JBoss AMQ 7 Technical Deep Dive

Advanced Messaging for the Cloud

Ted Ross
Senior Principal Software Engineer
May 4, 2017

Presentation Outline

- Overview of AMQ7
- Technical Discussion of AMQ 7 Operation
- Cloud-Messaging Demonstration

Overview of AMQ7

AMQ7. GA. Today.

AMQ7 At A Glance

AMQ Broker

AMQ Interconnect

AMQ Clients

AMQ7 Broker

- High performance general-purpose message broker
- Asynchronous core with thread pooling for improved scale and performance
- Support for multiple protocols
  - Legacy “Core” protocol
  - Legacy “Openwire” protocol
  - Standard AMQP
  - Standard MQTT
  - STOMP

AMQ7 Interconnect

- All new message router for AMQP
- Separates message routing from message storage
- Integrates clients and brokers in flexible, scalable networks
- Provides direct/brokerless message delivery
- Provides security
- Leverages the extensive capabilities of the AMQP protocol

AMQ7 Clients

- Standard JMS2
- Reactive AMQP clients for better integration
  - C/C++
  - Python
  - Javascript (browser and Node.js)
  - .NET
- Legacy clients
  - AMQ6 (ActiveMQ5)
  - HornetQ
  - MRG-M

A Word about Performance

Broker Performance

- SpecJMS - Transaction rate, Durable message rates, filtering, etc.

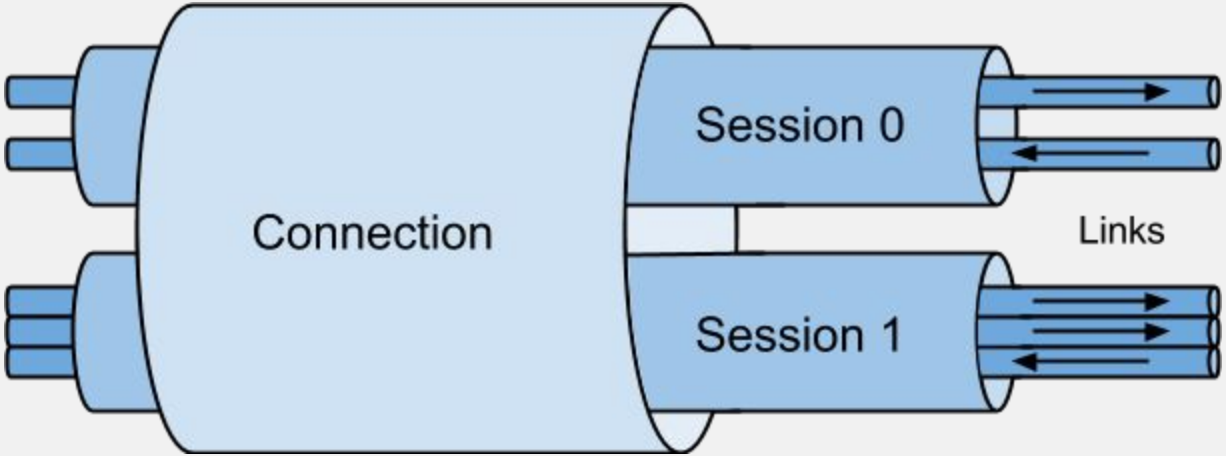
Router Performance

- Raw latency and throughput

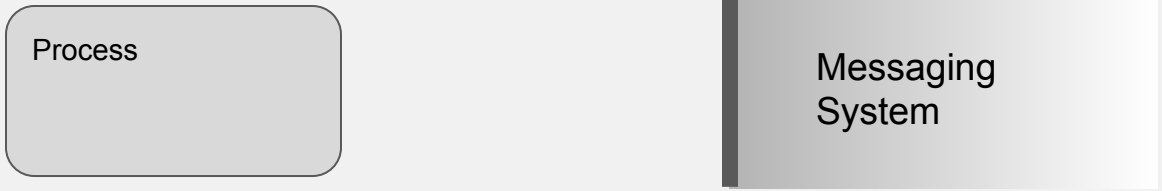
AMQ7 is Next-Generation Messaging for Enterprise, Cloud, and IoT

Diving Deeper

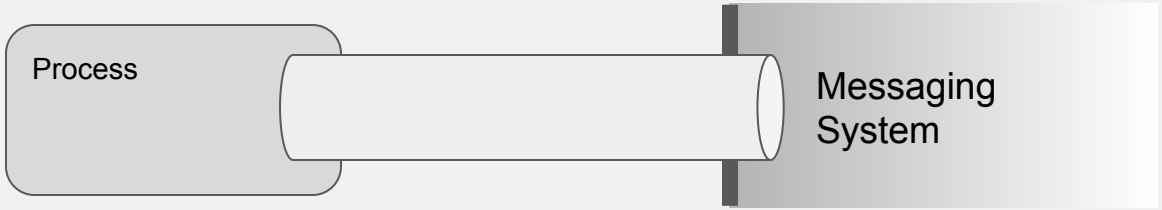
AMQP Anatomy



Message Producer

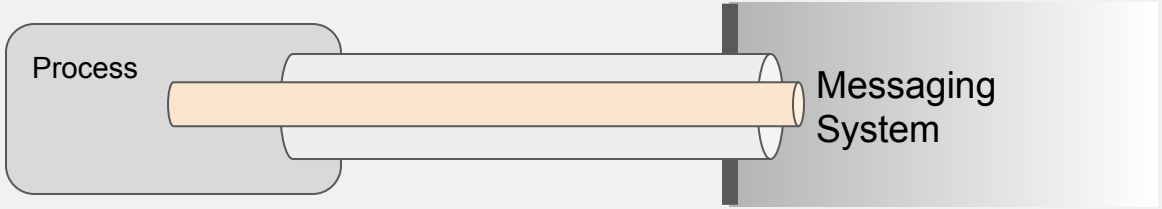


Message Producer



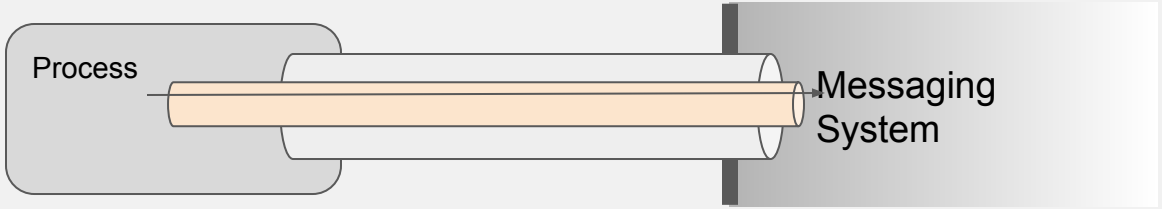

```
on_start():
    conn = container.connect(hostname)
```

Message Producer



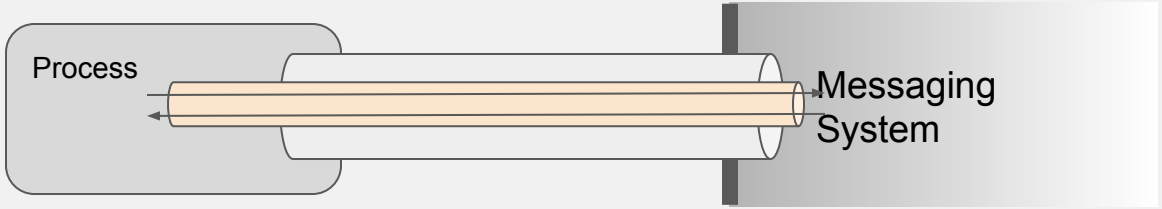

```
on_start():
    conn = container.connect(hostname)
    sender = container.create_sender(conn, “Service”)
```

Message Producer



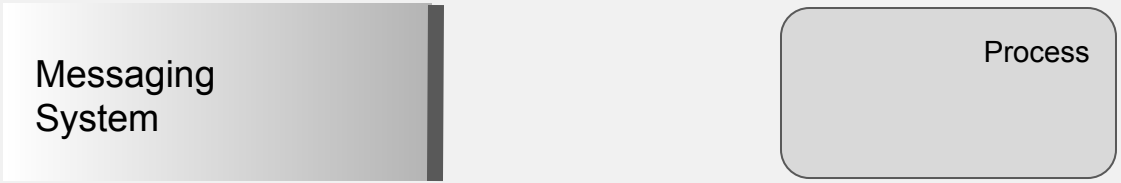

```
on_start():
    conn = container.connect(hostname)
    sender = container.create_sender(conn, “Service”)
on_sendable(event):
    msg = Message(headers, body)
    sender.send(msg)
```

Message Producer

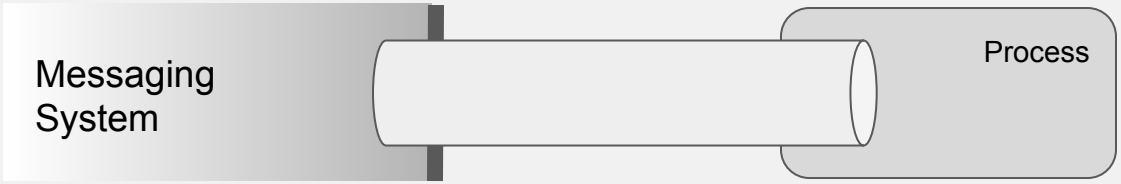



```
on_start():
    conn = container.connect(hostname)
    sender = container.create_sender(conn, “Service”)
on_sendable(event):
    msg = Message(headers, body)
    sender.send(msg)
on_accepted(event):
    # message delivery confirmed
```

Message Consumer

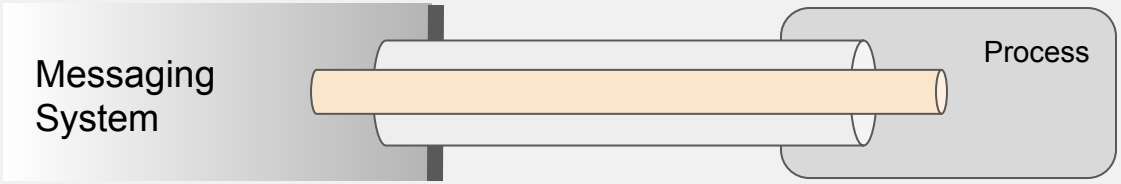


Message Consumer



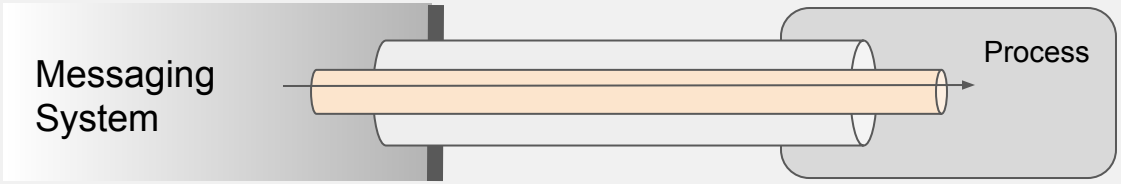

```
on_start():
    conn = container.connect(hostname)
```

Message Consumer



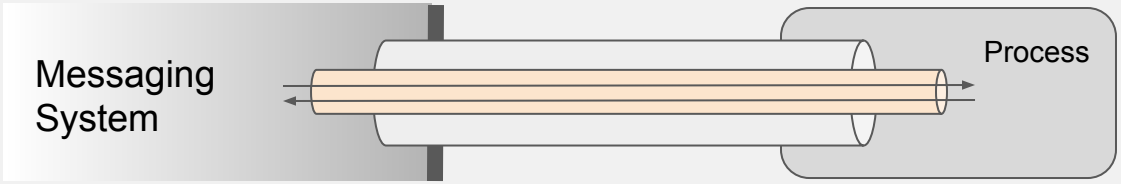

```
on_start():
    conn = container.connect(hostname)
    receiver = container.create_receiver(conn, “Service”)
```

Message Consumer




```
on_start():
    conn = container.connect(hostname)
    receiver = container.create_receiver(conn, “Service”)
on_message(event):
    Process(event.message)
```

Message Consumer

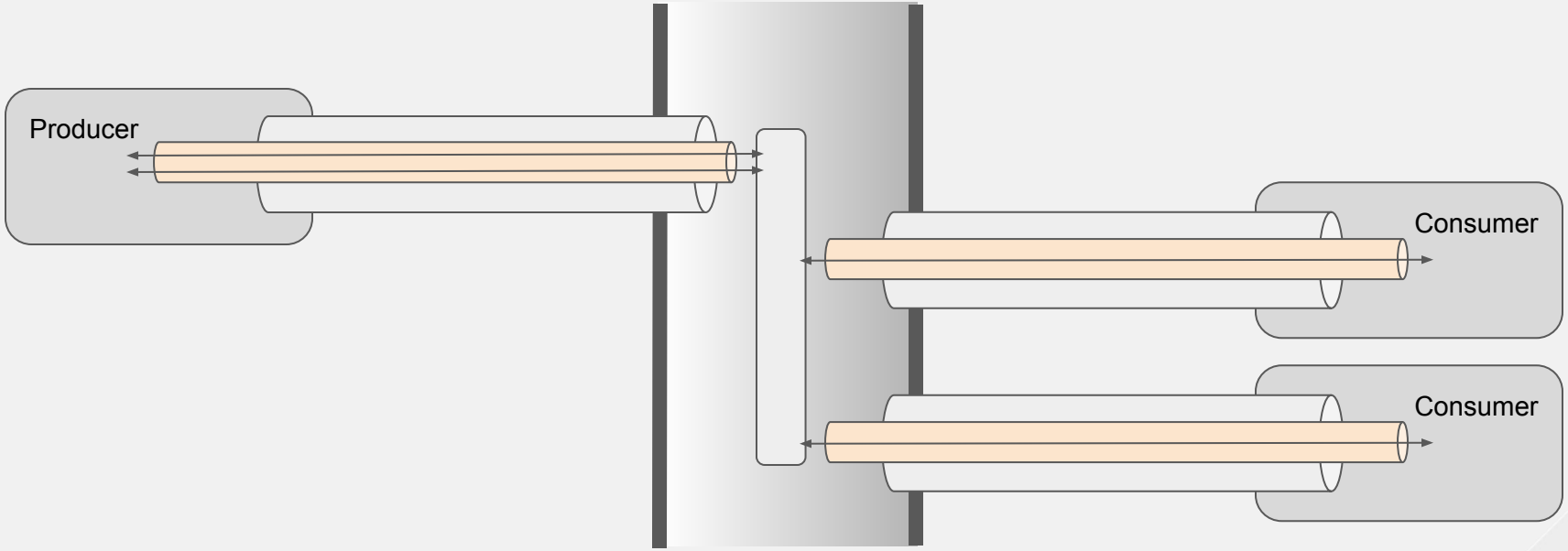



```
on_start():
    conn = container.connect(hostname)
    receiver = container.create_receiver(conn, “Service”)
on_message(event):
    Process(event.message)
    container.accept(event.delivery)
```

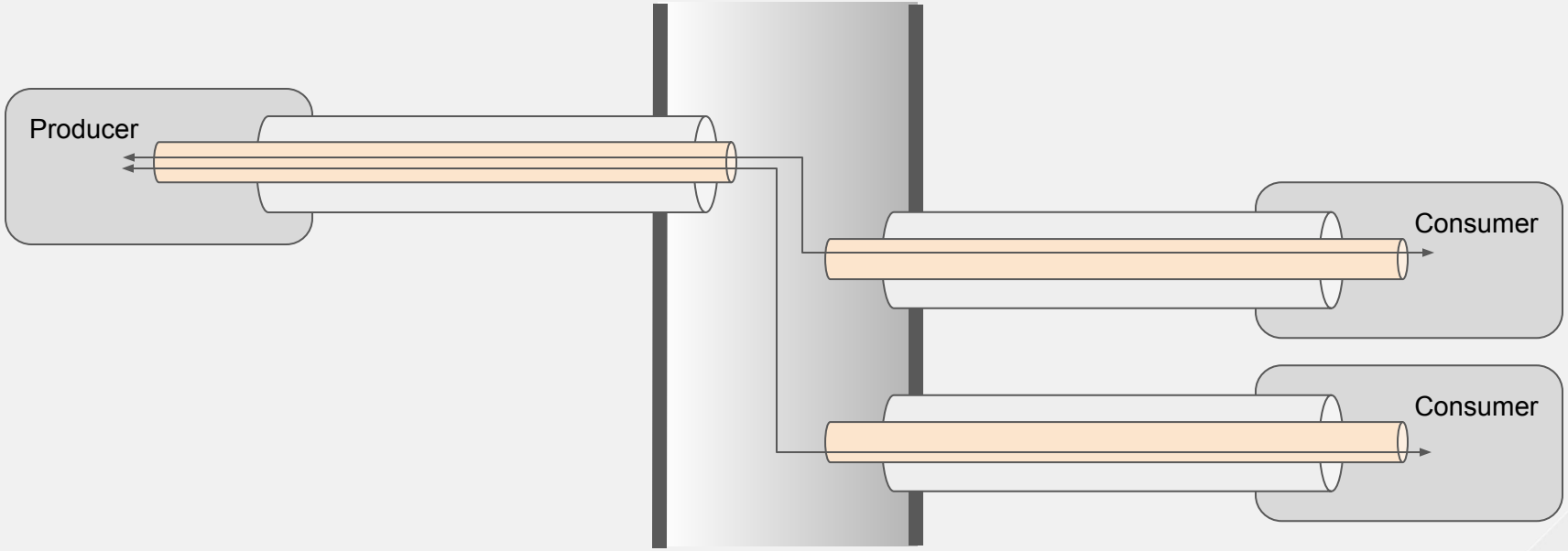
AMQP Protocol Features

- Full-Duplex and Asynchronous
- Message encoding: Body and Headers/Annotations
- Settlement and Disposition
  - Settlement: Best Effort; At-Least-Once; Exactly-Once
  - Disposition: Accepted, Rejected, Released
- Flow Control
  - Message Credit
  - Session Frames
- Multiplexing
- Addressing

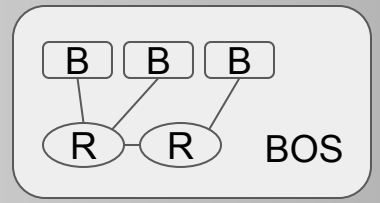
Brokered Messaging



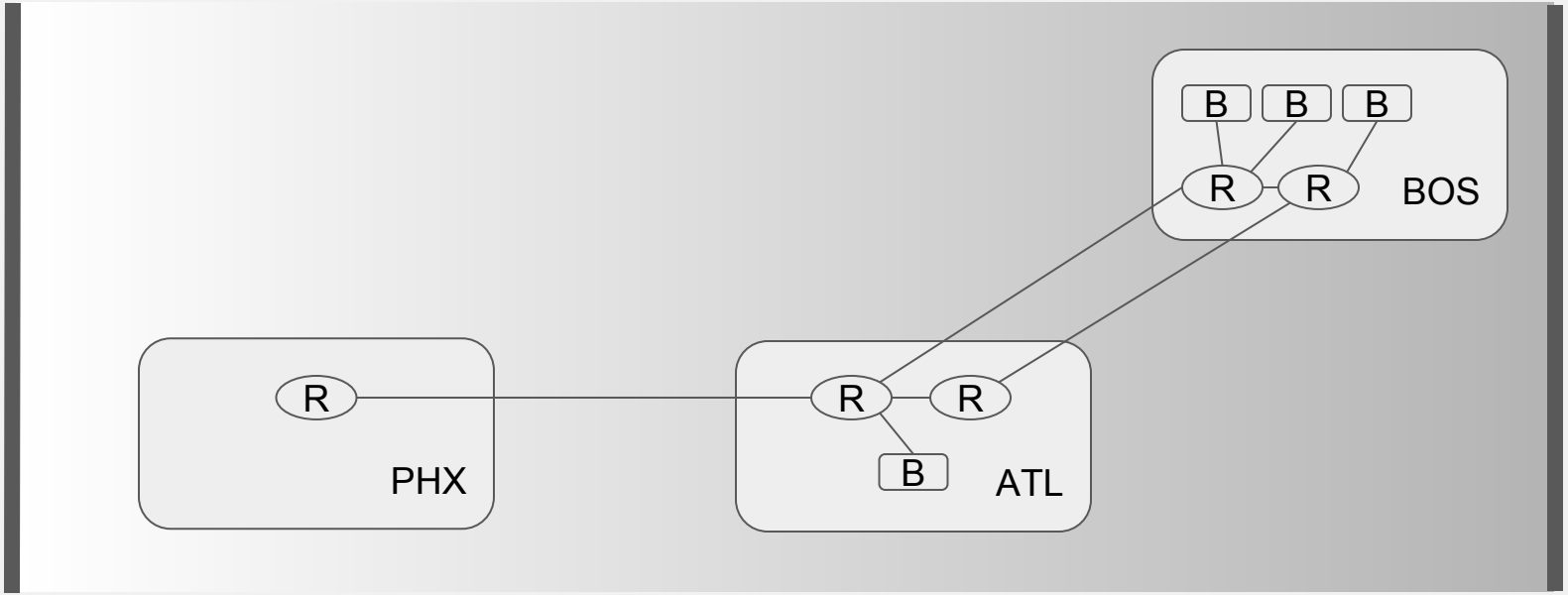
Non-Brokered Messaging



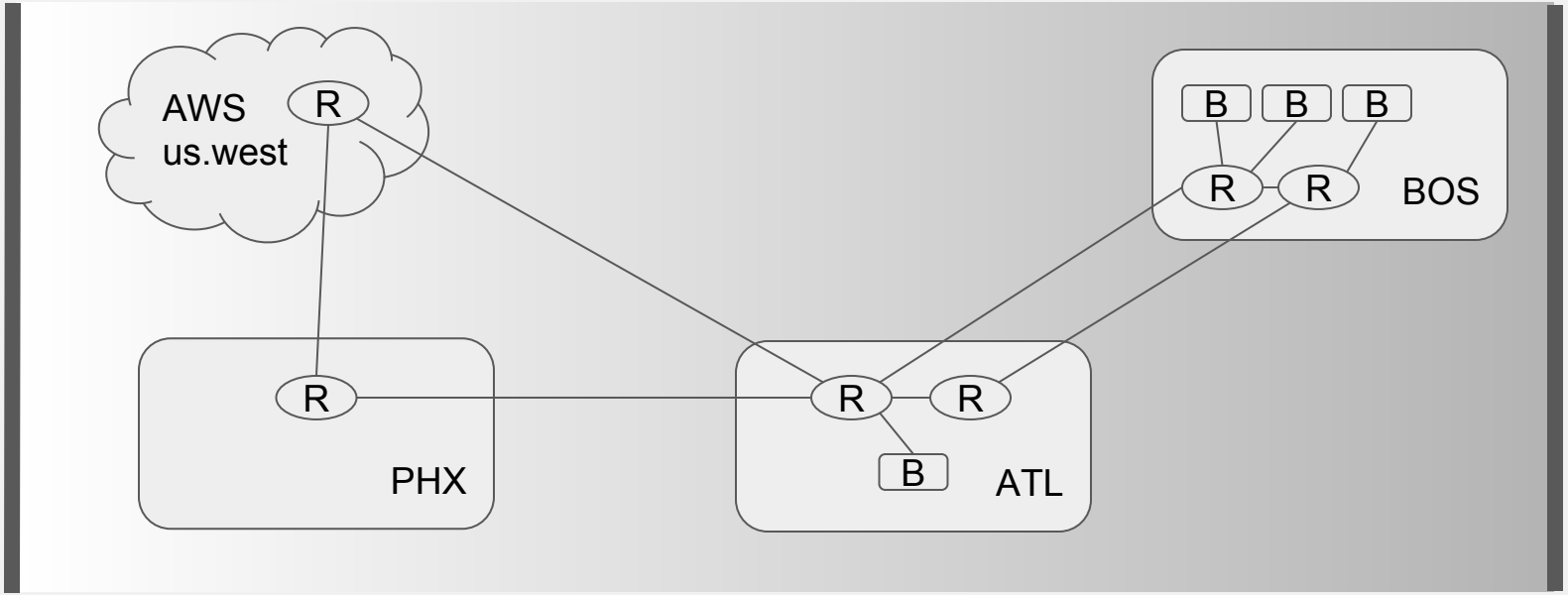
Scaling Out



Scaling Out

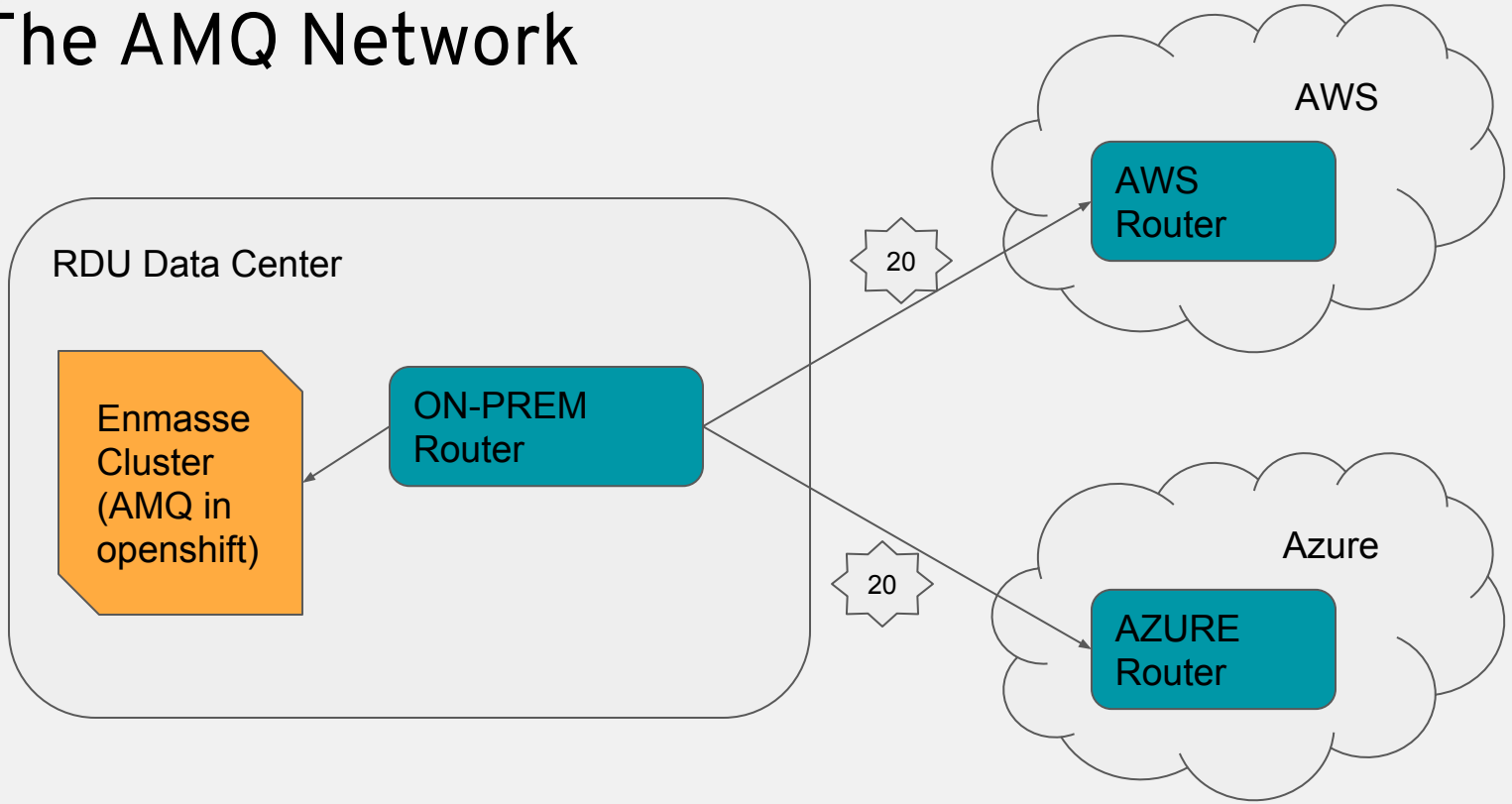


Scaling Out



Hybrid Cloud Demonstration

The AMQ Network



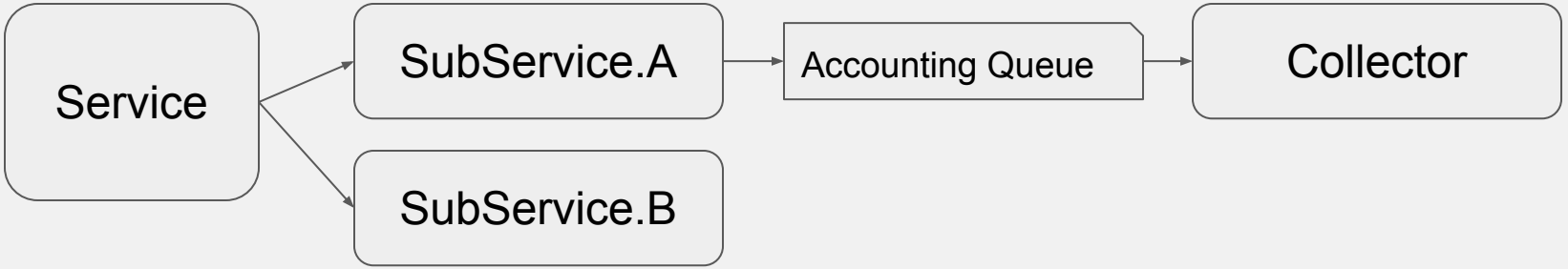
Security Configuration

ON-PREM, AWS, and AZURE routers mutually authenticate using a dedicated x.509 Certificate Authority

Connection roles are explicit. Inter-router connections are separate from normal (client access) connections.

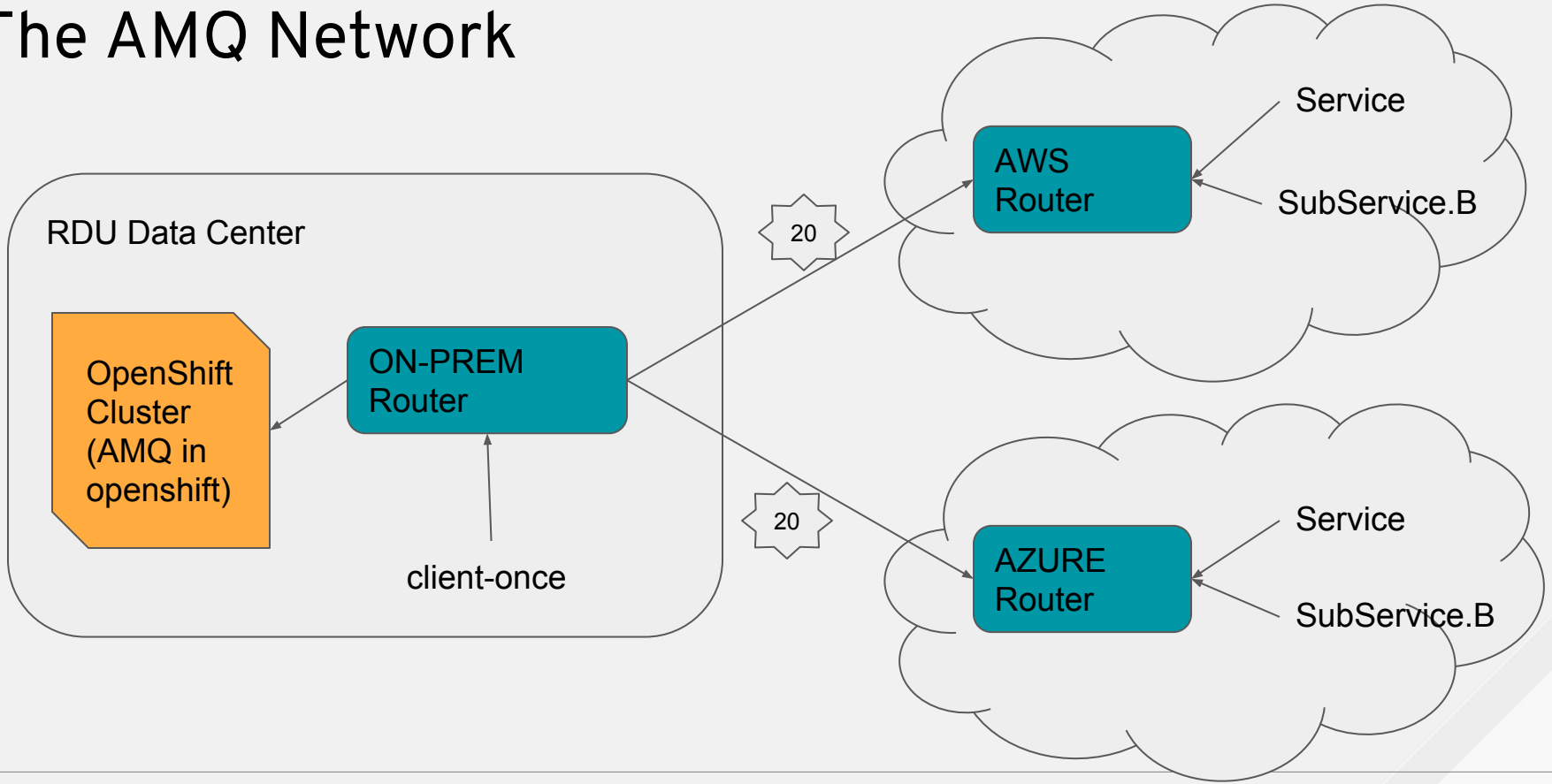
Client access to the cloud routers is not exposed outside the cloud provider.

The Application



- Service is an internal Enterprise application service
- All services are hosted inside the enterprise (in openshift)
- Service and SubService.B are also hosted in the public cloud for overflow
- SubService.A uses sensitive data and is not deployed outside the enterprise

The AMQ Network



RED HAT SUMMIT

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos

LEARN. NETWORK.
EXPERIENCE
OPEN SOURCE.