Connect IoT with RSockt

Andy Shi  Alibaba cloud
andy.shi@Alibaba-inc.com
Twitter: @andyshi
RSocket: Application protocol providing Reactive Streams semantics

- Binary Message-based protocol
- Four interaction models
- Bi-directional
- Multiplexed / Connection Oriented
Demo Setup
Why is there a broker?
Code Snippet

return Mono.<RSocket>just(handler)

...  
    .doOnSuccess(rSocket -> {  
        //do something when connection is established  
    })  
    .doOnTerminate(() -> {  
        // do something when connection is terminated  
    });
Connection oriented

- Establishes long connections
- Setup frame carries metadata
- Connection subscribes to events
Benefit of connection oriented

- No 3rd party registry
- No 3rd party health check
- Session resumption
Device connects to broker

RSocketFactory.connect()

.setupPayload(DefaultPayload.create(JsonUtil.serialize(setup Data)))
    ...

.transport(TcpClientTransport.create("tcp://47.90.241.121:80"))
Bi-directional

- Security
- IP management
- Connection management
The design of RSocke

- Binary Message-based protocol
- Four interaction models
- Bi-directional
- Multiplexed / Connection Oriented
Four interaction models

- Request-response
- Fire and forget
- Request-stream
- Channel
Reactive Streams

- Asynchronous
- Non-blocking
- Back pressure
RSocket flow control
Difference from gRPC

- gRPC has protobuf and HTTP/2
- RSocket supports different encoding scheme, including protobuf
RSocket vs. HTTP/2

- Flow control end to end vs. hop to hop
- More communication models
Difference from MQTT (and Kafka)
The development of RSocke

- Starts from spec
- \url{http://rsocket.io/}
- Polyglot support
- Becoming a member of Linux Foundation
Thank You!