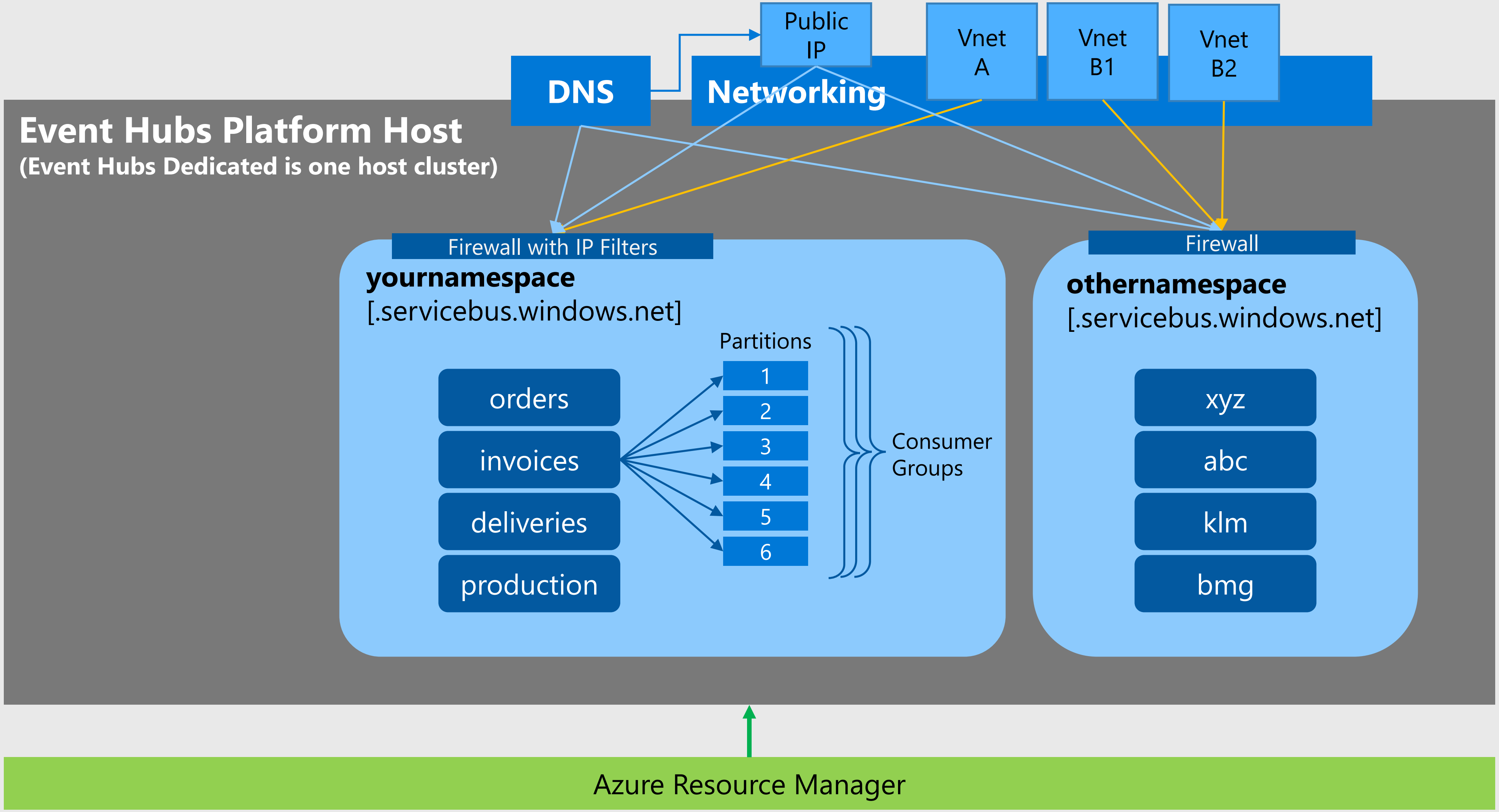
far less costly to operate
(fully managed, no hardware)

just a quick network hop away from your existing workloads

easy to integrate into your network via VPN Gateway or Express Route

Microsoft Azure

Event Hubs Namespaces: Virtual Brokers



Event Hubs

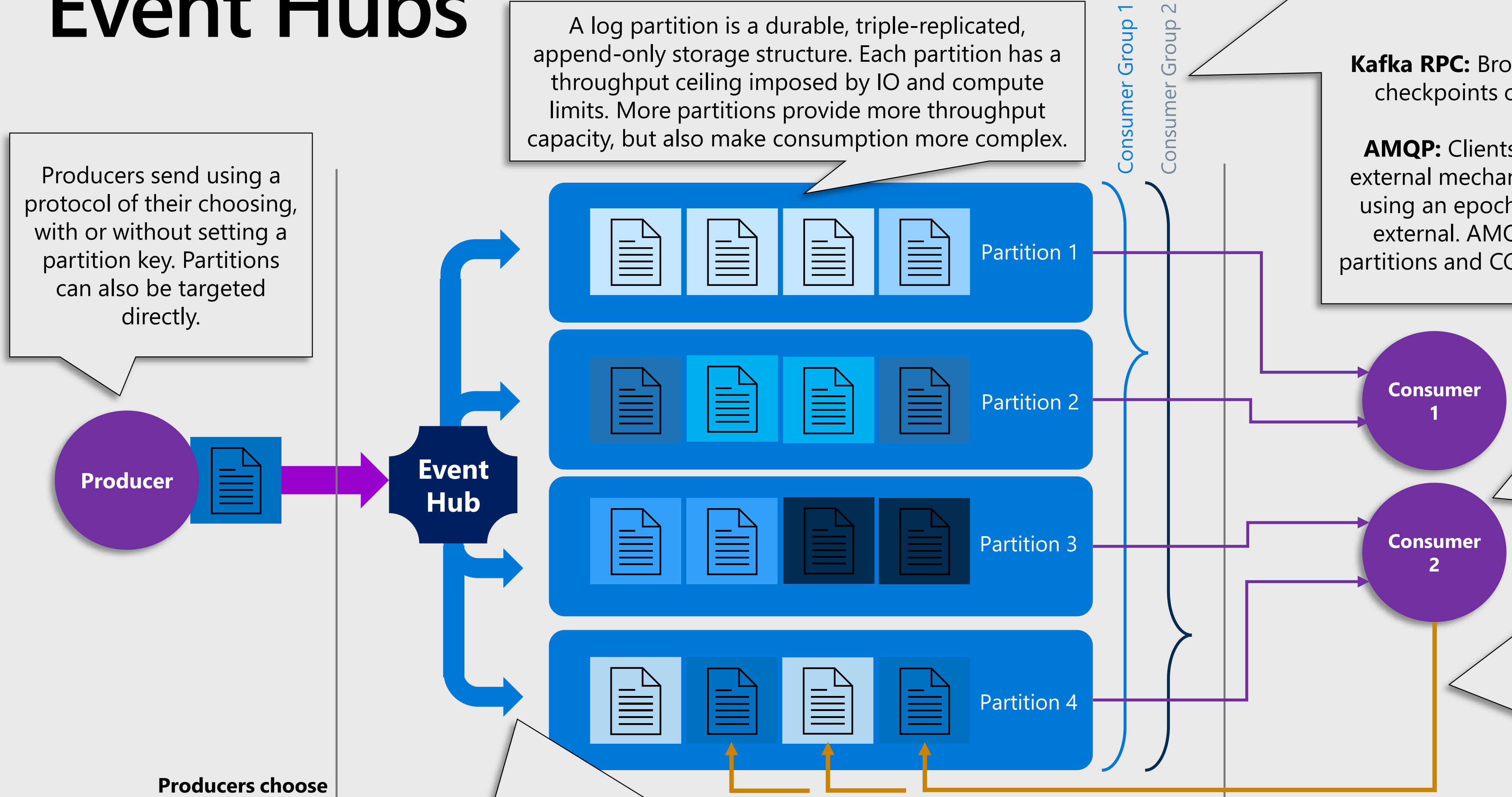
Producers send using a protocol of their choosing, with or without setting a partition key. Partitions can also be targeted directly.

A log partition is a durable, triple-replicated, append-only storage structure. Each partition has a throughput ceiling imposed by IO and compute limits. More partitions provide more throughput capacity, but also make consumption more complex.

A consumer group is an optional mechanism to allow multiple parallel consumers to coordinate who owns which partition.

Kafka RPC: Broker assigns partitions and checkpoints can be stored in the CG.

AMQP: Clients choose leader using an external mechanism and can break leases using an epoch model. Checkpoints are external. AMQP with server-assigned partitions and CG checkpoints on roadmap.



Consumers on a consumer group own one or more partitions exclusively to ensure in-order processing.

Consumers choose the initial log offset:

Start (oldest), End (newest), Timestamp, Sequence Number

Read progress is noted in checkpoints that can be stored anywhere.

An Event Hub is an event stream manager handling one or more independent log partitions. Different from Apache Kafka®, Event Hubs can optionally assign events to partitions on the server-side.

Producers choose their preferred protocol

OASIS AMQP 1.0
Apache Kafka® RPC
HTTPS 1.1

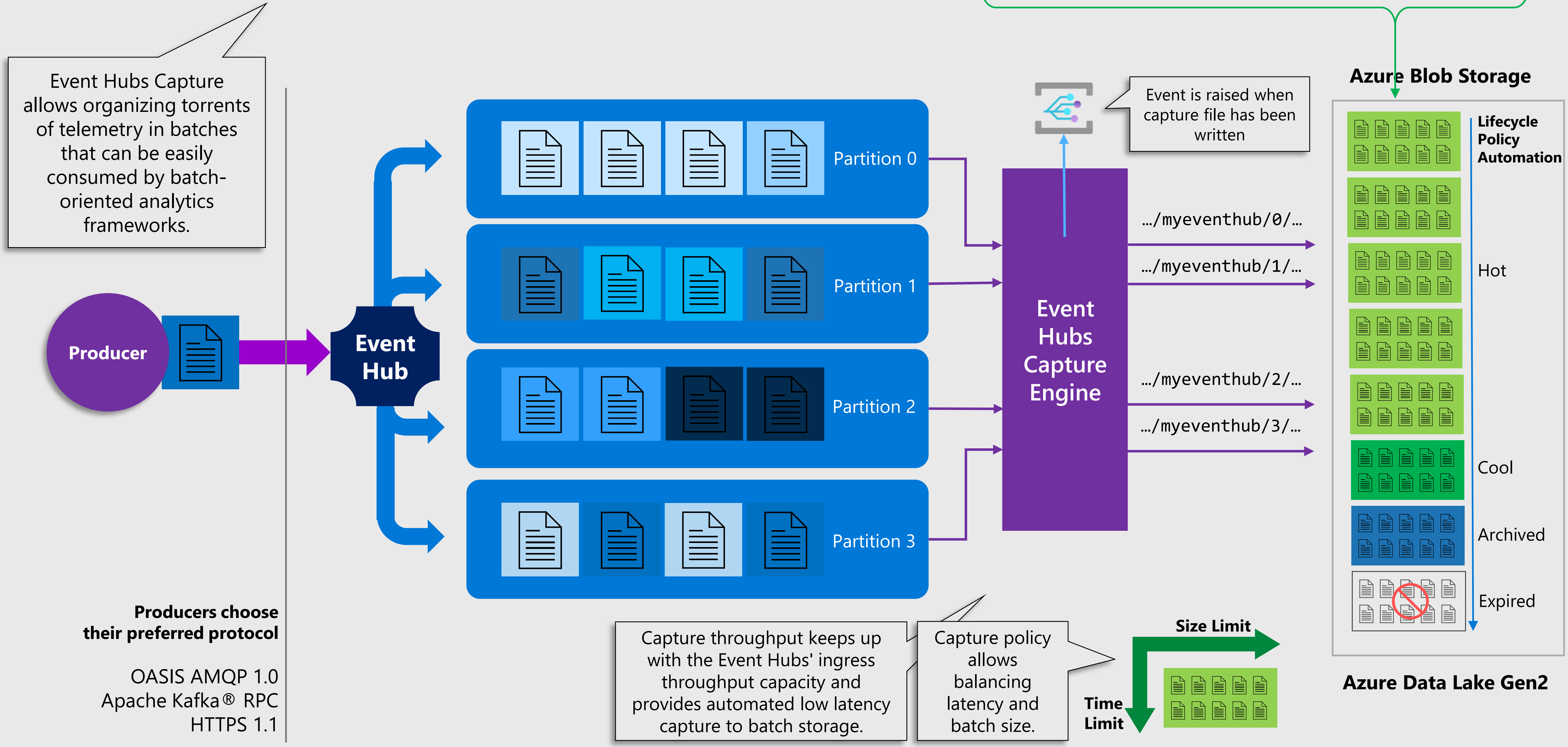
Consumers choose their own preferred protocol

OASIS AMQP 1.0
Apache Kafka® RPC

Event Hubs Capture

`{Namespace}/{EventHub}/{PartitionId}/{Year}/{Month}/{Day}/{Hour}/{Minute}/{Second}`

`https://mystorageaccount.blob.core.windows.net/mycontainer/mynamespace/myeventhub/0/2017/12/08/03/03/17.avro`



Event Hubs Capture allows organizing torrents of telemetry in batches that can be easily consumed by batch-oriented analytics frameworks.

Event is raised when capture file has been written

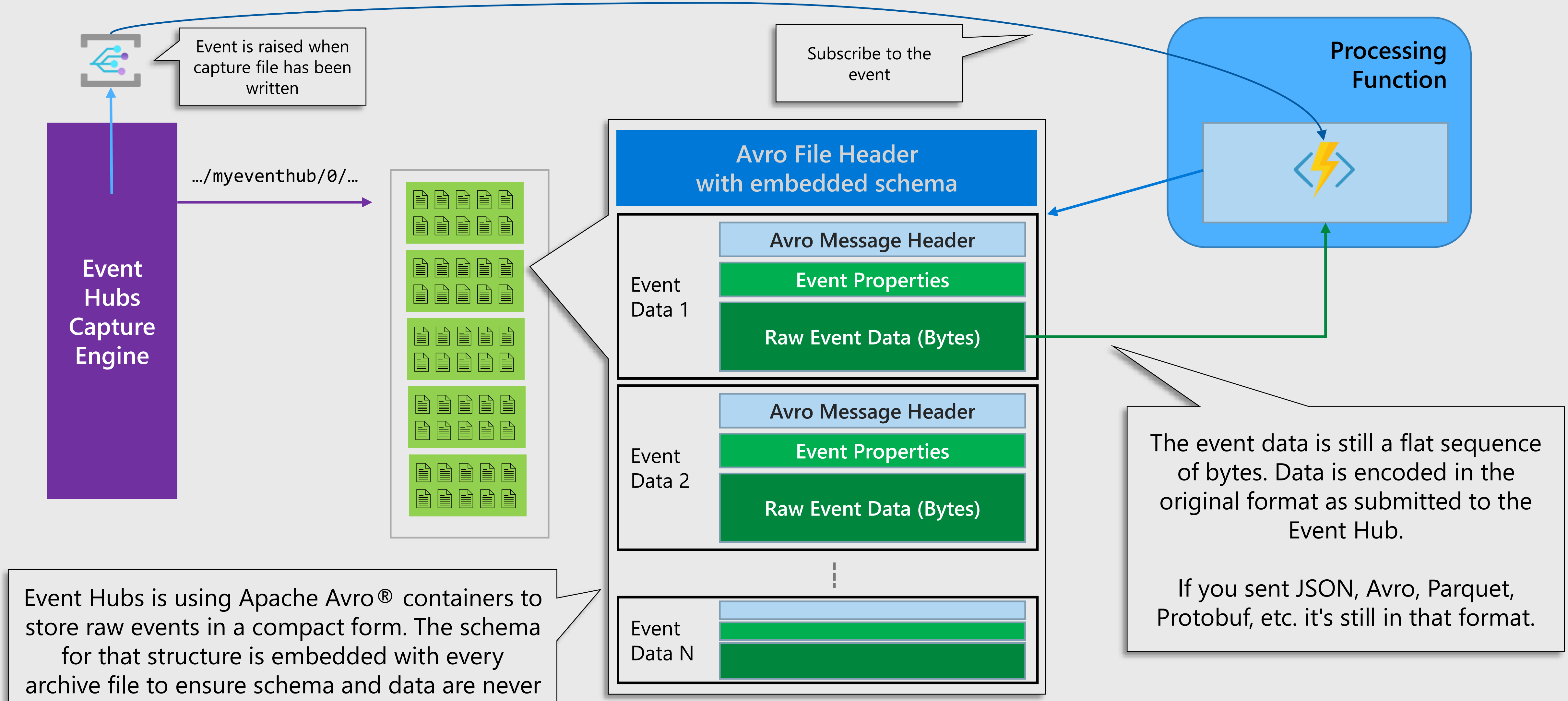
Capture throughput keeps up with the Event Hubs' ingress throughput capacity and provides automated low latency capture to batch storage.

Capture policy allows balancing latency and batch size.

Producers choose their preferred protocol

OASIS AMQP 1.0
Apache Kafka® RPC
HTTPS 1.1

Event Hubs Capture Avro Archives



The event data is still a flat sequence of bytes. Data is encoded in the original format as submitted to the Event Hub.

If you sent JSON, Avro, Parquet, Protobuf, etc. it's still in that format.

Event Hubs is using Apache Avro® containers to store raw events in a compact form. The schema for that structure is embedded with every archive file to ensure schema and data are never separated or ambiguous.

* Apache Kafka isn't any of these things, either.

Event Hubs* is not...

... a publish/subscribe broker. Partitions are not subscriptions. They are chosen by the producer or the broker on ingress. There is also no server-side filtering.

→ Azure Service Bus, Azure Event Grid

... a queue broker. Read progress over the log is handled by the client and there is no event-level ownership and delivery state handling.

→ Azure Service Bus

... a discrete event distribution engine. Event Hubs does not do push deliveries, and delivery failures need to be tracked individually.

→ Azure Event Grid

... a database or long-term event store. Event Hubs exists to catch, store and provide fast access to event data organized around time axis. As data ages (days, not months), you need better indexing.

→ Azure Cosmos DB, Azure SQL, Azure Synapse, Azure Data Explorer

Apache Kafka® Clients

https://cwiki.apache.org/confluence/pages/viewpage.action?pageId=30736606

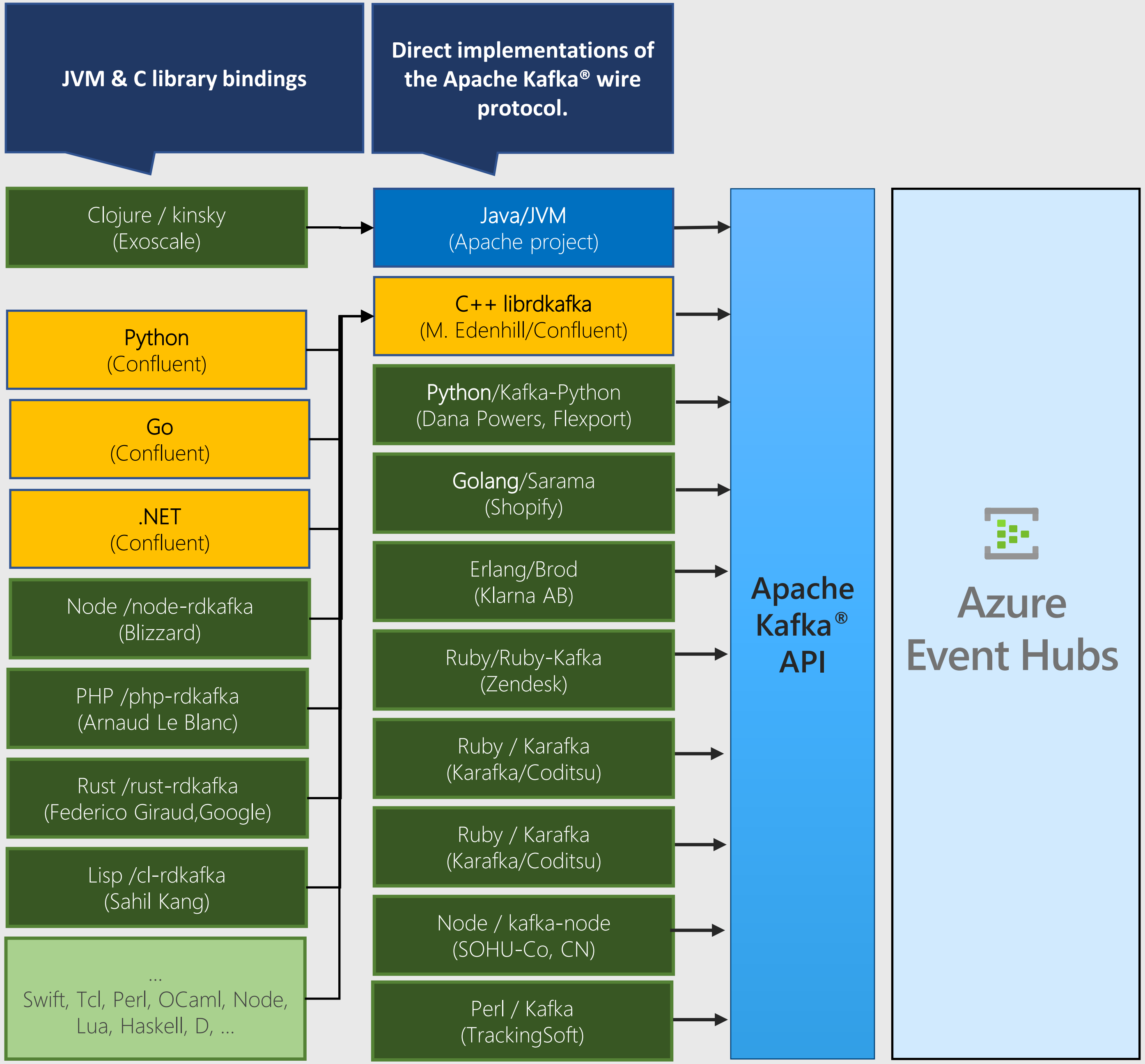
Azure Event Hubs is compatible with the producer and consumer APIs of all current Apache Kafka clients.

Only the Java/JVM client is part of the Apache Kafka project. All other clients are under various non-foundation ownerships and licenses. Azure does not provide QFE support for any of these clients but does provide support for service-side compatibility.

Apache Software Foundation

Non-Foundation, Community / Best Effort Support

Non-Foundation, Commercial Support Option



JVM & C library bindings

Direct implementations of the Apache Kafka® wire protocol.

Clojure / kinsky (Exoscale)

Python (Confluent)

Go (Confluent)

.NET (Confluent)

Node /node-rdkafka (Blizzard)

PHP /php-rdkafka (Arnaud Le Blanc)

Rust /rust-rdkafka (Federico Giraud,Google)

Lisp /cl-rdkafka (Sahil Kang)

… Swift, Tcl, Perl, OCaml, Node, Lua, Haskell, D, …

Java/JVM (Apache project)

C++ librdkafka (M. Edenhill/Confluent)

Python/Kafka-Python (Dana Powers, Flexport)

Golang/Sarama (Shopify)

Erlang/Brod (Klarna AB)

Ruby/Ruby-Kafka (Zendesk)

Ruby / Karafka (Karafka/Coditsu)

Ruby / Karafka (Karafka/Coditsu)

Node / kafka-node (SOHU-Co, CN)

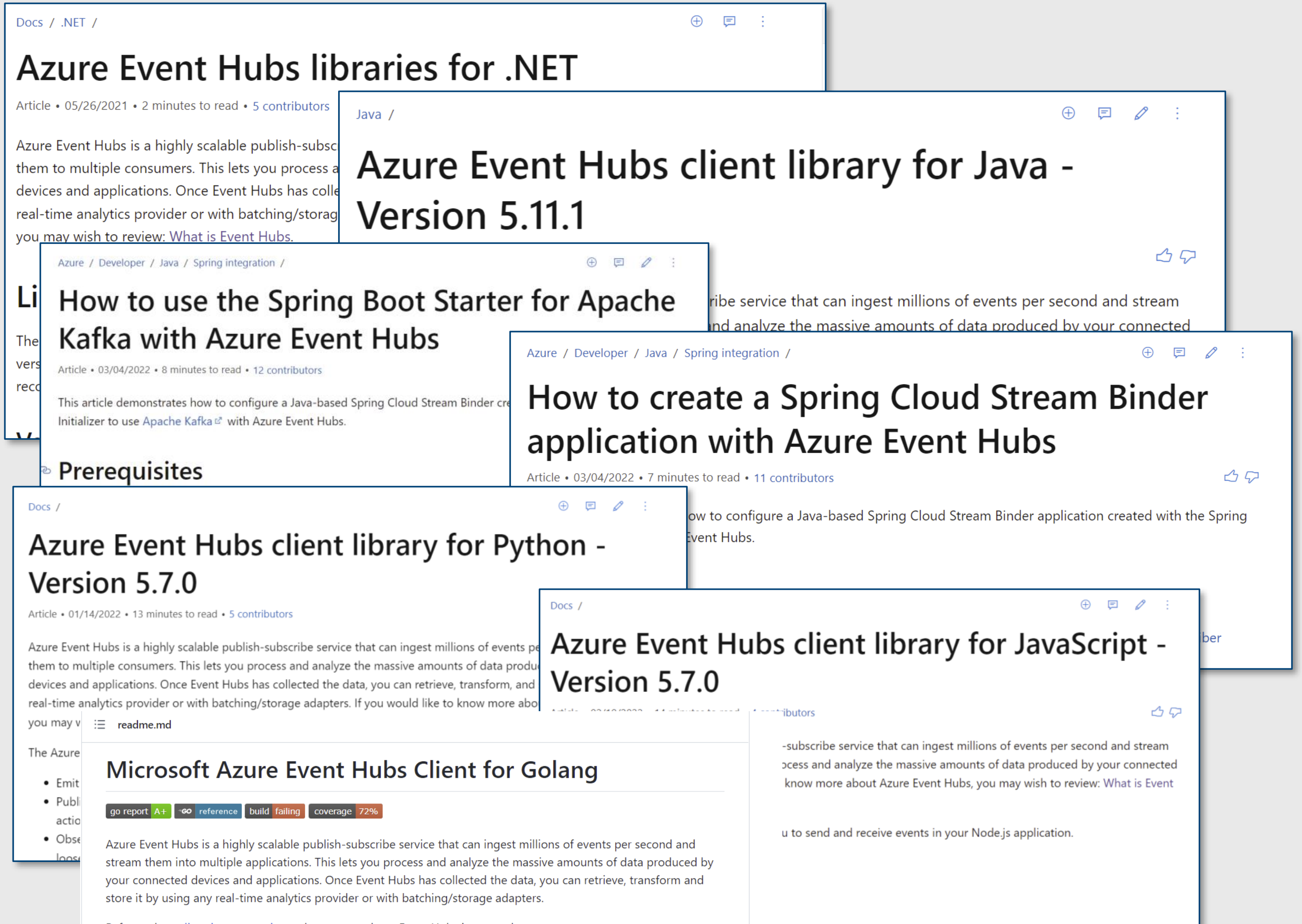
Perl / Kafka (TrackingSoft)

Apache Kafka® API

Azure Event Hubs

Event Hubs SDK Support: Azure SDKs

- .NET
- Java
- Java Spring Boot
- Python
- JavaScript/Typescript
- Go



Docs / .NET /

Azure Event Hubs libraries for .NET

Article • 05/26/2021 • 2 minutes to read • 5 contributors

Azure Event Hubs is a highly scalable publish-subsc
them to multiple consumers. This lets you process a
devices and applications. Once Event Hubs has colle
real-time analytics provider or with batching/storag
you may wish to review: What is Event Hubs.

Li

The
vers
reco

Java /

Azure Event Hubs client library for Java - Version 5.11.1

ribe service that can ingest millions of events per second and stream
nd analyze the massive amounts of data produced by your connected

Azure / Developer / Java / Spring integration /

How to use the Spring Boot Starter for Apache Kafka with Azure Event Hubs

Article • 03/04/2022 • 8 minutes to read • 12 contributors

This article demonstrates how to configure a Java-based Spring Cloud Stream Binder cre
Initializer to use Apache Kafka with Azure Event Hubs.

Prerequisites

Azure / Developer / Java / Spring integration /

How to create a Spring Cloud Stream Binder application with Azure Event Hubs

Article • 03/04/2022 • 7 minutes to read • 11 contributors

ow to configure a Java-based Spring Cloud Stream Binder application created with the Spring
vent Hubs.

ber

Docs /

Azure Event Hubs client library for Python - Version 5.7.0

Article • 01/14/2022 • 13 minutes to read • 5 contributors

Azure Event Hubs is a highly scalable publish-subscribe service that can ingest millions of events pe
them to multiple consumers. This lets you process and analyze the massive amounts of data produ
devices and applications. Once Event Hubs has collected the data, you can retrieve, transform, and
real-time analytics provider or with batching/storage adapters. If you would like to know more abo
you may v

The Azure

- Emit
- Publ
  acti
- Obse
  loos

Docs /

Azure Event Hubs client library for JavaScript - Version 5.7.0

ibutors

-subscribe service that can ingest millions of events per second and stream
ocess and analyze the massive amounts of data produced by your connected
know more about Azure Event Hubs, you may wish to review: What is Event

u to send and receive events in your Node.js application.

readme.md

Microsoft Azure Event Hubs Client for Golang

Azure Event Hubs is a highly scalable publish-subscribe service that can ingest millions of events per second and stream them into multiple applications. This lets you process and analyze the massive amounts of data produced by your connected devices and applications. Once Event Hubs has collected the data, you can retrieve, transform and store it by using any real-time analytics provider or with batching/storage adapters.

Compatible Generic AMQP 1.0 Clients

Client Stack	URL
Apache Qpid Proton-C (Go, Python, Ruby, C++)	https://qpid.apache.org/proton/
Apache Qpid Messaging API	https://qpid.apache.org/components/messaging-api
Apache Qpid Proton-J	https://qpid.apache.org/proton/
Apache NMS AMQP	https://activemq.apache.org/components/nms/providers/amqp/
AMQP .NET Lite (all variants of .NET, incl. Nano & Micro)	https://github.com/Azure/amqpnetlite
Azure AMQP (our own server stack)	https://github.com/Azure/azure-amqp
Azure uAMQP C (Python, PHP)	https://github.com/Azure/azure-uamqp-c https://github.com/Azure/azure-uamqp-python
Rhea (NodeJS)	https://github.com/amqp/rhea
Go AMQP	https://github.com/Azure/go-amqp
Vert.X AMQP Client	https://vertx.io/docs/vertx-amqp-client/java/

Azure Event Hubs Deep Dive

Architecture

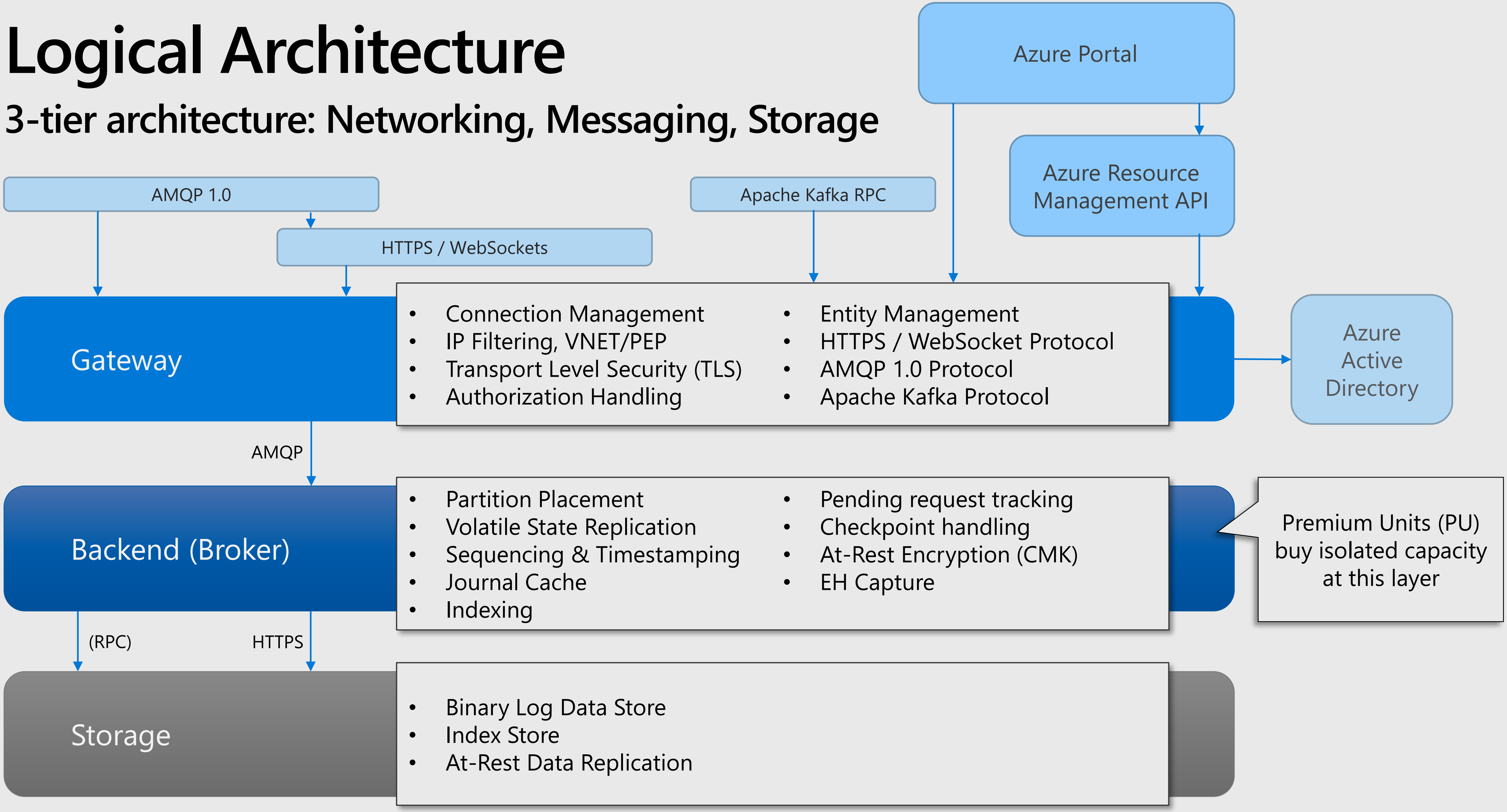
Internal Architecture

Protocols

Performance Metrics

Logical Architecture

3-tier architecture: Networking, Messaging, Storage



Azure Portal

Azure Resource Management API

AMQP 1.0

HTTPS / WebSockets

Apache Kafka RPC

Gateway

- Connection Management
- IP Filtering, VNET/PEP
- Transport Level Security (TLS)
- Authorization Handling
- Entity Management
- HTTPS / WebSocket Protocol
- AMQP 1.0 Protocol
- Apache Kafka Protocol

Azure Active Directory

AMQP

Backend (Broker)

- Partition Placement
- Volatile State Replication
- Sequencing & Timestamping
- Journal Cache
- Indexing
- Pending request tracking
- Checkpoint handling
- At-Rest Encryption (CMK)
- EH Capture

Premium Units (PU) buy isolated capacity at this layer

(RPC)

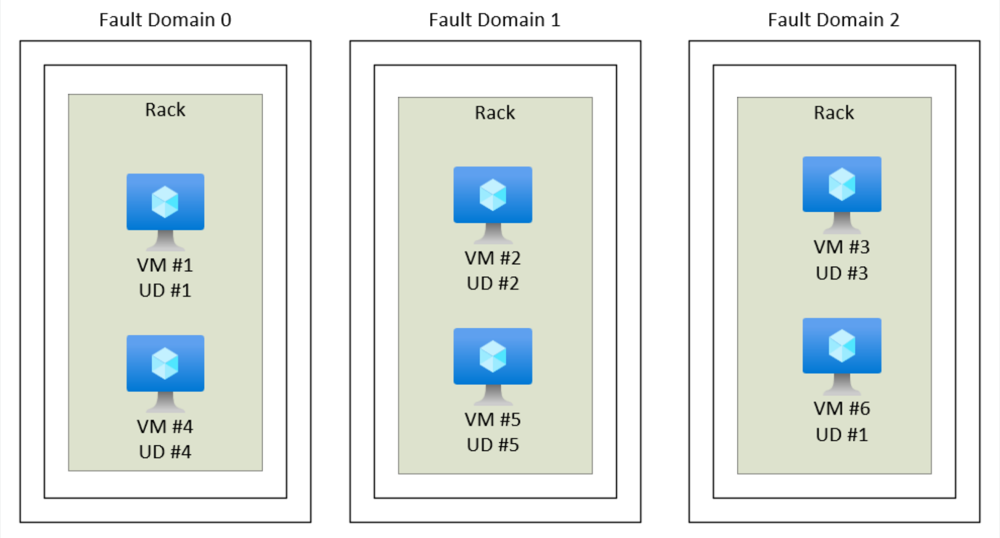
HTTPS

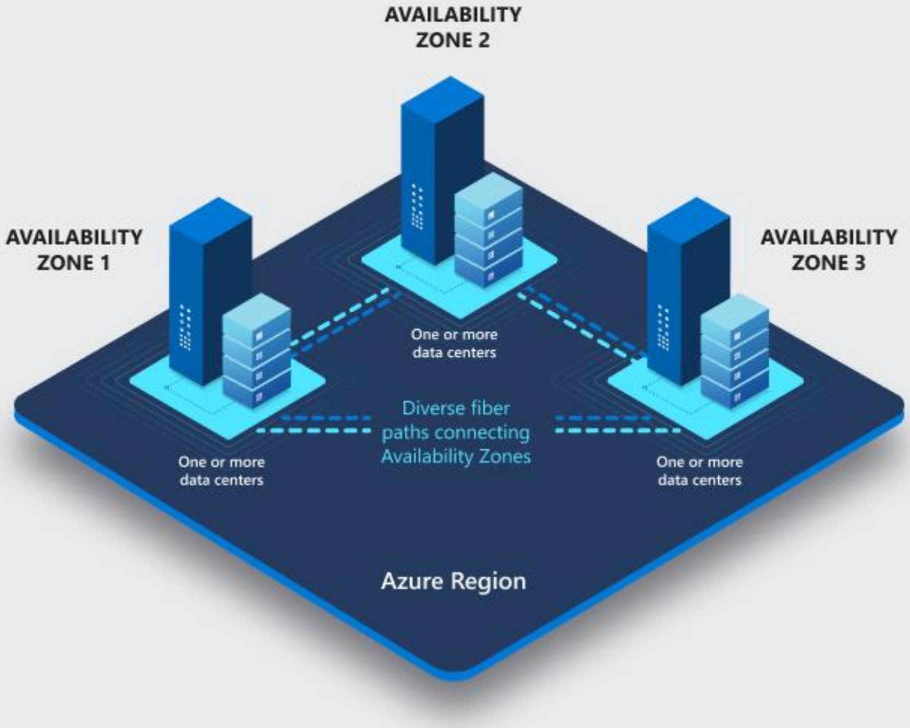
Storage

- Binary Log Data Store
- Index Store
- At-Rest Data Replication

Fault Domains & Availability Zones

High-Availability starts at the physical reality



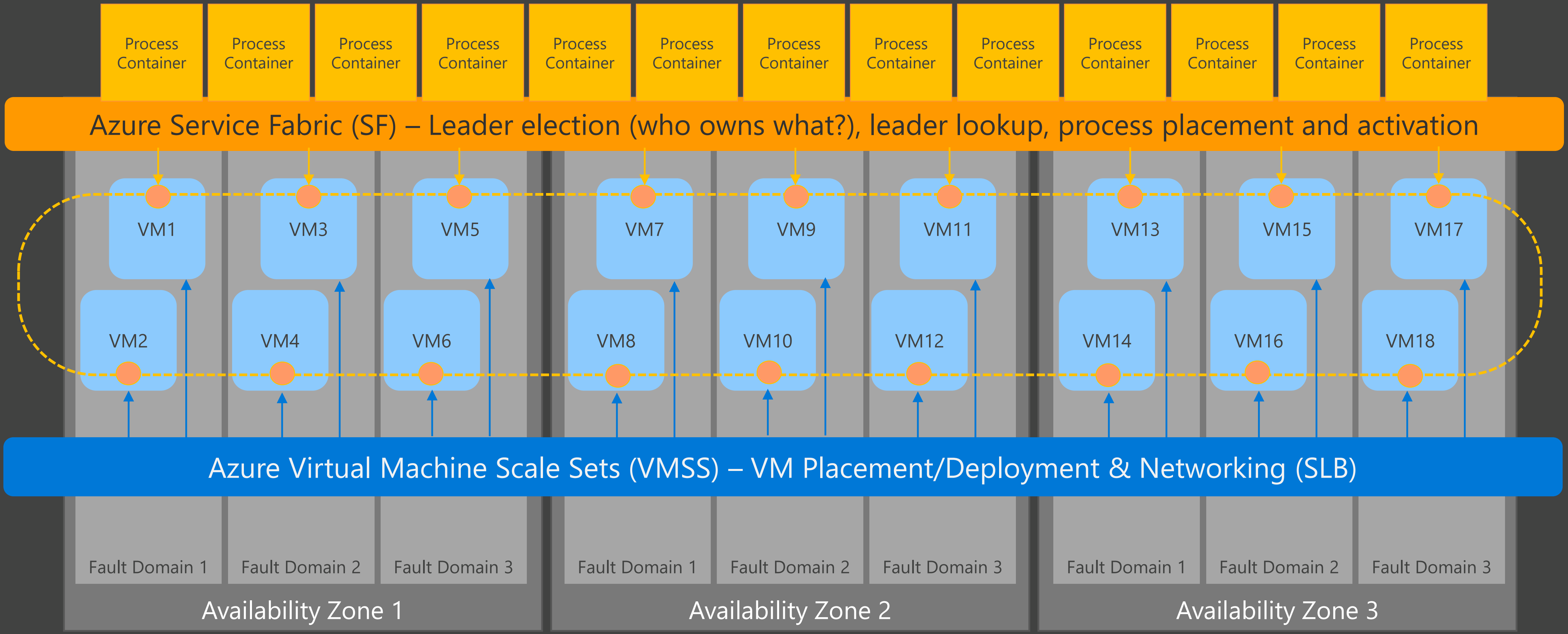


Fault Domain Placement:
Cluster VMs are spread across at least 3 fault domains such that the loss of a rack or network poses no availability risk. Recovery from a fault domain failure is fully automated and the system maintains SLA.

Availability Zones Placement:
Each cluster spans three availability zones and maintains SLA without any tolerance for data loss when one or two zones fail.

Backend and Gateway Clusters

Logical Architecture meets Placement



Backend and Gateway Clusters

Principles

Everything is automated. There's no manual intervention in placement of VMs or processes or workloads.

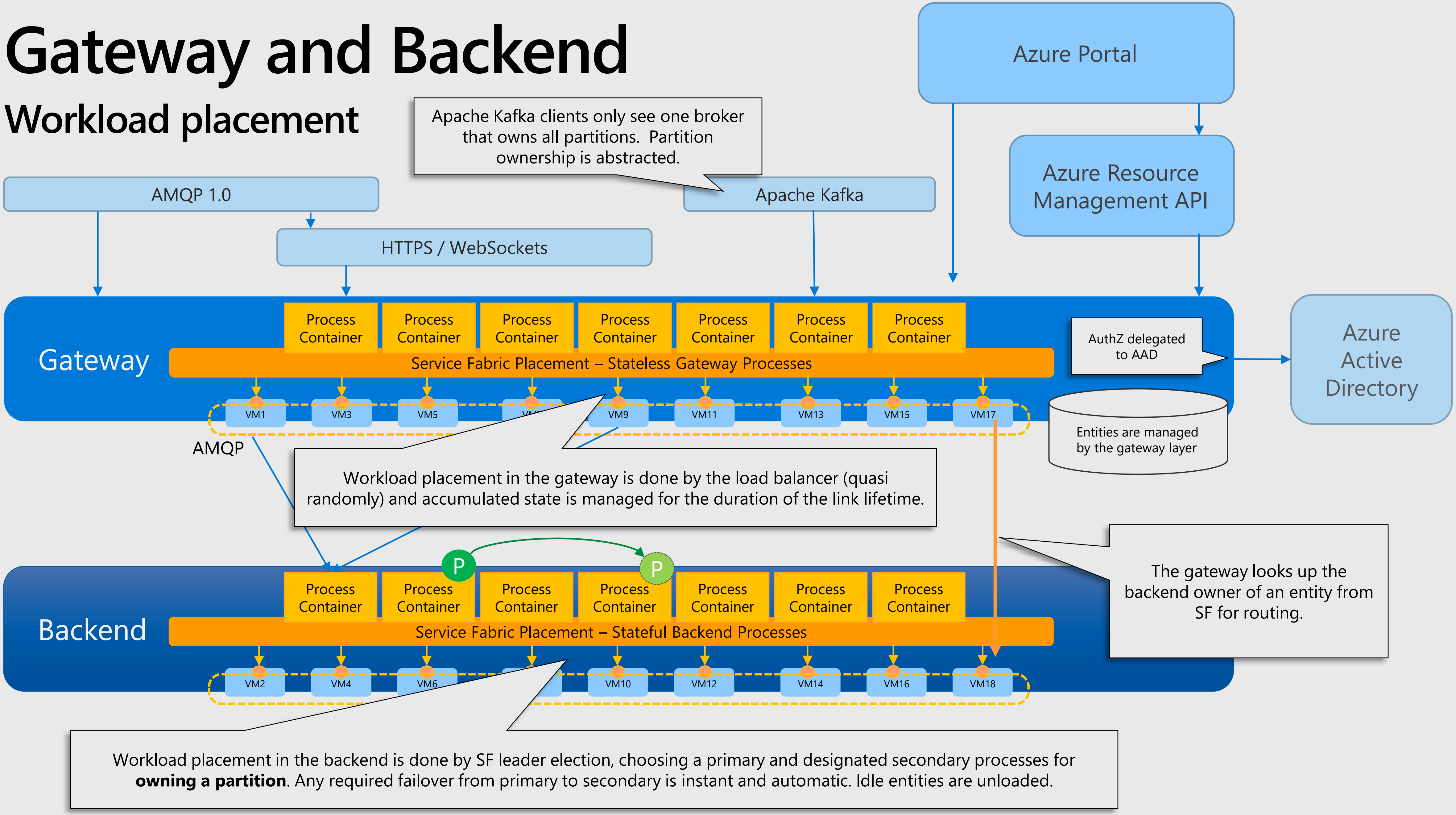
Each cluster has a well-known number of virtual machines. We don't use auto-scaling with VMs. We may (rarely) choose to initiate scale up/down to a different level.

Process placement and servicing/upgrade cycles are controlled by Azure Service Fabric.

Allocation of namespaces to clusters are based on well-established heuristics (Standard) or deterministic capacity limits (Premium). Customer-defined allocation in Dedicated.

Gateway and Backend

Workload placement



Apache Kafka clients only see one broker that owns all partitions. Partition ownership is abstracted.

Workload placement in the gateway is done by the load balancer (quasi randomly) and accumulated state is managed for the duration of the link lifetime.

The gateway looks up the backend owner of an entity from SF for routing.

Workload placement in the backend is done by SF leader election, choosing a primary and designated secondary processes for **owning a partition**. Any required failover from primary to secondary is instant and automatic. Idle entities are unloaded.

Backend: Event Hubs Premium.

Workload placement

Namespace PU are split across processes:
1 PU = 2 Proc (8GB Mem), 1 Core/Proc (2C)
2 PU = 2 Proc, 1 Core/Proc + 1 Core (3C)
4 PU = 2 Proc, 2 Core/Proc + 1 Core (5C)
8 PU = 4 Proc, 2 Core/Proc + 1 Core (9C)
16 PU = 4 Proc, 4 Core/Proc + 1 Core (17C)

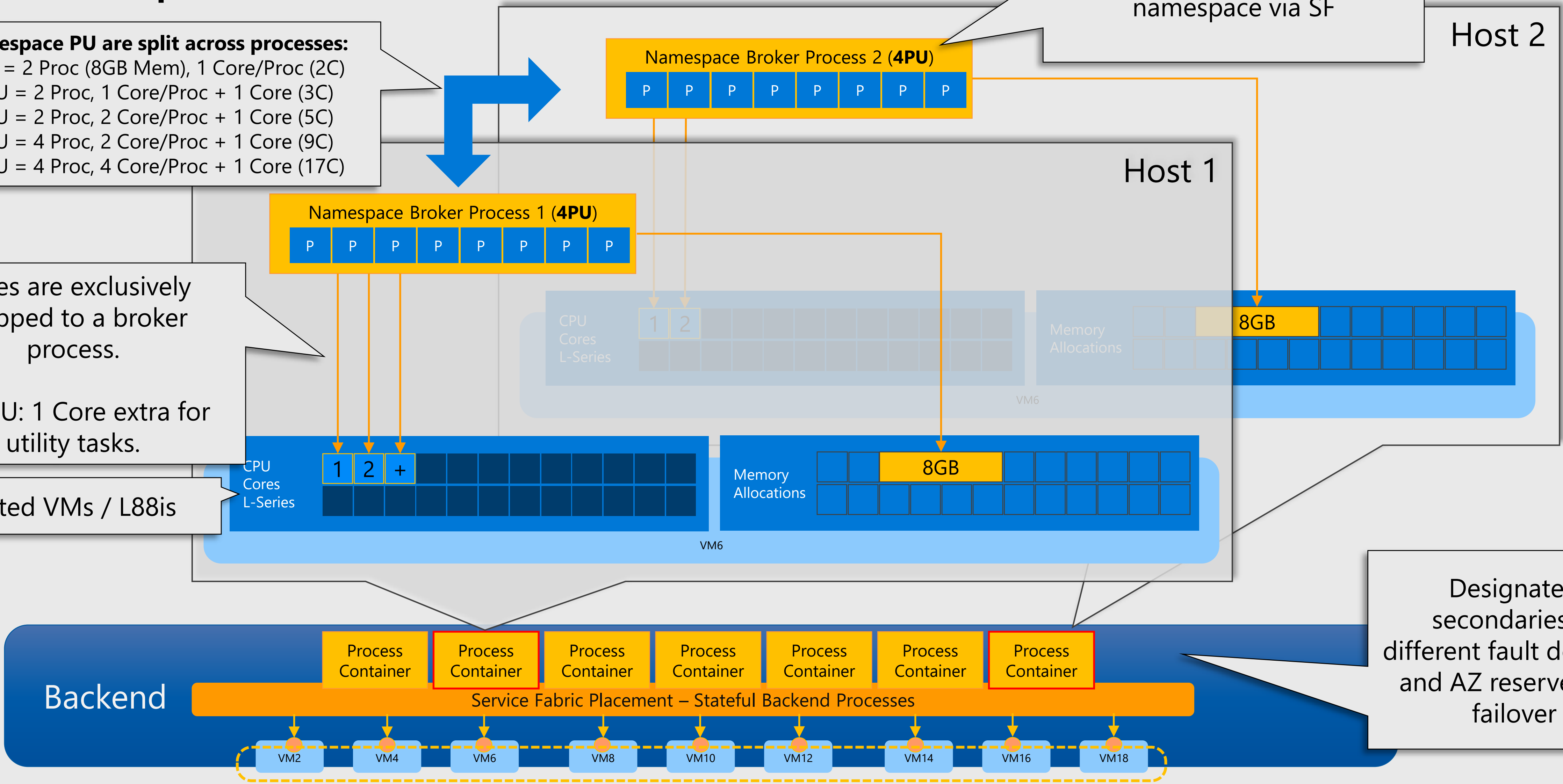
Partition ownership is dynamically mapped to the process(es) associated with a namespace via SF

Cores are exclusively mapped to a broker process.

>=2PU: 1 Core extra for utility tasks.

Isolated VMs / L88is

Designated secondaries in different fault domains and AZ reserved for failover

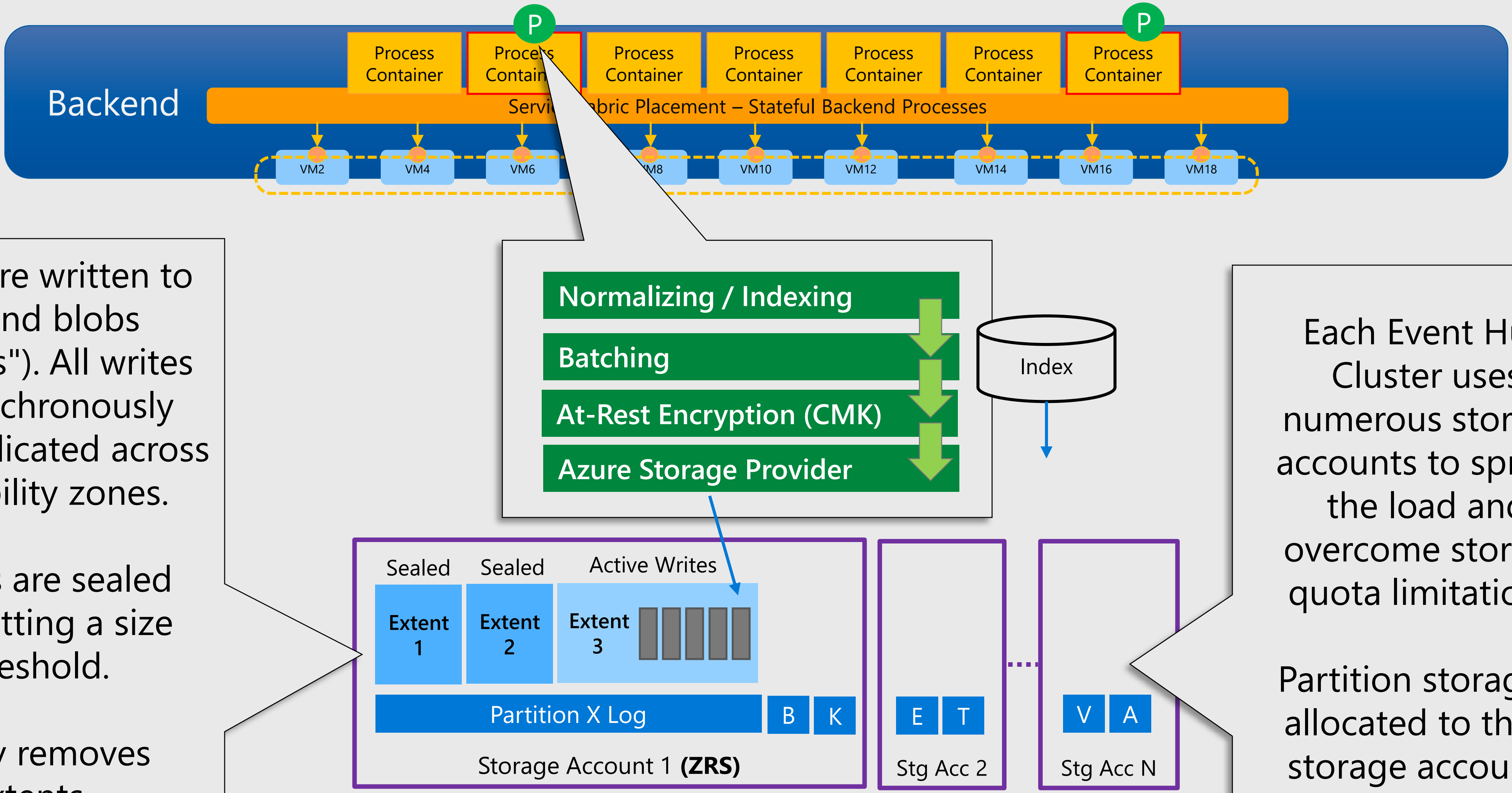


Storage Layer – Event Hubs Standard

Events are written to append blobs ("extents"). All writes are synchronously triple replicated across availability zones.

Extents are sealed after hitting a size threshold.

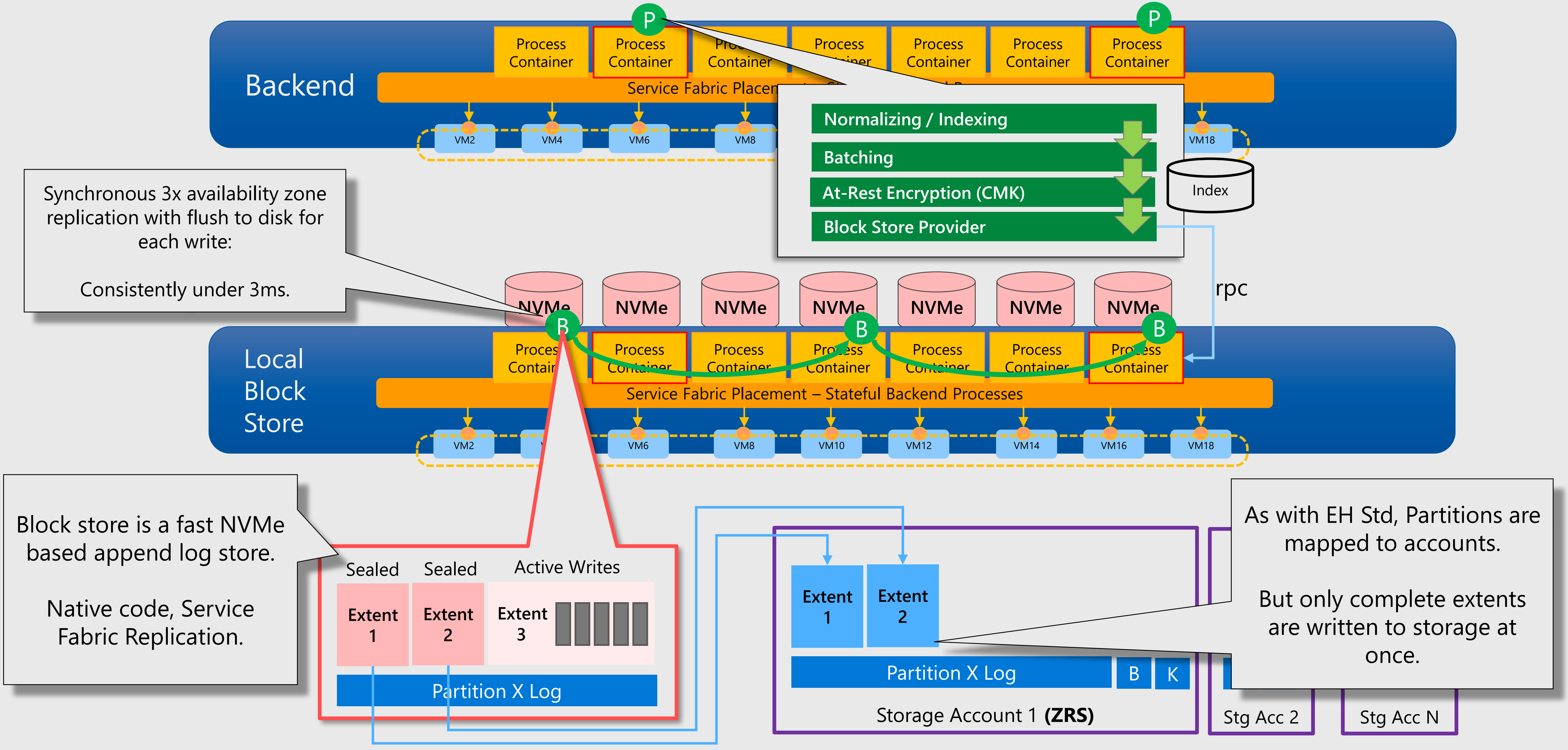
Expiry removes extents.



Each Event Hub Cluster uses numerous storage accounts to spread the load and overcome storage quota limitations.

Partition storage is allocated to those storage accounts.

Storage Layer – Event Hubs Premium



Synchronous 3x availability zone replication with flush to disk for each write:

Consistently under 3ms.

Block store is a fast NVMe based append log store.

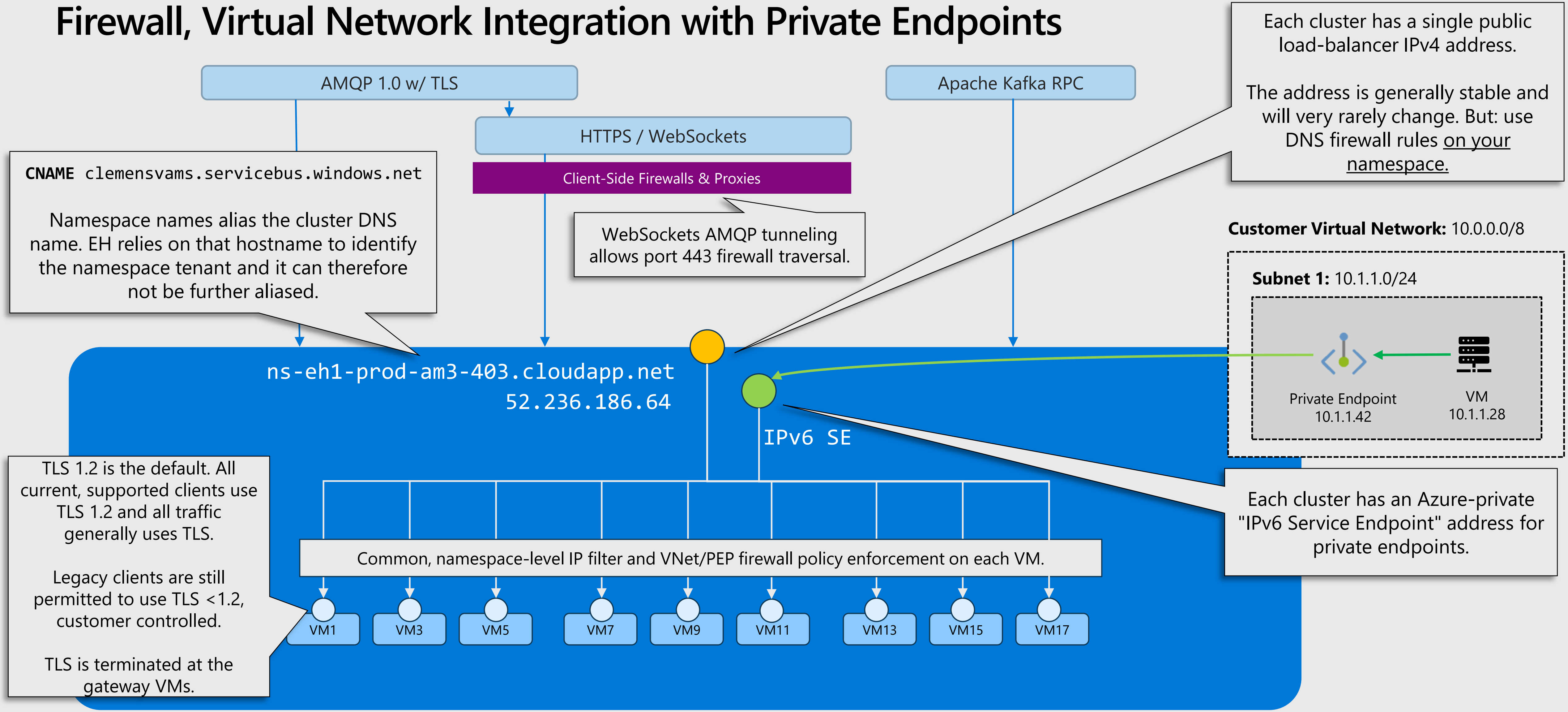
Native code, Service Fabric Replication.

As with EH Std, Partitions are mapped to accounts.

But only complete extents are written to storage at once.

Networking Features

Firewall, Virtual Network Integration with Private Endpoints



CNAME clemensvams.servicebus.windows.net

Namespace names alias the cluster DNS name. EH relies on that hostname to identify the namespace tenant and it can therefore not be further aliased.

WebSockets AMQP tunneling allows port 443 firewall traversal.

Each cluster has a single public load-balancer IPv4 address.

The address is generally stable and will very rarely change. But: use DNS firewall rules on your namespace.

Customer Virtual Network: 10.0.0.0/8

Subnet 1: 10.1.1.0/24

Each cluster has an Azure-private "IPv6 Service Endpoint" address for private endpoints.

Common, namespace-level IP filter and VNet/PEP firewall policy enforcement on each VM.

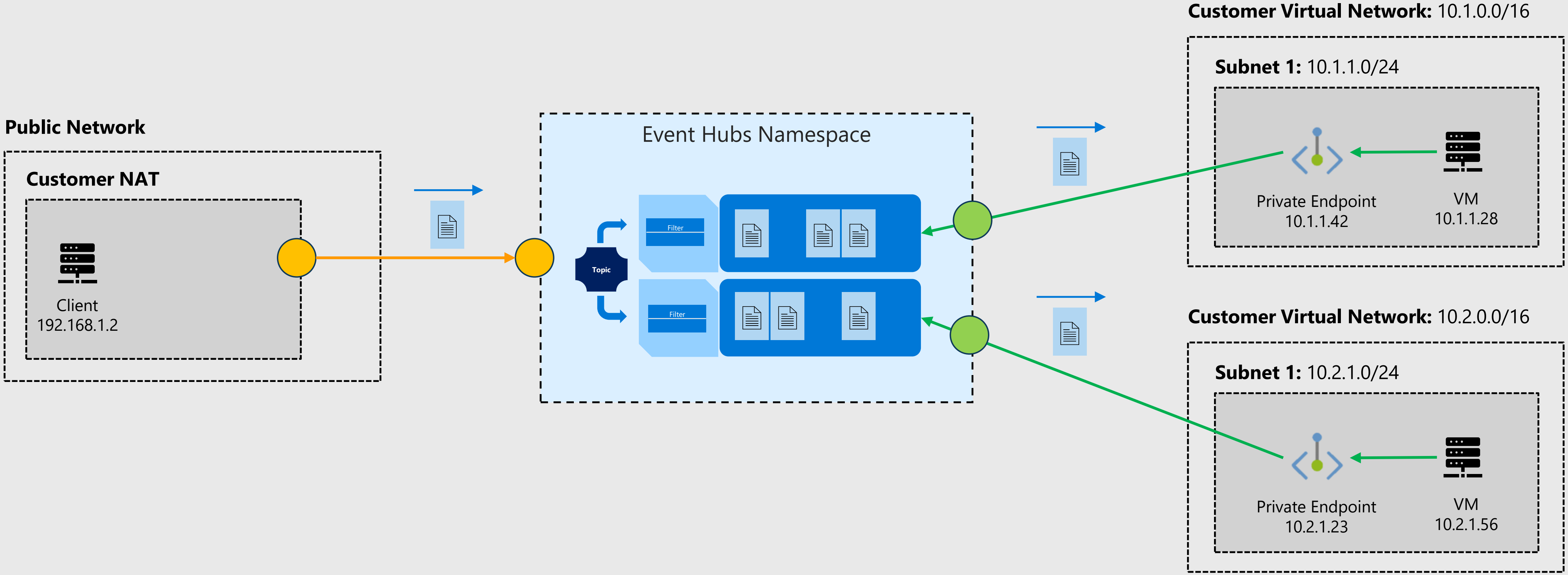
TLS 1.2 is the default. All current, supported clients use TLS 1.2 and all traffic generally uses TLS.

Legacy clients are still permitted to use TLS <1.2, customer controlled.

TLS is terminated at the gateway VMs.

Networking Features

Event Hubs namespaces can be attached to one or more virtual networks and the public IP address space concurrently and act as safe "Layer 7" stream bridges.



Authorization

Azure Active Directory RBAC and Shared Access Signatures (SAS)

Event Hubs has two authorization models.

SAS (Local): Simple model for external clients that cannot use Azure Active Directory for any reason.

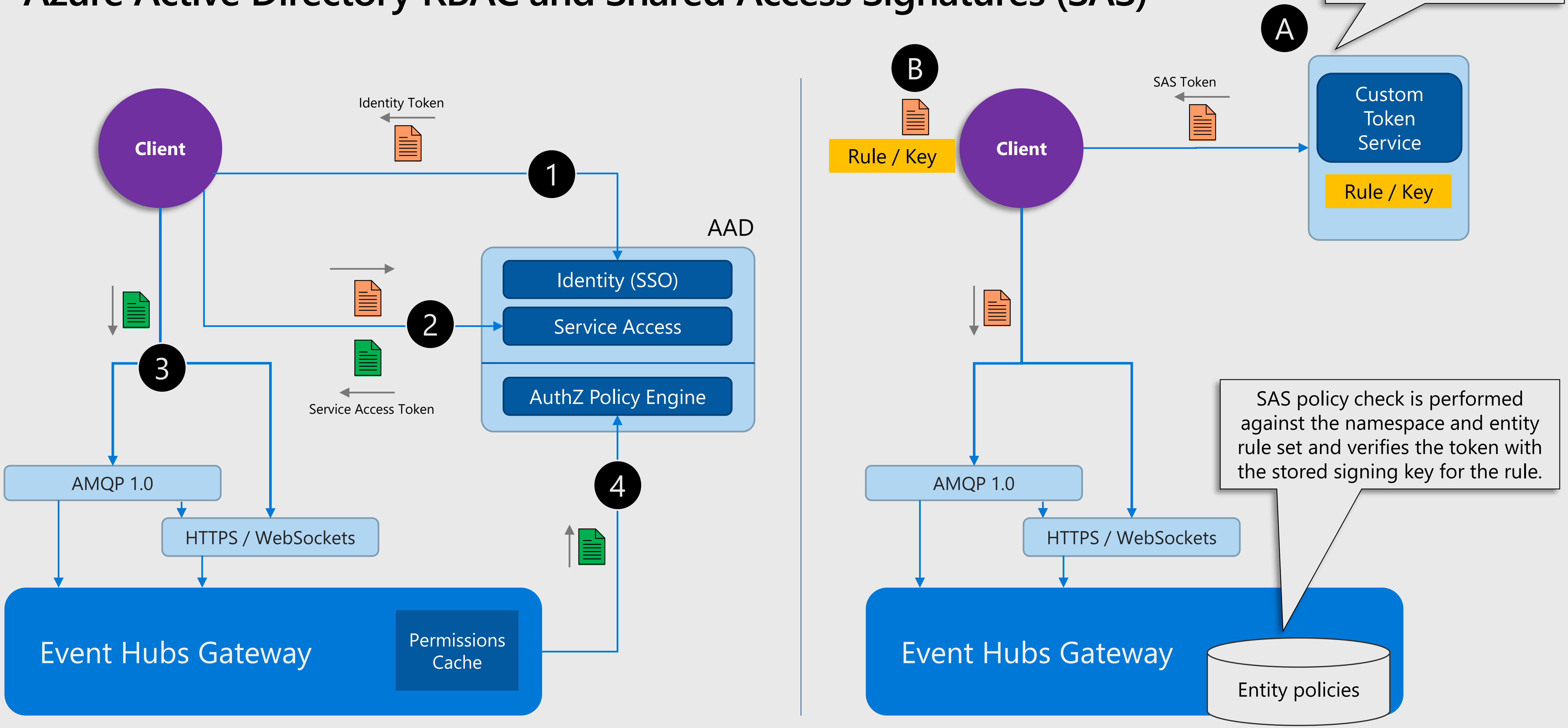
- Very simple. Rights at namespace or entity scope: "Send", "Listen" (Receive/Subscribe), "Manage"
- No accounts. 12 named rules per scope, each rule has a key. Rules combine a set of rights.
- HMACSHA256 signed **tokens** are issued using name and key and passed to the service.
- Tokens can be issued in the client or by some security token service holding the key
- *Can be turned off via the portal on the overview page under "Local Authentication"*

AAD RBAC: Integrated Azure identity model, also for service-level integration (managed identity)

- Very detailed, operation-level permission set that can be assigned to roles
- Several standard control- and data-plane roles (like Data Owner, Data Sender, Data Receiver)
- Roles can be assigned at the namespace and entity level

Authorization

Azure Active Directory RBAC and Shared Access Signatures (SAS)



Allows very fine grained control with tokens scoped to a certain time and/or EH and/or publisher policy

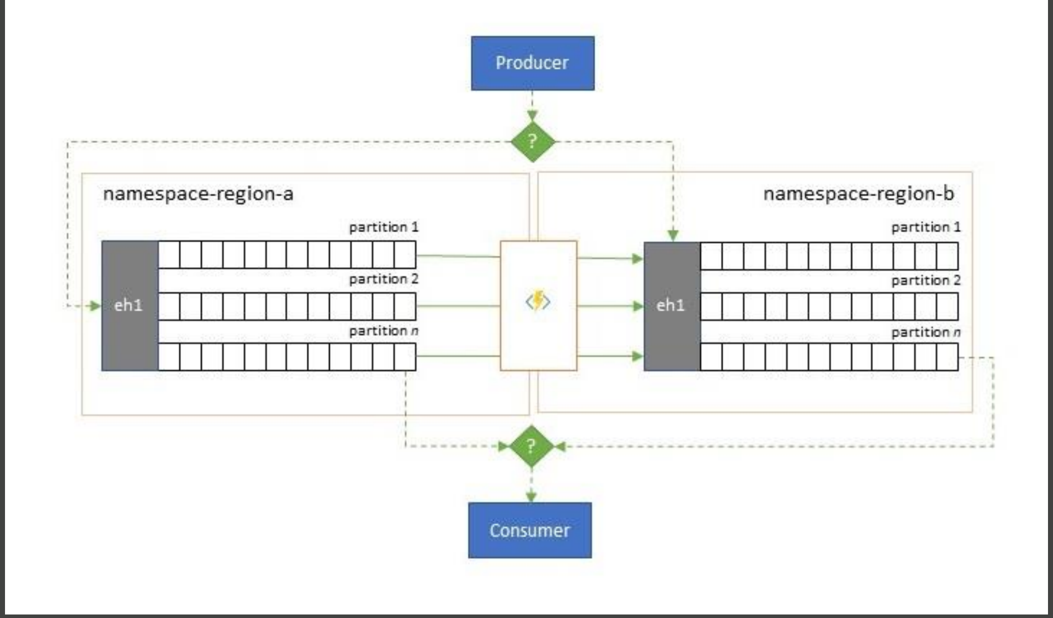
SAS policy check is performed against the namespace and entity rule set and verifies the token with the stored signing key for the rule.

Event Hubs Premium

Replication & Federation

Replication Tasks

Providing code-free replication of data between Service Bus and Event Hubs



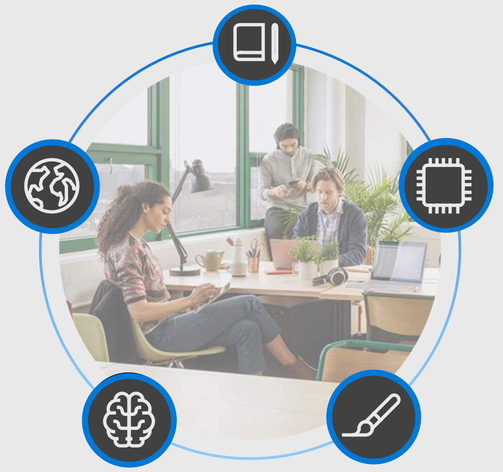
Problem

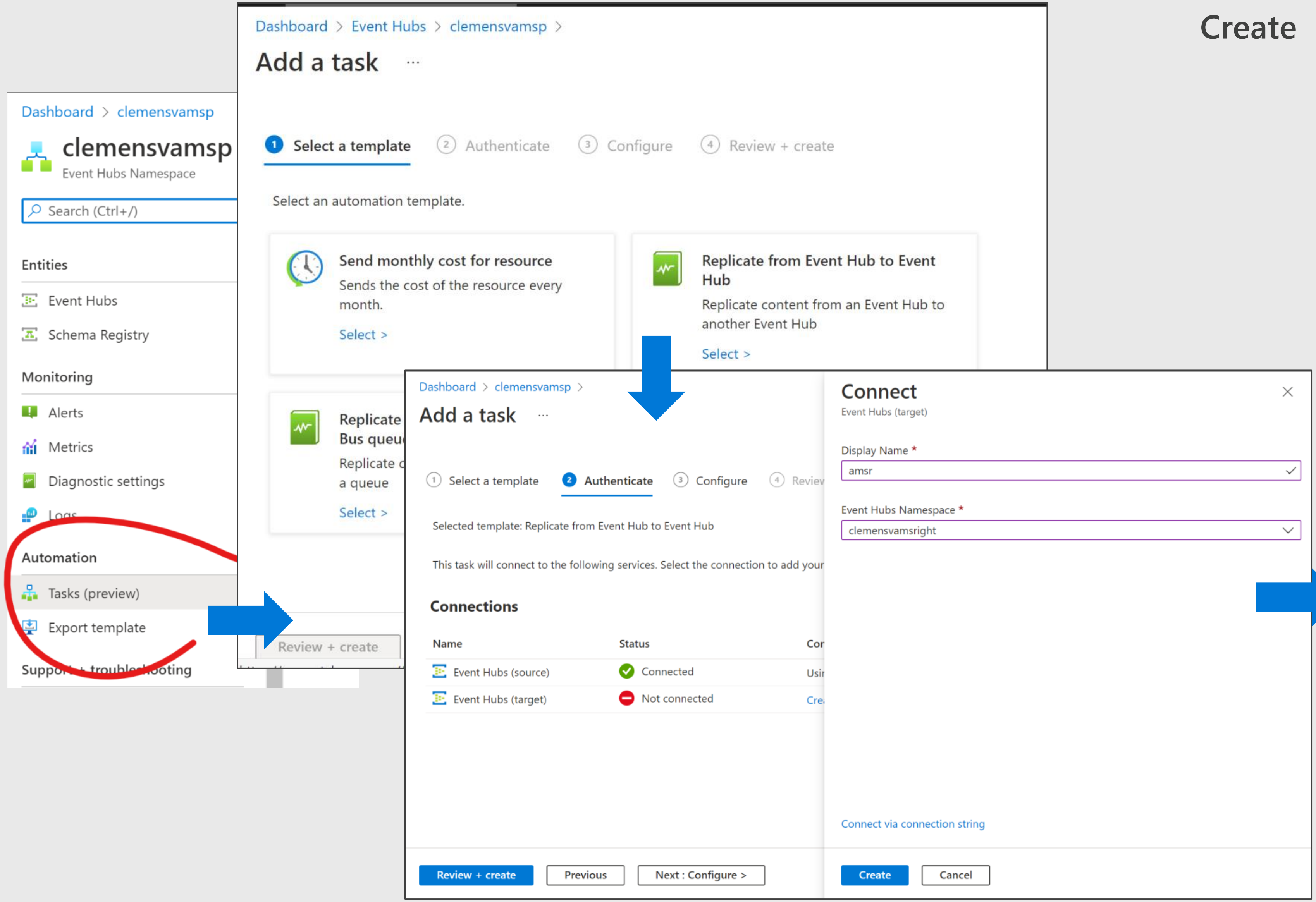
In many scenarios, event data needs to be moved or merged across regions.

Solution

Quick and simple replication between namespaces through templates available on Azure Portal’s Tasks.

Automation Tasks to the rescue





Dashboard > clemensvamsp

clemensvamsp
Event Hubs Namespace

Search (Ctrl+/)

Entities
- Event Hubs
- Schema Registry

Monitoring
- Alerts
- Metrics
- Diagnostic settings
- Logs

Automation
- Tasks (preview)
- Export template

Support + troubleshooting

Dashboard > Event Hubs > clemensvamsp >

Add a task ...

1 Select a template | 2 Authenticate | 3 Configure | 4 Review + create

Select an automation template.

Send monthly cost for resource
Sends the cost of the resource every month.
Select >

Replicate from Event Hub to Event Hub
Replicate content from an Event Hub to another Event Hub
Select >

Replicate ... Bus queu...
Replicate ... a queue
Select >

Review + create

Dashboard > clemensvamsp >

Add a task ...

1 Select a template | 2 Authenticate | 3 Configure | 4 Review

Selected template: Replicate from Event Hub to Event Hub

This task will connect to the following services. Select the connection to add your

Connections

Name	Status	Con
Event Hubs (source)	✓ Connected	Usi
Event Hubs (target)	Not connected	Cre

Review + create | Previous | Next : Configure >

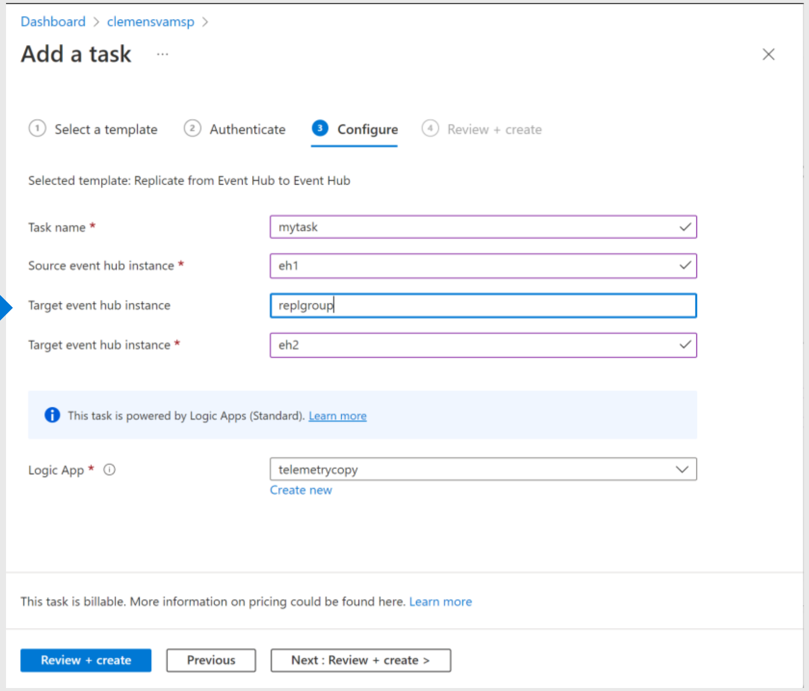
Connect
Event Hubs (target)

Display Name *: amsr

Event Hubs Namespace *: clemensvamsright

Connect via connection string

Create | Cancel



Dashboard > clemensvamsp >

Add a task ...

1 Select a template | 2 Authenticate | 3 Configure | 4 Review + create

Selected template: Replicate from Event Hub to Event Hub

Task name *: mytask

Source event hub instance *: eh1

Target event hub instance: replgroup

Target event hub instance *: eh2

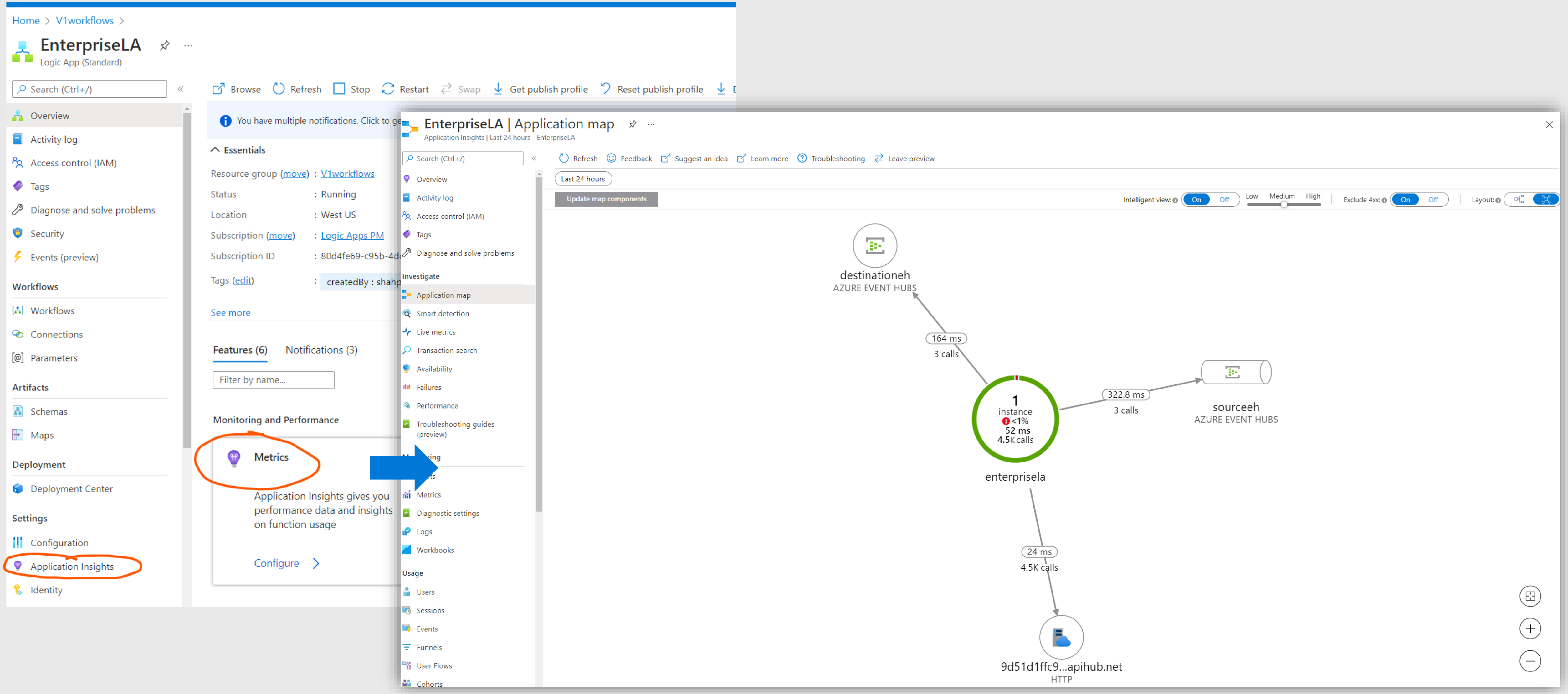
This task is powered by Logic Apps (Standard). Learn more

Logic App *: telemetrycopy
Create new

This task is billable. More information on pricing could be found here. Learn more

Review + create | Previous | Next : Review + create >

EventHub replication visualization



Home > V1workflows >

EnterpriseLA

Logic App (Standard)

Search (Ctrl+/)

Browse | Refresh | Stop | Restart | Swap | Get publish profile | Reset publish profile

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Security
Events (preview)

Workflows
Workflows
Connections
Parameters

Artifacts
Schemas
Maps

Deployment
Deployment Center

Settings
Configuration
Application Insights
Identity

You have multiple notifications. Click to ge

Essentials

Resource group (move)	: V1workflows
Status	: Running
Location	: West US
Subscription (move)	: Logic Apps PM
Subscription ID	: 80d4fe69-c95b-4d
Tags (edit)	: createdBy : shahp

See more

Features (6) | Notifications (3)

Filter by name...

Monitoring and Performance

Metrics

Application Insights gives you performance data and insights on function usage

Configure >

EnterpriseLA | Application map

Application Insights | Last 24 hours - EnterpriseLA

Search (Ctrl+/)

Refresh | Feedback | Suggest an idea | Learn more | Troubleshooting | Leave preview

Last 24 hours

Update map components

Intelligent view: On Off | Low Medium High | Exclude 4xx: On Off | Layout:

Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems

Investigate
Application map
Smart detection
Live metrics
Transaction search
Availability
Failures
Performance
Troubleshooting guides (preview)

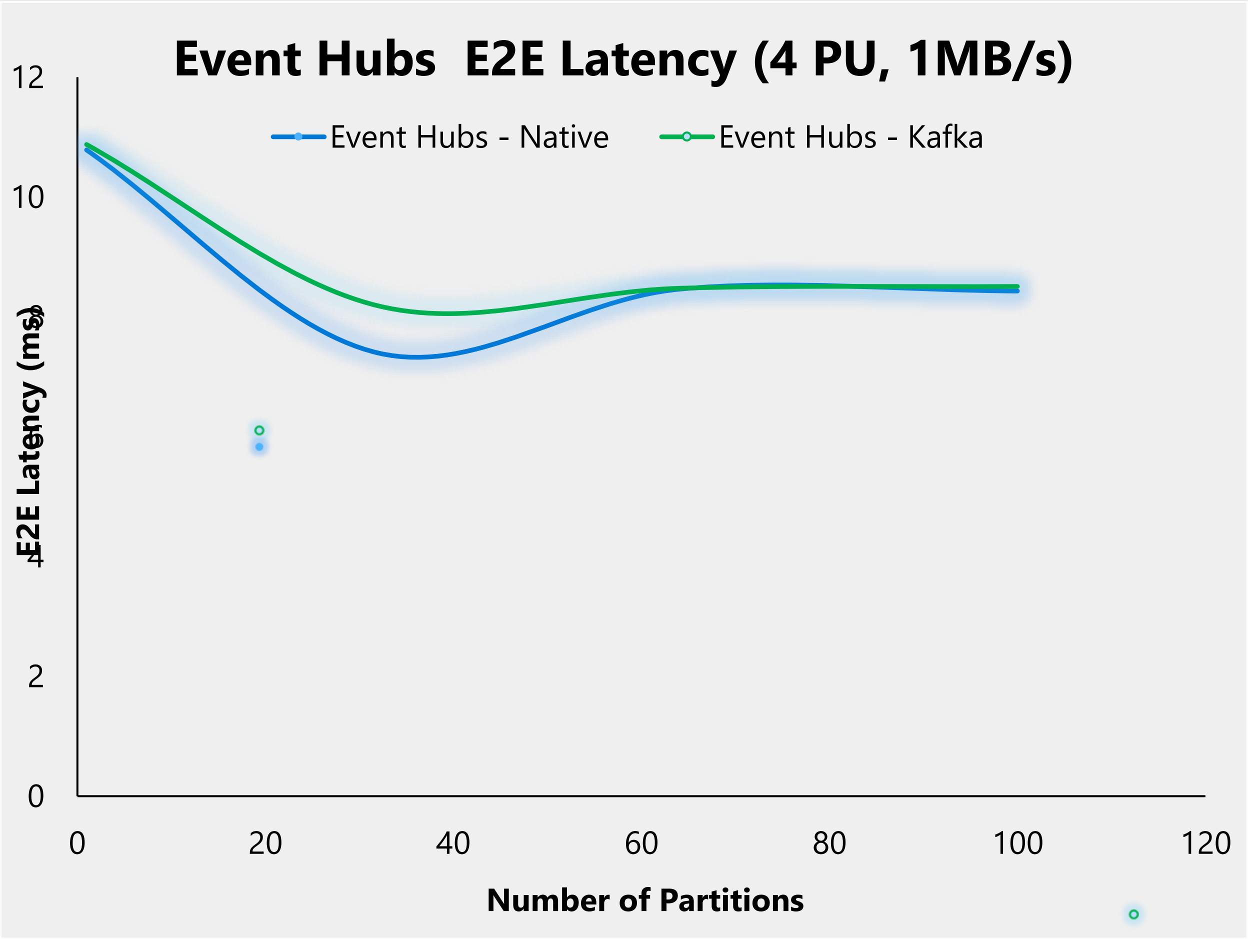
ring
ts
Metrics
Diagnostic settings
Logs
Workbooks

Usage
Users
Sessions
Events
Funnels
User Flows
Cohorts

Event Hubs Premium

Performance!

Event Hubs Premium is *Fast*!

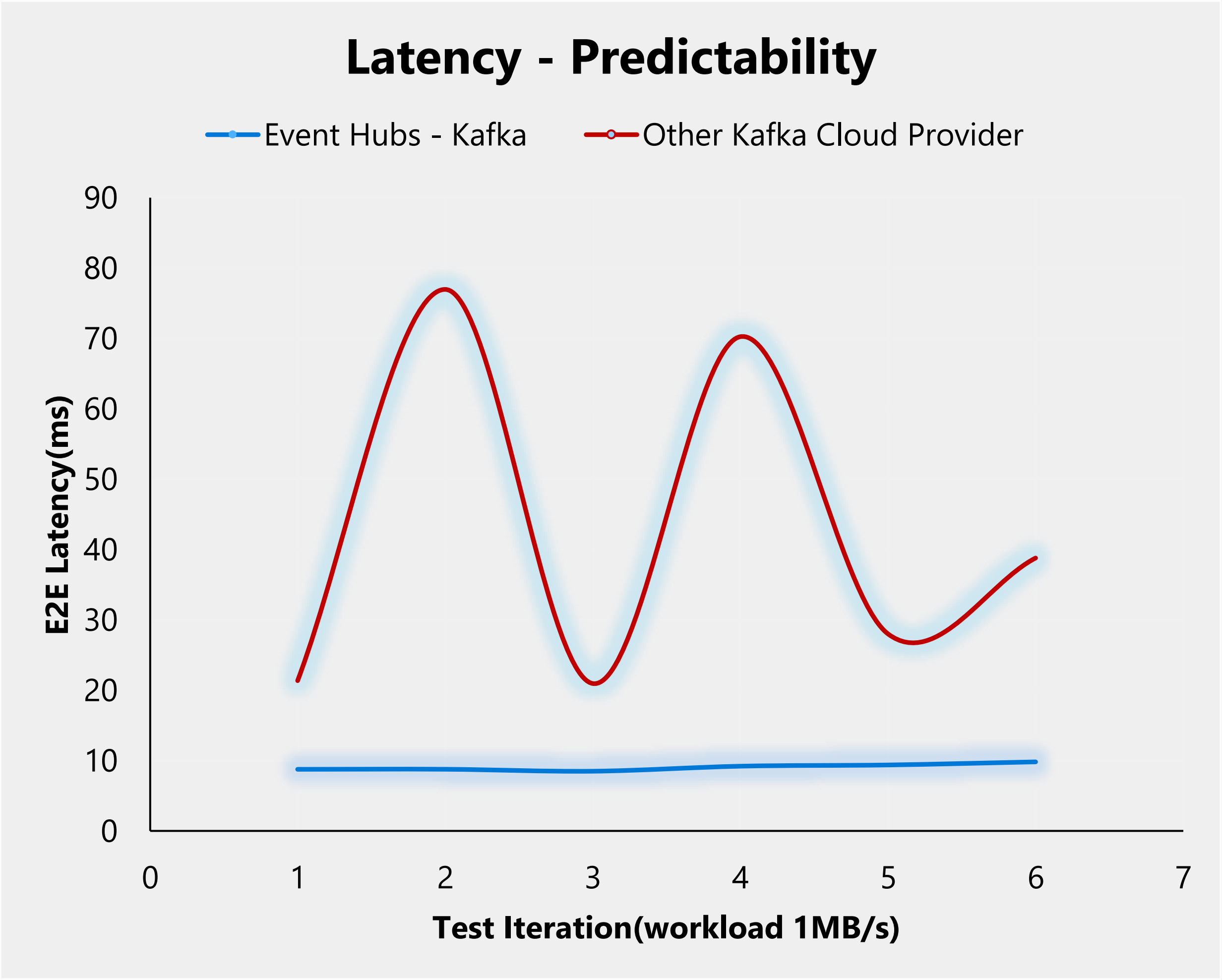


Event Hubs Premium end-to-end latency < 10ms for most Event Streaming workloads.

Performance Metric	Expected Range	Comments
Throughput	~5-10 MB/s per 1 PU	Maximum throughput can be achieved by scaling partitions For both Kafka and AMQP
Latency	<10 ms	With 1 MB/s load, 4 PU namespace For both Kafka and AMQP

Event Hubs Premium – Predictability

Event Hubs Premium offers consistent end-to-end latency of < 10ms.



Effective Global Reliability

Uptime – Is the cluster available to accept connections?

100%

Reliability – Are individual operations suceeding?

>99,9999% weekly average

Azure Event Hubs

Platform-as-a-Service Event Stream Broker

Use the Apache Kafka® API, but with far lower cost and better performance.

Fully managed: You use the features, Azure deals with everything else

AMQP 1.0 standards compliant, Apache Kafka® wire-compatible

Polyglot Azure SDK and cross-platform client support

Industry-leading reliability and availability

Fast.

That was a lot, wasn't it?

Q&A

Microsoft

© Copyright Microsoft Corporation. All rights reserved.