

IBM MQ Hybrid Cloud Architectures

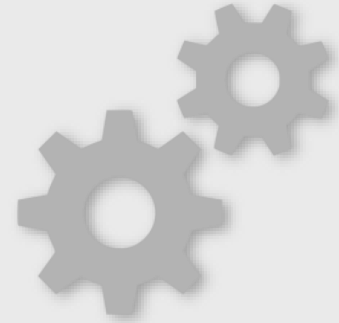
Jamie Squibb
IBM UK

November 2018
Session **JK**



Agenda

- Why Hybrid Cloud?
- Benefits of messaging?
- Adopting cloud
- Architectures
- Connectivity
- Containers



What are the business pressures driving the cloud requirement?

Driving agility, to support new revenue streams

- New engaging apps
- Innovation groups
- Enable developers via self service
- Developer choice leading to new technologies

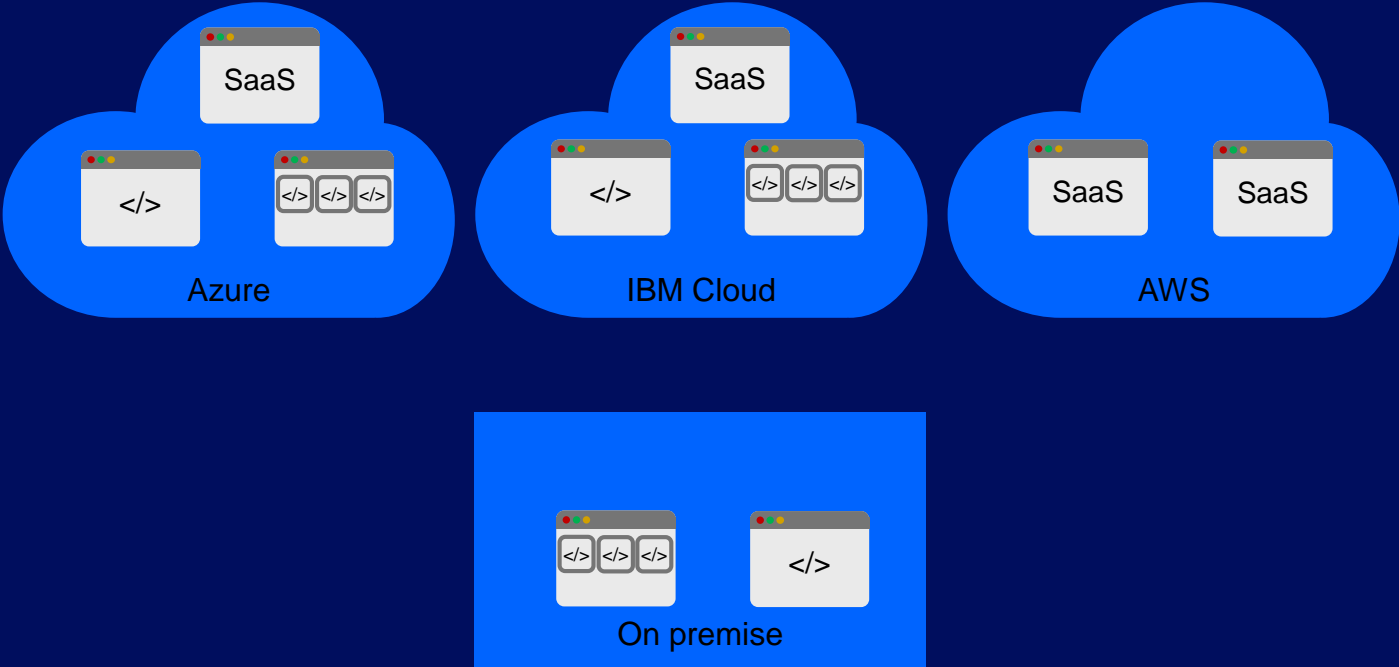
Focus on core competencies

- Lift and shift applications to the cloud
- Migrate to SaaS

Total cost of ownership

- Reduce the lifecycle cost of software
- Promote SaaS to reduce maintenance and ever-greening costs

A modern enterprise will include many clouds, some explicit, some implicit



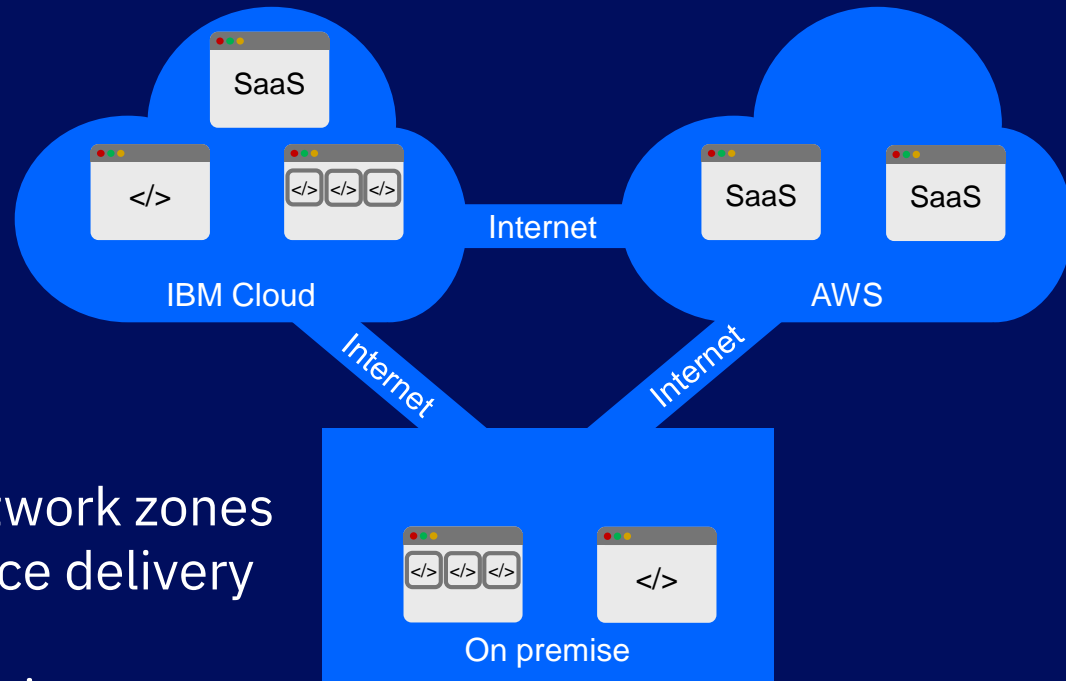
Challenges for **business critical data** in a multi-cloud strategy, and what capabilities are needed?

Challenges

- Store and forward
- Network isolation
- Firewalls
- Reliability
- Security

Capabilities

- Message routing across network zones
- Assured reliable exactly once delivery
- Message store
- Flexible communication choices

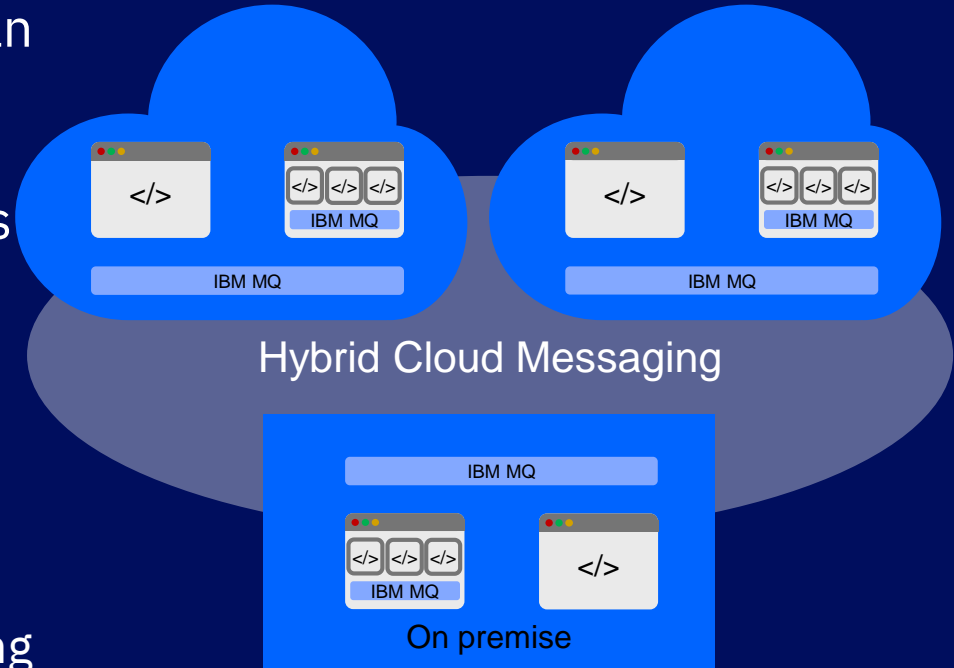


How messaging can help

Fragile networks between clouds can lead to a weak multi-cloud solution.

IBM MQ overcomes fragile networks enabling business critical data to be exchanged.

- Messaging available within each environment to assure local access
- Reliable and secure communication across environments using messaging

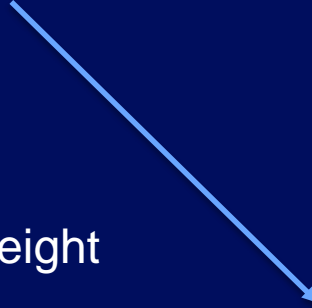


How messaging can help

All the benefits of cloud, with access to your enterprise data



- Doing more with less
- Being more ready to change
- Making the development process less heavyweight
- Paying for what you use
- Integrating with other cloud services
- Rapidly scaling up and down with demand

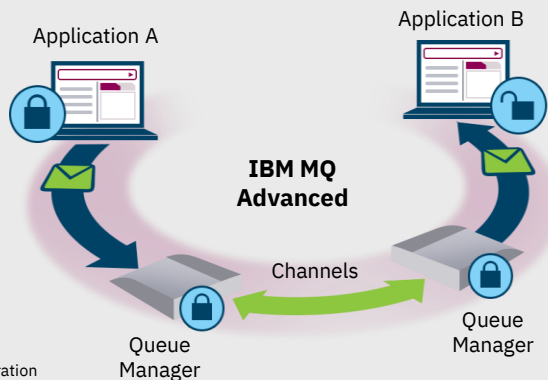


- Customer profiles
- Purchases (online orders)
- Data requests (e.g. insurance quotes)
- Website comments

End-to-end message level security

MQ Advanced Message Security (AMS) provides the capability to encrypt messages **in transit** and **at rest** between sender and receiver.

1. Automatically encrypted by the sending client so that it can only be decrypted by the intended recipient
2. Or encrypted by the queue manager on receipt, for cases where the application deployment cannot be updated



Benefits

- No application code changes required
- Goes beyond TLS channel security, which only protects data in transit between processes
- Message data can only be read by the intended receiving application code
- Not on the queue by the system administrator
- Not on the disk by your infrastructure or cloud provider
- **Proven, trusted approach to fulfilling compliance requirements such as GDPR, PCI, HIPAA, etc.**


Run IBM MQ in any location or cloud exactly as you need it



Celebrating
25 years

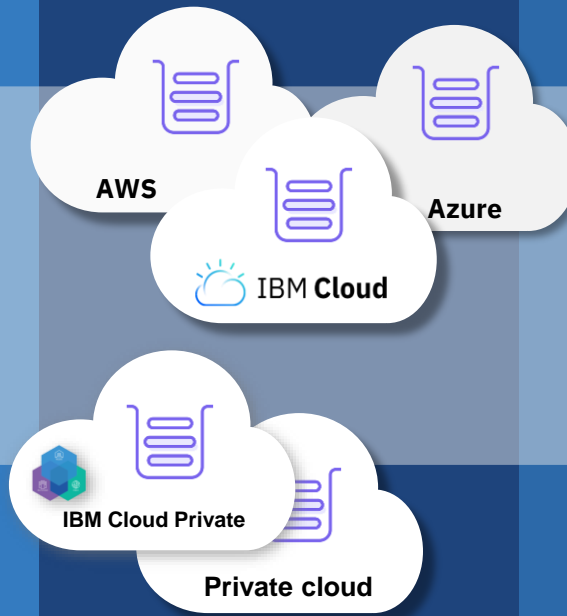
On-premise, software and the MQ Appliance



IBM Z 
Linux **AIX**
Windows **Solaris**
HPE **IBMi**
Appliance **...**



Run it yourself in any cloud, public or private



Let IBM host it for you with its managed SaaS MQ service in public cloud



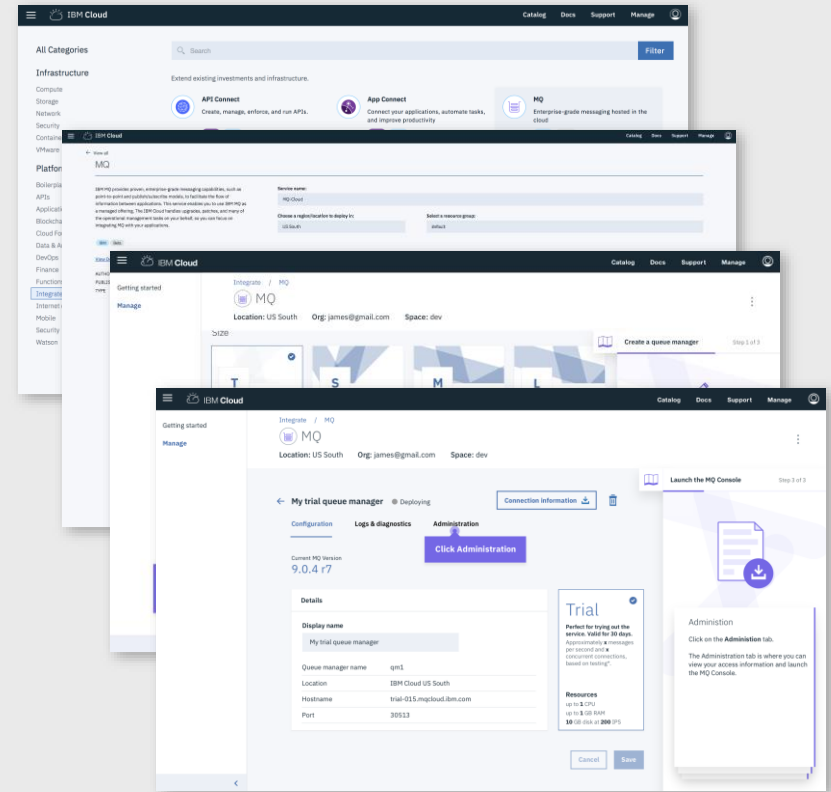
MQ on Cloud

Have IBM provision your queue managers directly into the Cloud

IBM owns the infrastructure and the responsibility to keep the systems up to date and running

The user owns the configuration and the monitoring of the messaging

Try the service for free
www.ibm.com/cloud/mq



Rethink your use of MQ

Don't confuse old MQ practices for MQ itself

"MQ is too hard to use"

"Our MQ system is too complicated to change"

"MQ isn't cloud, it's too old!"



How many of these do you have?

- Hand crafted, shared queue managers
- Applications hard coding connection details
- Applications bound to a single IP address
- Edge security at most
- Internal architecture complexity exposed to the applications
- A lengthy change control process
- Manual installation, deployment and configuration

Dare to re-invent
your approach!

Managing MQ

Management of systems is evolving towards **cattle** not **pets**. What does that mean for MQ?

Consistent configuration and operations across multiple queue managers

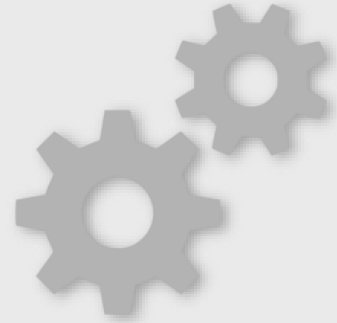
Automated deployment

Configuration as code

Self service

Collection and analysis of diagnostic data

Simple integration into standard DevOps and automation tooling



Help and advice with MQ in the Cloud

It has always been important for MQ to run where it is needed and integrated into the tools of your choice

For many that means **clouds**

We have been investigating and demonstrating running MQ in various **public** and **private** clouds.

Using a variety of tooling for provisioning, configuration and monitoring

And we've been sharing that information for everyone to use

IBM developerWorks

Marketplace

David Ware

IBM Messaging for admins and developers

Products Downloads Videos Sample Code Blogs Docs Get Help

MQ on Cloud

All IBM MQ MQ on Distributed MQ Appliance MQ for z/OS MQ on Cloud

Why IBM MQ on Cloud

MQ started life in the data center but you can now take the many benefits of IBM MQ to the cloud, its security, robustness and integration capabilities. And by doing this you'll also reap the benefits that a cloud deployment brings, whether it's automated deployments, scalable architectures, centralised administration. If you're running MQ in a cloud environment, visit the MQdev blog to see examples of how to run IBM MQ on AWS, Azure, OpenStack and Docker. You can also see examples of how to centralize key MQ metrics and MQ error logs.

Latest AWS blog updates

- New! AWS Quickstart for IBM MQ
- IBM MQ - Using Active Directory for authorisation in Unix queue managers
- IBM MQ - Using AWS CloudWatch to monitor queue managers
- MQ on AWS: Sending MQ error logs to CloudWatch
- MQ on AWS: PoC of high availability using EFS
- Basic deployment of MQ on AWS

Latest OpenStack blog updates

- MQ on OpenStack, part three: Automated client connection PoC using MQ v9 CCST URL feature.
- MQ on OpenStack, part two: Managing an MQ environment using Heat
- MQ on OpenStack, part one: Creating an image using Packer

Latest Docker blog updates

- New! Updates to the MQ Docker support statement
- New! Running the MQ docker image on the Kubernetes service in Bluemix
- Introducing the IBM MQ image on the Bluemix Container Service
- MQ in Docker is now supported for production use
- Introducing a Docker image for MQ Advanced for Developers

Latest Error logging blog updates

- MQ on AWS: Sending MQ error logs to CloudWatch
- Sending MQ error logs to the Bluemix Lognet service
- Storing and searching MQ error logs in Elasticsearch

Latest Monitoring blog updates

- Sending MQ metrics to the Bluemix Lognet service
- IBM MQ - Further integration with open-source monitors
- IBM MQ - Using Prometheus and Grafana to monitor queue managers

GitHub Repositories

- AWS
- Azure
- OpenStack
- Docker

developer.ibm.com/messaging/mq-on-cloud



Marks & Spencer – UK retailer

MQ PRINCIPLES

- Cloud QMs are Disposable & Immutable
- Script Everything
- Dedicated MQ QMs per App
 - Client connections
 - JMS
 - Common Message Envelopes
- MQ QM 't-shirt' Models for on-Prem

M&S
EST. 1884



© 2018 Marks and Spencer



MQ and Cloud at Marks & Spencer

Session #9176
James Blackburn
IBM Think, March 2018

M&S
EST. 1884

© 2018 Marks and Spencer

LEARNINGS

MQ Cloud Journey challenged us to 'Think Differently'.

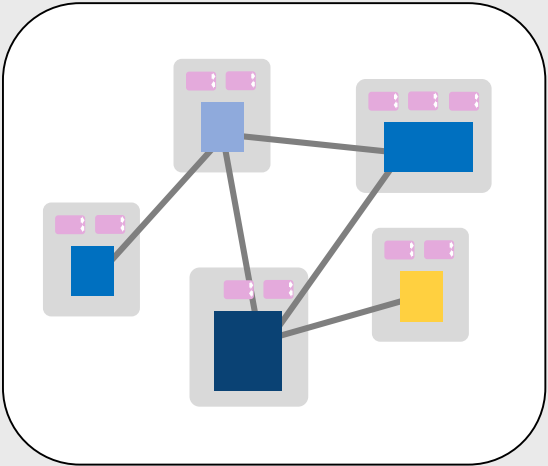
- Bigger fewer QMs are often more complex to manage
- MQ on Cloud, 'just worked'
- MQ CD stream is stable
- MQ QMs really can be disposable
- Everyone can be an MQ admin
- Delivery Organisation, a Hybrid Team for a Hybrid Solution

M&S
EST. 1884

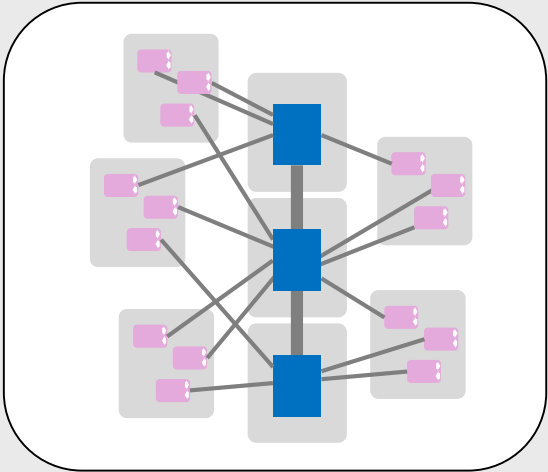
© 2018, Marks and Spencer

think 2018

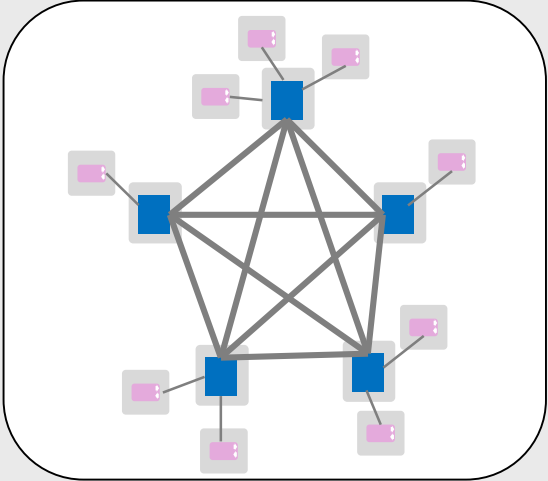
Typical MQ architectures



Classic



Hub



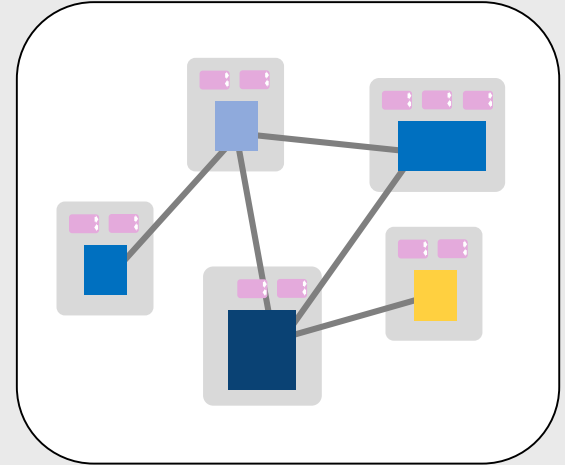
Decentralized



More suited to cloud scenarios

Classic 'pet' architecture

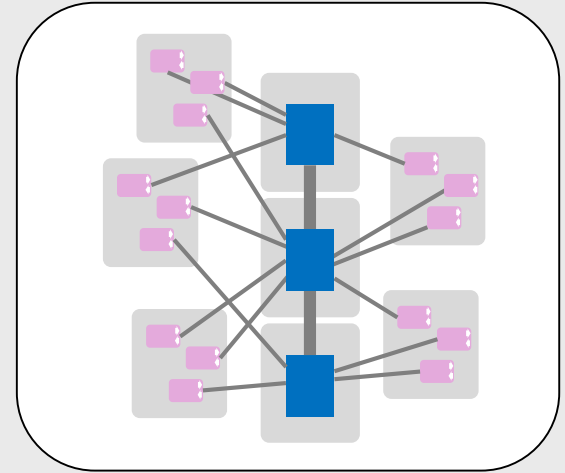
- Used for connectivity of heterogeneous systems, providing store and forward to overcome system and network outages
- Isolation through dedicated queue managers, tightly bound to the application runtimes
- This is one of the 'original' deployment patterns for MQ and often results in a bespoke, tuned deployment for individual components
- Can lead to **hard to deploy, manage** and **maintain** systems over time



Classic

Hub architecture

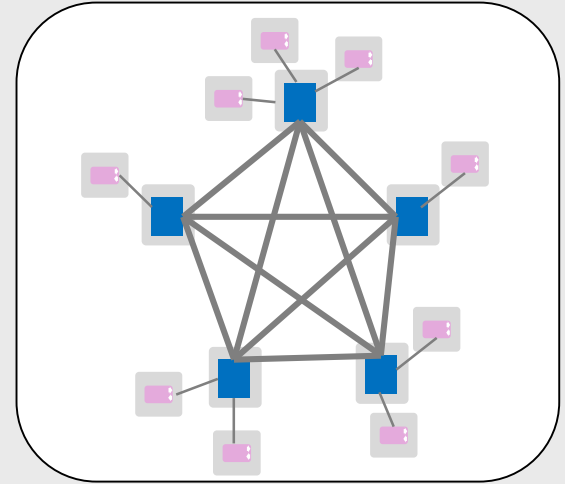
- A 'hub' (or backbone) of systems running multiple queue managers, based on a standard deployment
- Applications connecting as clients from remote systems. Looser coupling enables simpler deployments and independent scaling and maintenance
- This pattern has gained popularity as networks improve and administration costs go up



Hub

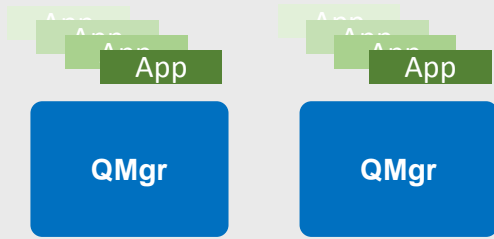
Decentralized architecture

- Each line of business or application has its own infrastructure and therefore its own queue managers
- Client connections to separate applications from the infrastructure
- Minimized central administration to reduce bureaucracy and speed up application deployments
- Popular as a way to satisfy greater autonomy for lines of business



Decentralized

Tenancy



Multi tenant

Potentially lower runtime overheads

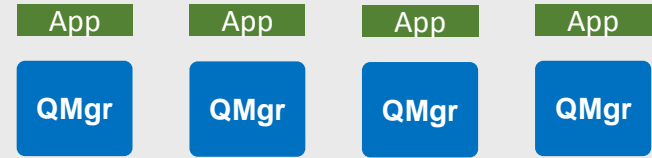
More care needed when configuring to achieve isolation

- Isolation of machine resources not possible

Harder/simpler to monitor

- Depends on your view of more queue managers

Fine-grained security required



Single tenant

Simple to configure, maintain and monitor

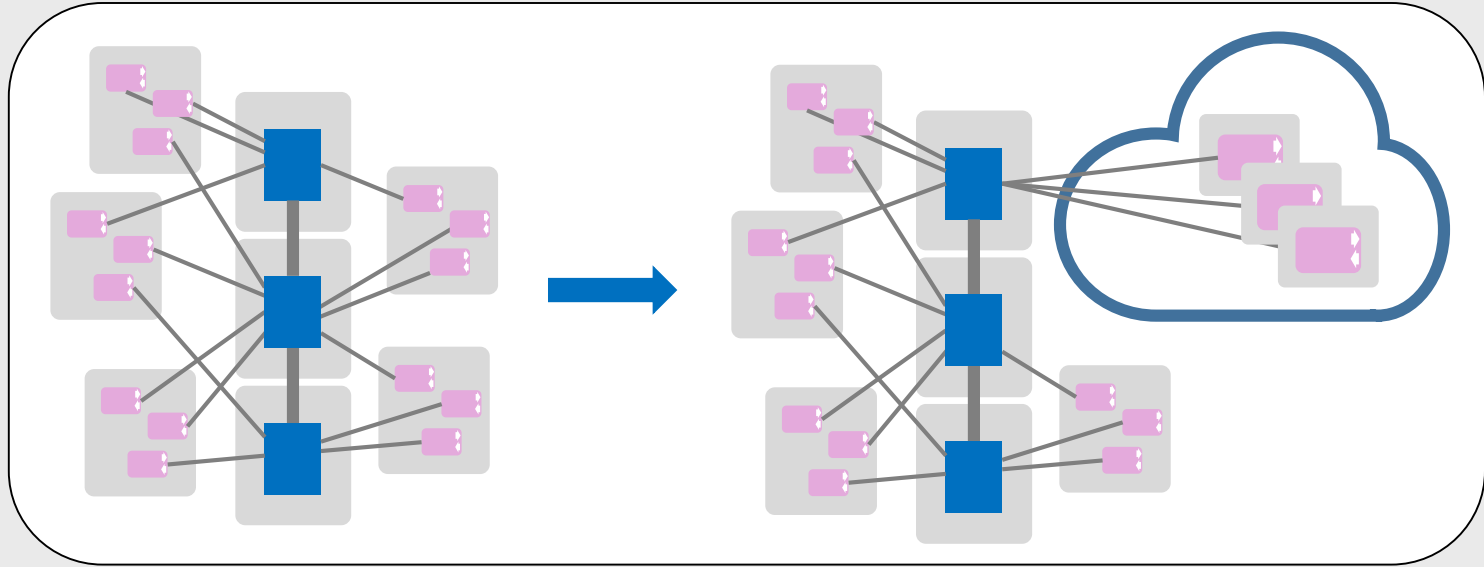
Very good isolation

A proliferation of queue managers

Harder when integration is required

Best suited to scalable, cloud deployments

Hybrid architectures - clients

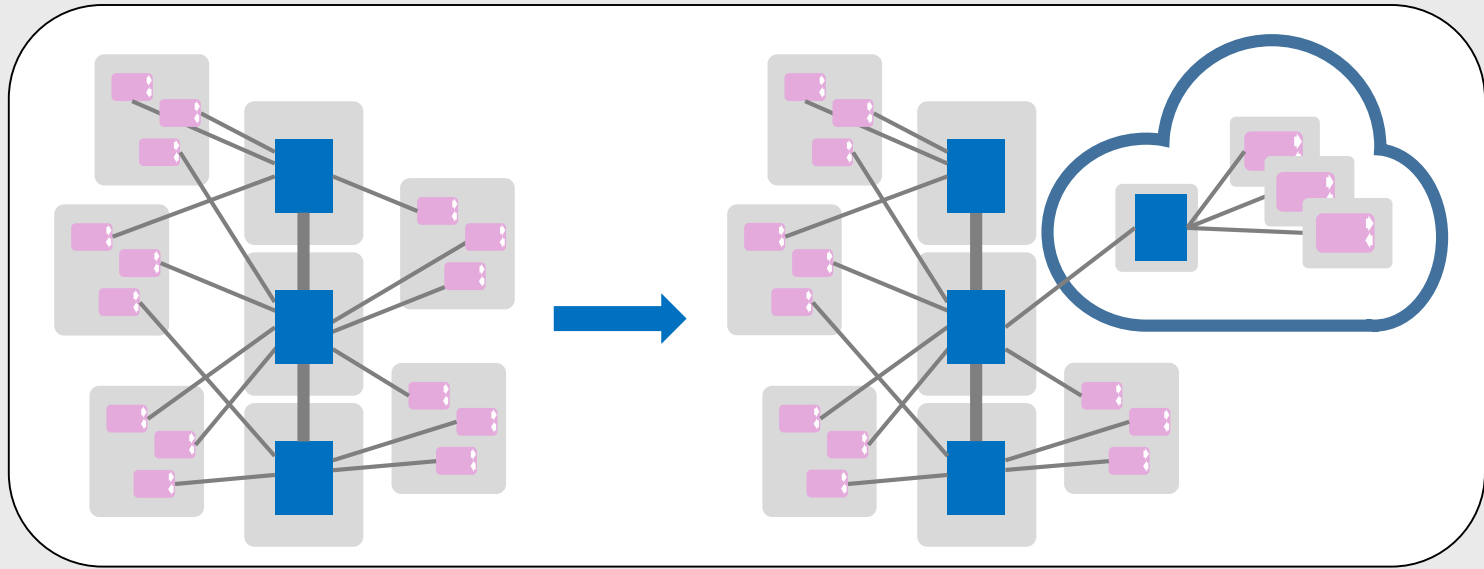


Run MQ clients in the cloud - connect to on-premise hub

Applications running in container, Cloud Foundry, server-less environment

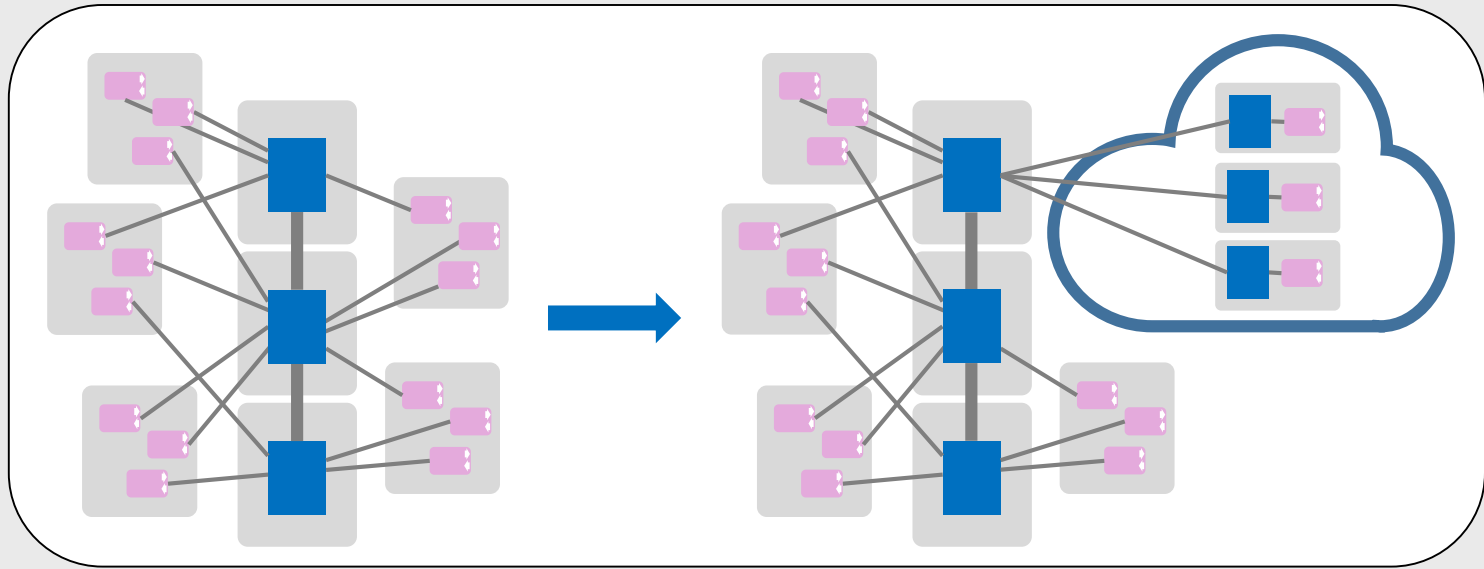
On z/OS, need to be aware of CHINIT costs/scaling of large number of clients

Hybrid architectures - clients and gateway queue manager



- Single queue manager running in the cloud
- Gateway queue manager connects to on-premise hub
- Not multi-tenancy - apps are scaled instances
- Provides for local communication between cloud apps without going back to on-premise
- Gateway reduces CHINIT costs on z/OS
- Good place to exploit MQ on Cloud

Hybrid architectures - clients and queue managers



Queue managers run in the cloud alongside apps

Run in VMs or containers - connect to on-premise hub

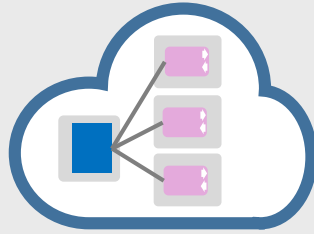
Running queue managers alongside apps might not be the best architecture for cloud

Hybrid architectures - comparison



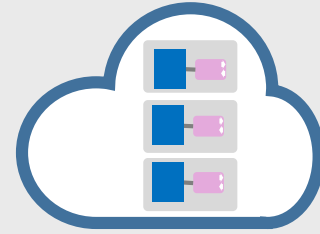
Clients

- ✓ Easier to scale
- ✓ Stateless
- ✓ Less administration
- ✗ Need to discover a QM
- ✗ Can't operate during network partition



Clients & Gateway QM

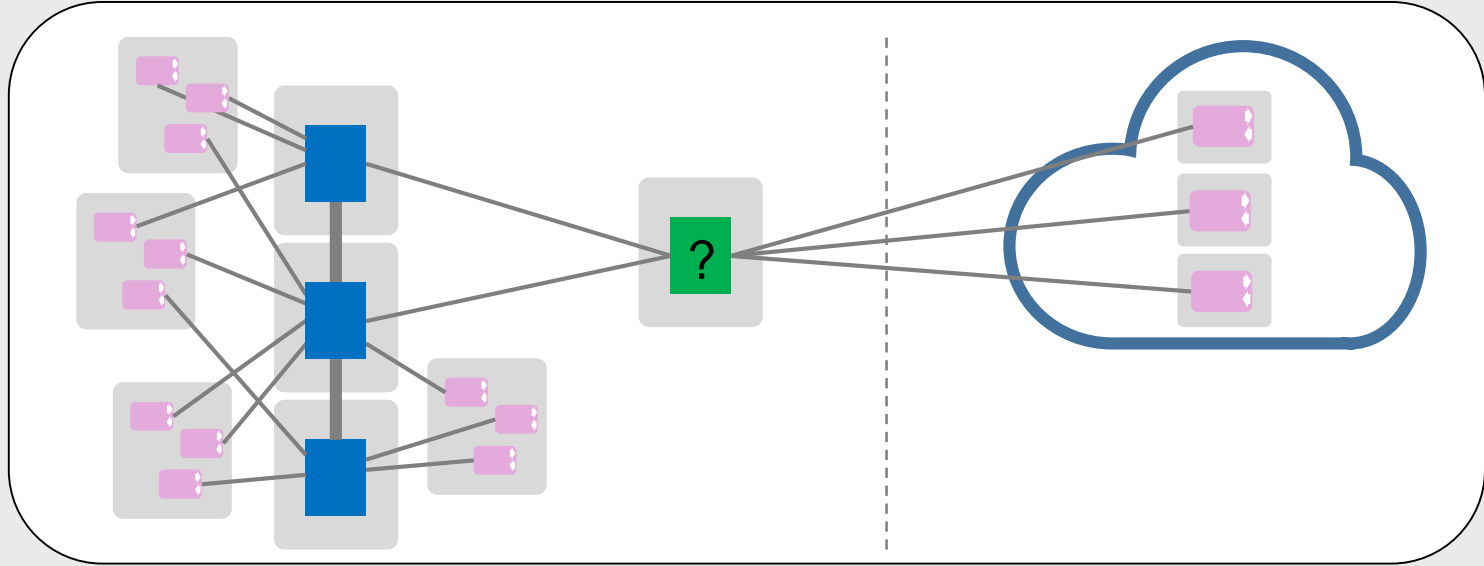
- ✓ Client service discovery simpler
- ✓ QM manages discovery and routing
- ✓ Single place to configure connectivity back to the enterprise
- ✗ Limits app scalability
- ✗ Queue manager a bit like a pet!



Clients & QMs

- ✓ QM buffers messages between outages
- ✓ Client service discovery easier
- ✗ More admin required
- ✗ Need access to each QMs logs
- ✗ Harder to scale down
- ✗ Can apps really do anything during an outage anyway?

Connectivity

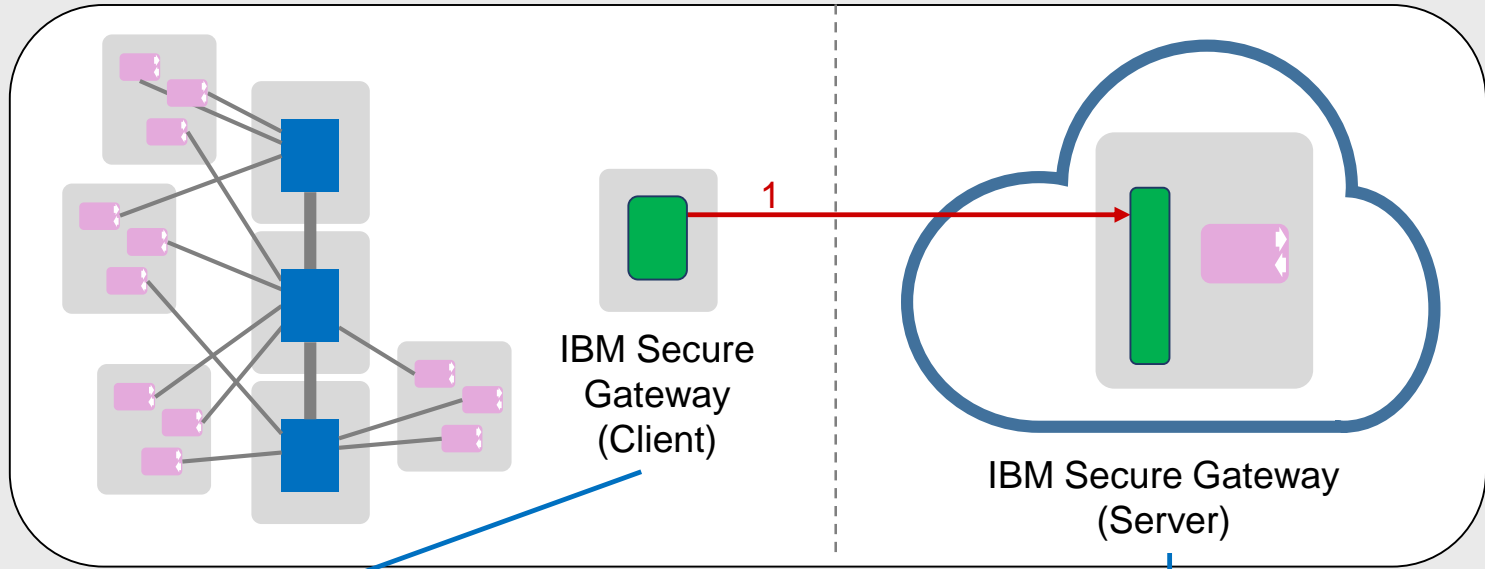


Need to route connectivity through firewall/DMZ (as per from any external network)

All cloud platforms provide ways to connect on-premise and cloud networks

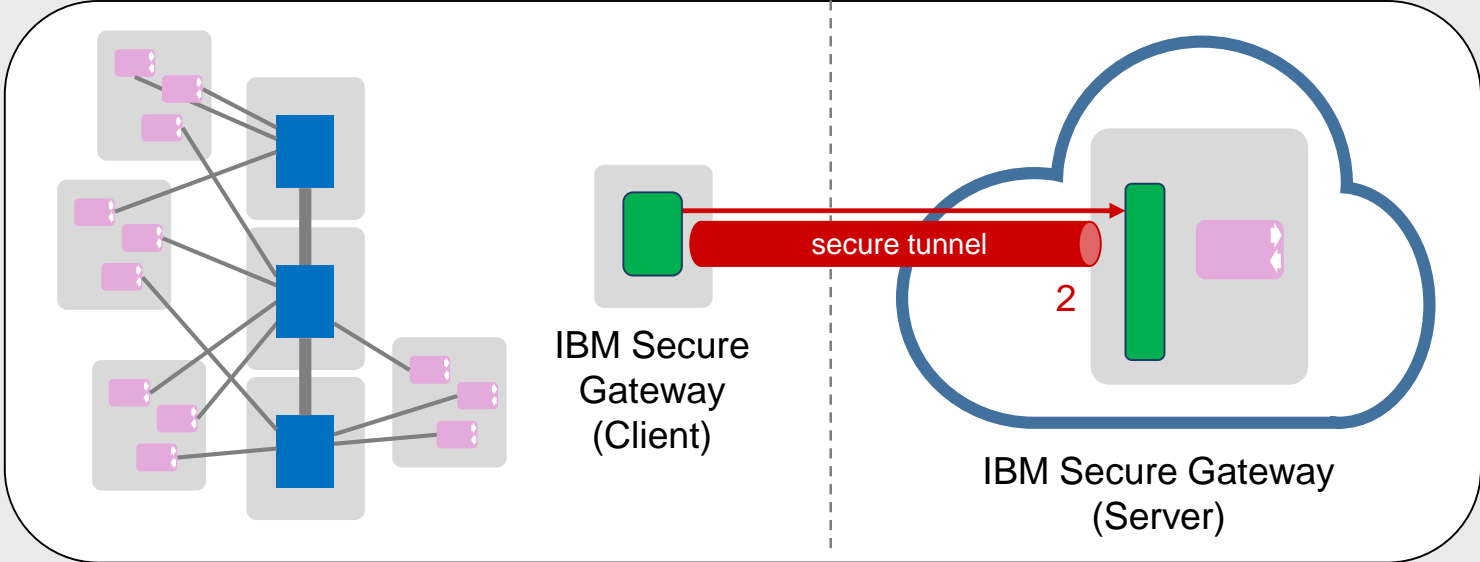
- e.g. IBM Secure Gateway, DirectConnect (AWS), VPN

Connectivity – IBM Secure Gateway



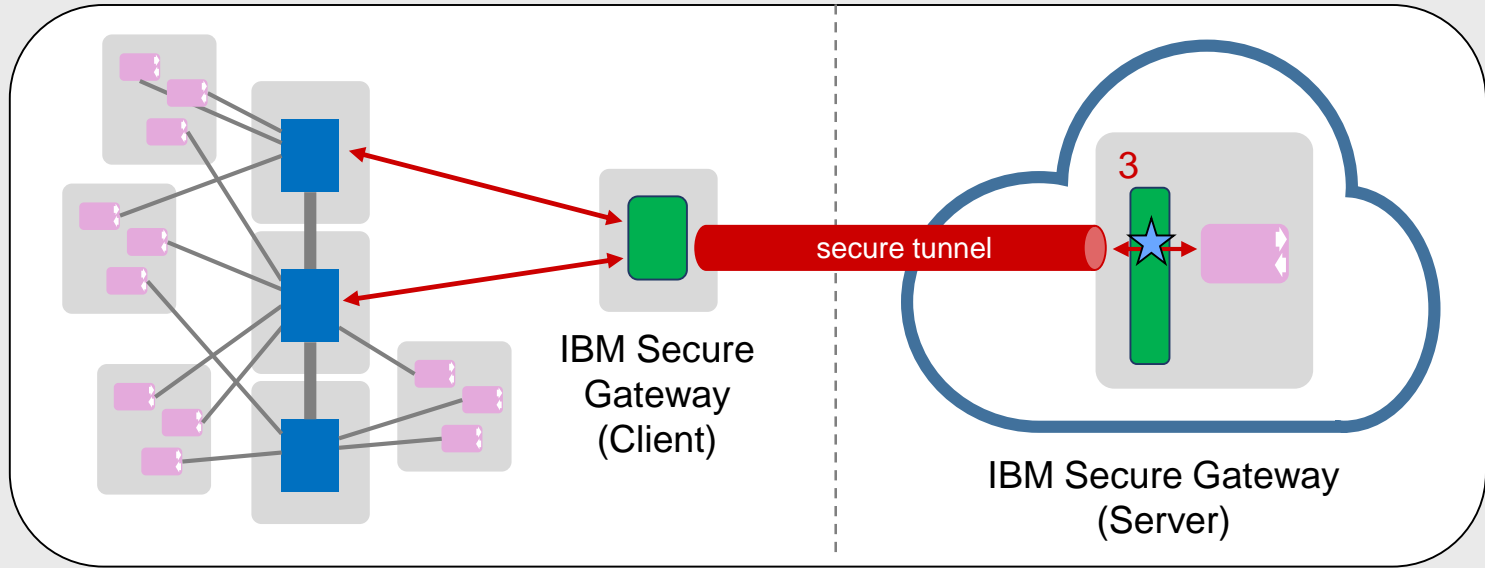
- Secure Gateway client runs on-premise
- Native Mac/Linux/Win app
- Docker
- DataPower
- Connects to Secure Gateway server

Connectivity – IBM Secure Gateway



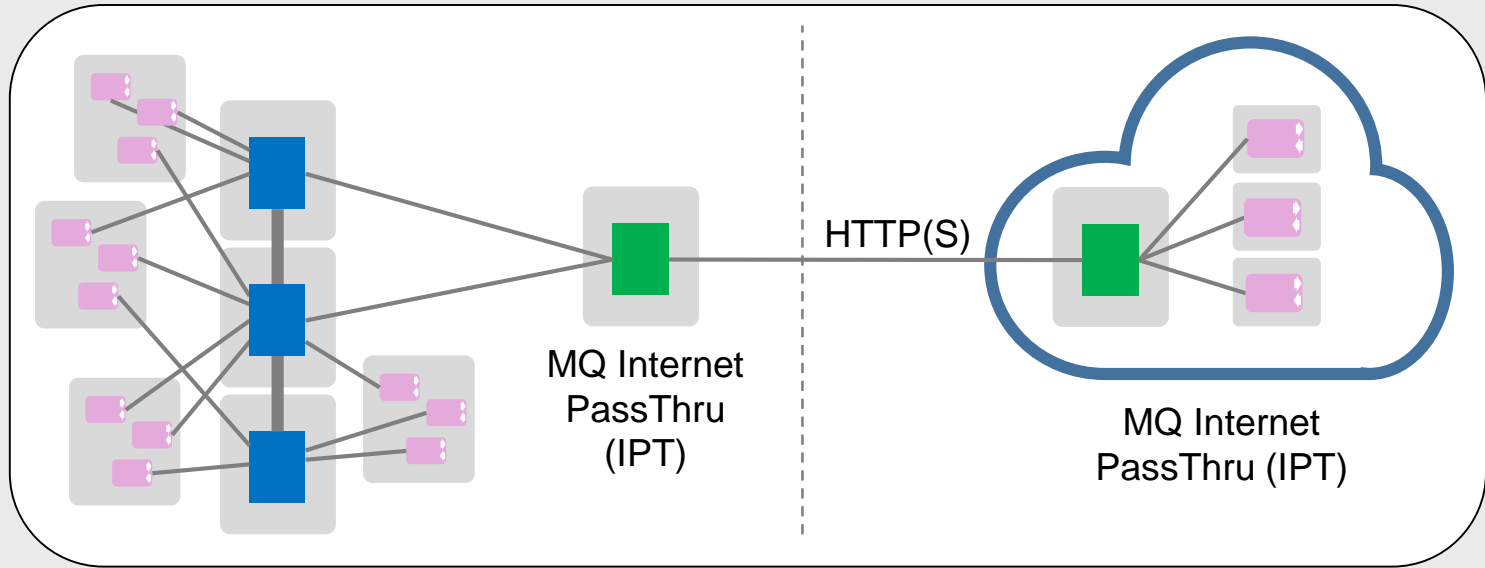
Secure Gateway sets up a tunnel to on-premise client

Connectivity – IBM Secure Gateway



- You configure valid routes from Secure Gateway client to on-premise network interfaces
- Cloud application connects to virtual address in cloud, e.g. `cap-sg-prd-1.integration.ibmcloud.com:17036`
- Secure gateway client routes packets to/from on-premise network
- Connectivity from app to tunnel secured with TLS and/or restricted IP ranges

Connectivity – MQ Internet PassThru (MS81)



Avoids the need for a direct TCP connection from cloud to on-prem

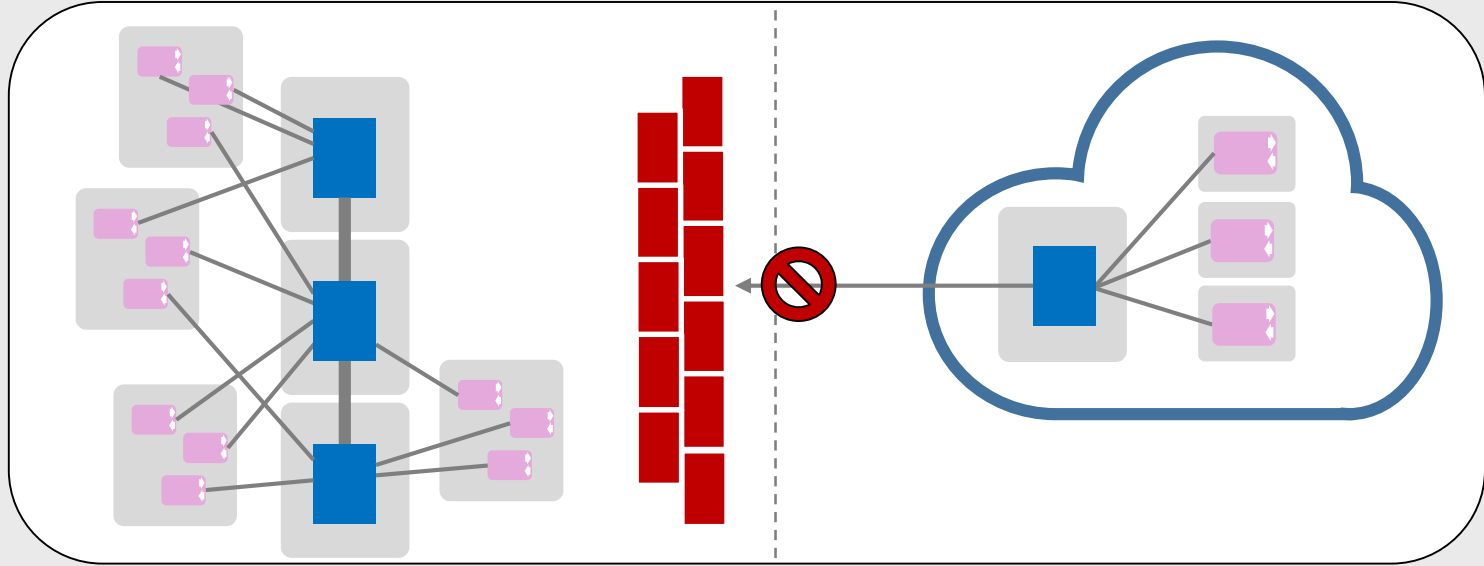
Tunnel MQ traffic over HTTP(S)

Avoids requirement for more complicated VPN configuration

Re-use on-prem IPT if you're already using it

Cloud agnostic

Connectivity – Server-requester channels

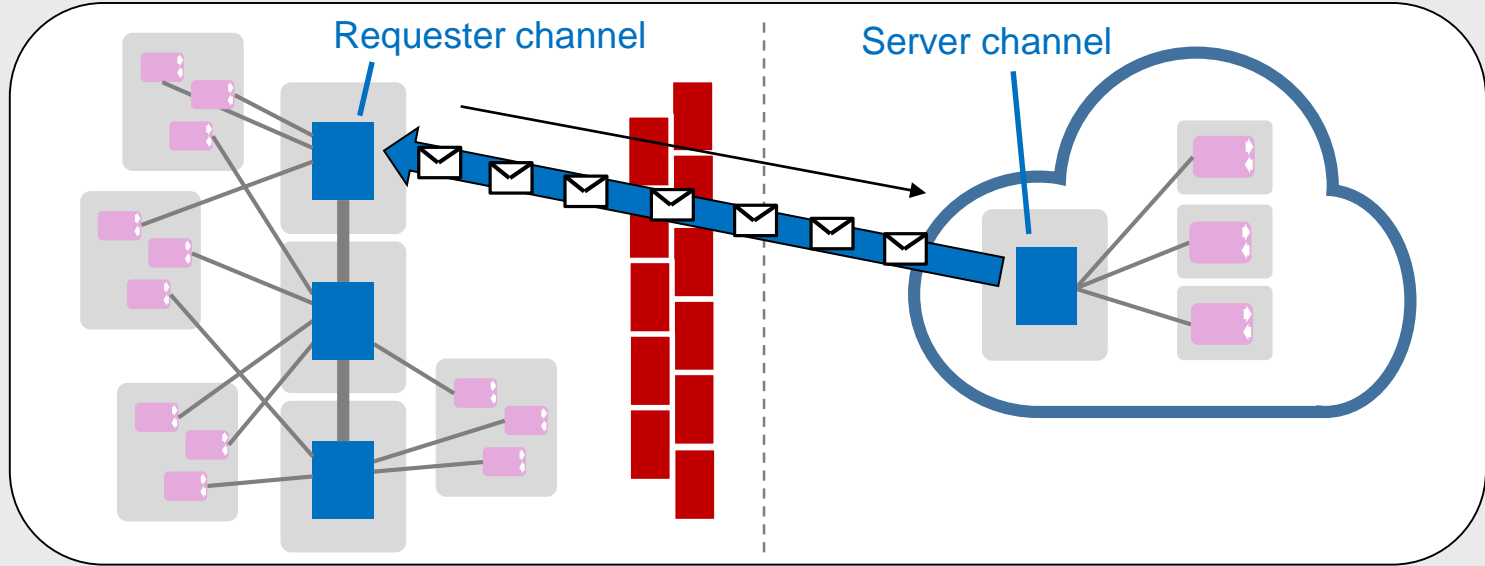


Enterprise network behind firewall

Cloud queue manager on public facing IP address

Cloud can't connect directly to enterprise queue manager, but ...
... enterprise queue manager can connect to cloud and request data

Connectivity – Server-requester channels



Requester channel initiates connection to cloud queue manager
Server channel sends data back on the connection initiated by the requester channel

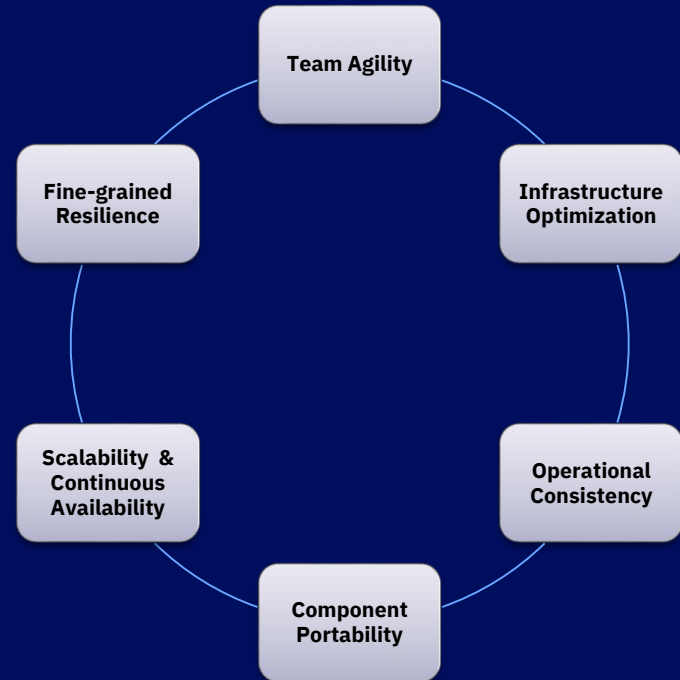
Adopting a cloud native approach for your MQ Estate

Introducing containers...

Package software into standardized units for development, shipment and deployment.

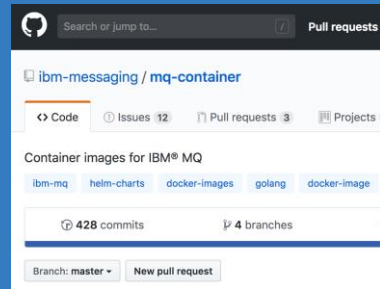
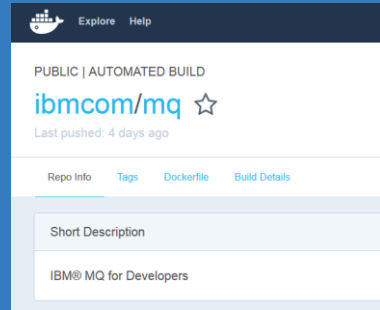
Containers are the native approach for deploying across a hybrid multi-cloud environment.

Containers provide numerous benefits, which IBM MQ can exploit, to modernize their estate.



MQ in Containers

MQ has been supporting Docker containers since 2015 with images on Docker Hub and Docker Store and sample setups on Github



[github.com/
ibm-messaging/
mq-container](https://github.com/ibm-messaging/mq-container)

MQ provides Helm charts for deploying MQ into Kubernetes platform, on-prem or on cloud, such as IBM Kubernetes Service



MQ Advanced is available as a fully supported product with **IBM Cloud Private**

deploy IBM certified software containers into an IBM provided Kubernetes platform or an existing Red Hat OpenShift ^{new}



IBM Cloud Private Solution Overview



IBM Middleware & Open Source – e.g. Data, Analytics and Developer Services

Cloud-enabled middleware, application runtimes, messaging, databases & analytics to optimize current investments and rapidly innovate



Core Operational Services

To simplify Operations Management, Security, DevOps, and hybrid integration



Kubernetes-based Container Platform

Industry leading container orchestration platform



Cloud Foundry

For prescribed application development & deployment



Terraform (CAM)

Infrastructure as Code for multi-cloud provisioning to public and on-prem private clouds

Runs on existing IaaS: **vmware**



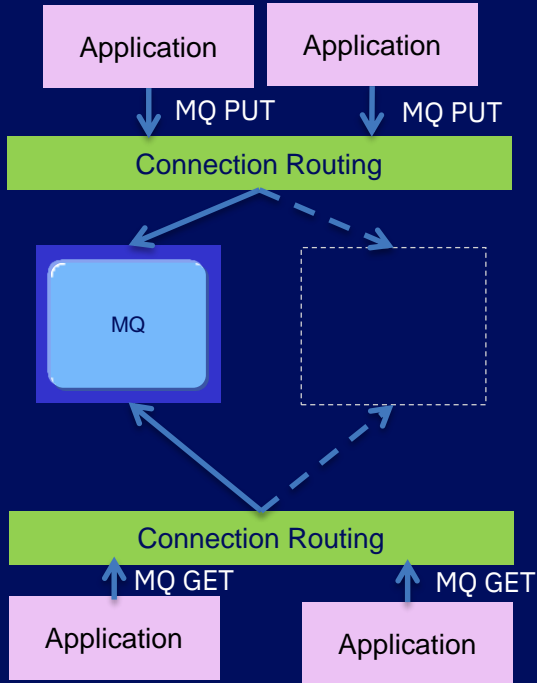
System Z



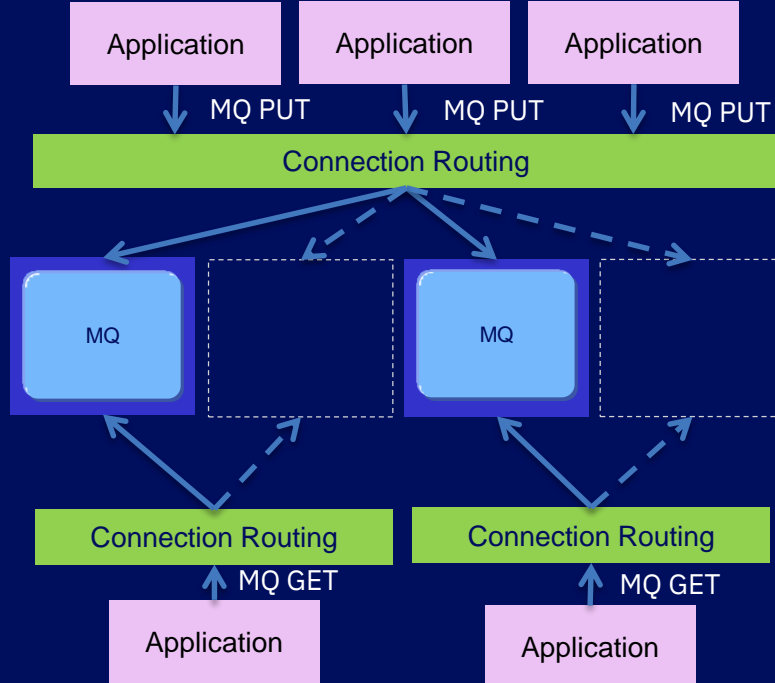
IBM Spectrum

Dell, Cisco, NetApp, Lenovo, ...

IBM MQ continuous availability



Single resilient container

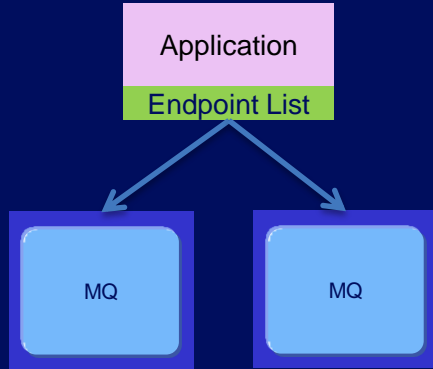


Multiple resilient containers

- ✓ Provide access to store messages, even in failover situations. Connection Routing provided to distribute the traffic across the available instances.
- ✓ Applications retrieving messages attach to the individual Queue Manager. Connection Routing provided to route traffic to container location.

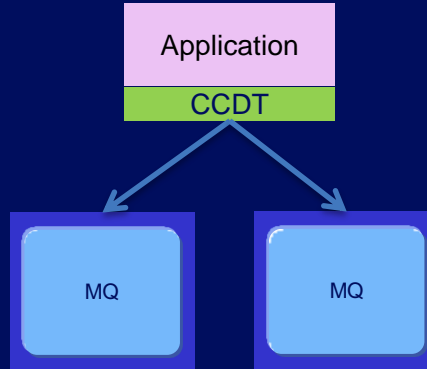
Connection routing

Static routing



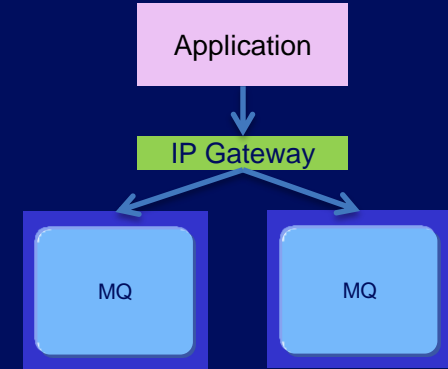
Client embeds endpoints
Performance impact when
primary unavailable
Brittle configuration
No load balancing

Client Connection Definition Table



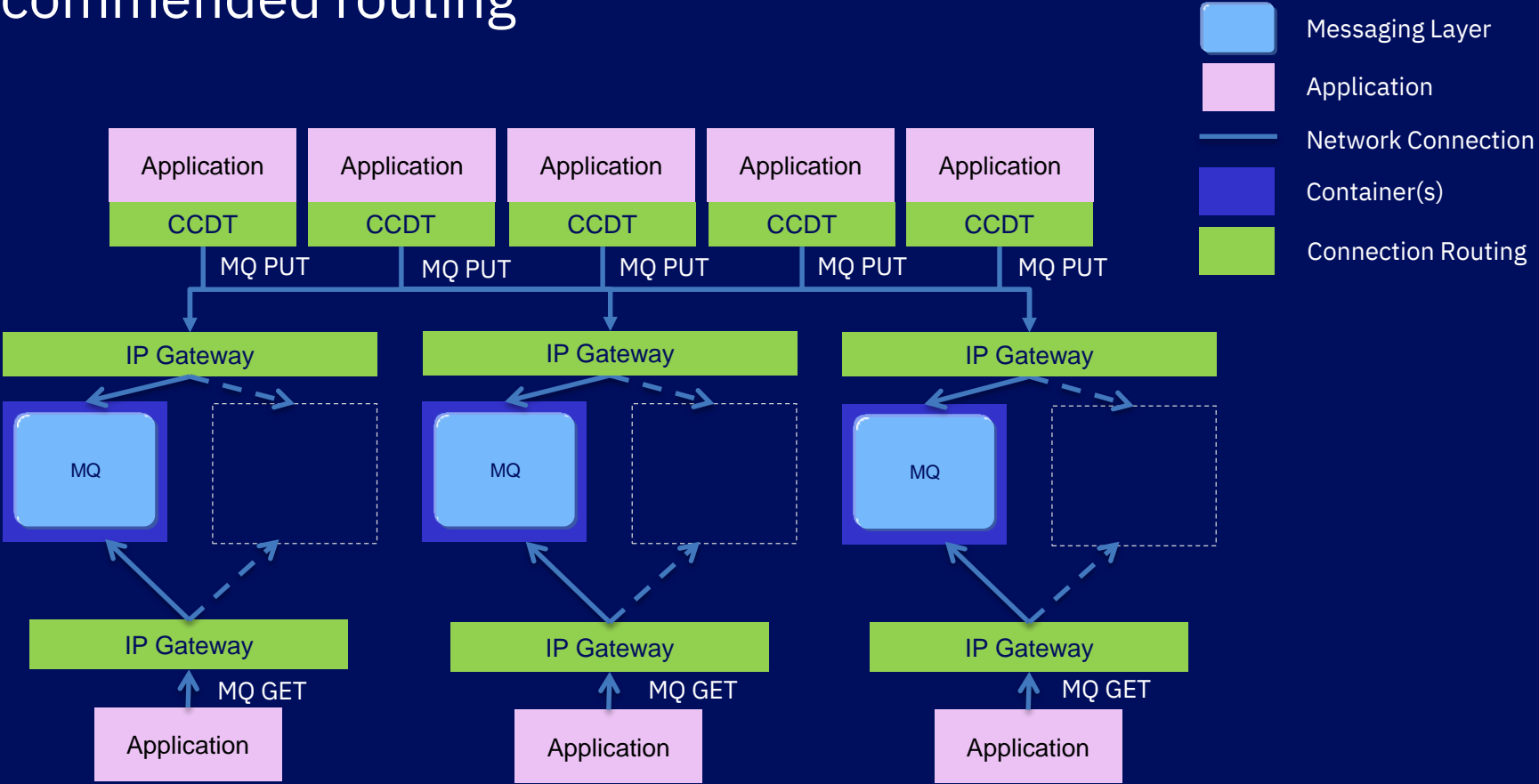
Client references endpoints
Enhanced workload
management strategies
Central configuration
management

Load balancer



Client references endpoints
Enhanced workload
management strategies
Central configuration
management
Not recommended for JMS

Recommended routing





Celebrating
25 years

Please note

IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice and at IBM's sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.

The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

We want your feedback!

- Please submit your feedback online at
 - <http://conferences.gse.org.uk/2018/feedback/JK>
- Paper feedback forms are also available from the Chair person
- This session is **JK**

