Recent Enhancements to Cloud Foundry Routing

Route Services and TCP Routing
Shannon Coen
Pivotal
@shalako
The CF Routing Team

Chris Pivotal
Leo GE
Mark IBM
Shash Pivotal
Edwin Pivotal

Alumni:
Atul GE
Fermin GE
Iryna Pivotal
Mike Pivotal
Yu EMC
Matt Pivotal
Ergin Pivotal
Mark Pivotal
CF Routing - What do we do?

HTTP Routing

- Dynamically configured
- Round-robin load balancing
- SSL Termination
- Websockets
- Sticky sessions
- Transparent retries
Route Services
The Opportunity

Developers are on their own for pre-processing of application requests

- Authentication
- Rate-limiting
- API Management
- Traffic analysis

Delivering these solutions customized for each app is a burden
Our Solution

- Route Services are a new class of marketplace service
- Enabling developers to inject services into their application request data path
- Developer self-service ==> Minimize time to market
User Experience

- Same discovery, same access controls
  $ cf marketplace

- Same provisioning workflow
  $ cf cs SERVICE PLAN INSTANCE -c PARAMETERS

- User-provided service instances supported
  $ cf cups INSTANCE -r ROUTE_SERVICE_URL

- Bind app route to a service instance
  $ cf brs DOMAIN --hostname HOSTNAME INSTANCE -c PARAMETERS

Thank you, CLI Team!
How Does It Work?

Management plane is the same

- Service broker enables provisioning, discovery, configuration
- Service broker may optionally return a Route Service URL
Route Services - How do they work?

Data path when service broker returns URL for Route Service

- Requests forwarded to Route Services are encrypted via TLS
Route Services - How do they work?

Broker doesn't have to return a URL - Route Service may already be in the data path

- Service broker enables developer self-service configuration via CF Marketplace
Your Imagination is Inspiring

- App auto sleep - Guillaume Berche, Orange (today at 4:15)
- SSL certificate provisioning - Guido Westenberg, Pivotal
- Weighted routing and automated cutover
"Here is my source code
Run it on the cloud for me
I do not care how"
”I do care, a bit
How other people use it
Please make that easy”
ApiGee Edge - http://dev.apigee.com/

- API Platform built for the digital value chain
- Analytics, Traffic Management, Security, Mediation, Extension, and so on.
- Developer Experience, both Consumer and API.
Apigee Edge Service Broker

➔ Node.js Source: [https://github.com/apigee/pivotal-cf-apigee](https://github.com/apigee/pivotal-cf-apigee)
➔ Developer Experience
  ◆ **cf create-service**
    • Broker validates credentials/access to Edge
  ◆ **cf bind-route-service**
    • Broker creates proxy in Edge environment
    • Broker returns proxy endpoint url
  ◆ Route Services handles the rest
Live demo

➔ Create service instance
➔ Bind route
➔ Quick API bits
Creating a Service Broker - Developer Experience

➔ It’s just APIs! \(_{(ツ)_/\)\

➔ But it’s easy to make them hard to use...
  ◆ [https://twitter.com/anthonypjshaw/status/726902680623308801](https://twitter.com/anthonypjshaw/status/726902680623308801)
  ◆ Documentation
  ◆ Security
  ◆ Sensible use of HTTP (URIs, Methods, so forth)

➔ Testing/mocks
➔ Local Dev environment
Deeper dive with Apigee:
Using Route Services to Manage APIs
Tomorrow 10:55am

More info:
docs.cloudfoundry.org/services
TCP Routing
The Opportunity

- So much software depends on non-HTTP protocols
- Cloud Foundry is a great place to develop and operate these applications also!
- Developer self-service ==> Minimize time to market
Introducing TCP Routing

TCP routing significantly expands the range of workloads that can be run on CF

- Internet of Things (e.g. MQTT, AMQP)
- Legacy workloads
- Non-persistent TCP services

Enables new use cases for existing workloads

- Terminate TLS at your app
User Experience

- Developer discovers a domain that supports TCP routing

$ cf domains
Getting domains in org scoen as scoen...
name                status  type            
cf.example.com      shared
TCP1.cf.example.com shared  tcp

- Push your app and specify the TCP domain

$ cf push myapp -d tcp1.cf.example.com --random-route
Binding tcp.cf.example.com:60010 to myapp...
OK
TCP Routing - How does it work?

Management plane

- CLI / GUI
- CF API
- Diego
- Routing API
- Emitter
- TCP Router

New!
How does it work?

TCP Router makes routing decision based on port request is received on

tcp1.cf.example.com:60089
10.0.112.10:60089
10.0.12.10:60089
10.0.48.66:60212
Protocol landscape - GE
TCP Routing use cases - GE

- MQTT Adapter for Kafka
  - Ingest telemetry data from devices over MQTT
  - Ingestion at very high throughput/scale
  - MQTT Adapter as CF app

- Proprietary protocol support
  - Some GE businesses need to use proprietary protocols to transfer data to cloud
  - Use TCP routing to deploy CF apps that can directly listen for non http traffic
  - Reduces operational overhead as CF manages lifecycle and health of the app
TCP Router Demo

http://app1

Smart phone

matt publish <tcpdomain>:<port>

matt subscribe

TCP router

matt

MQTT Broker

http

App1

CLOUD FOUNDRY
What's next for CF Routing?

- Performance benchmarking and improvements
- Replace NATS with Routing API
- Bring your own Router
- Weighted Routing
Get in touch:
slack.cloudfoundry.org #routing

Summit Office Hours:
Today 4:15pm in The Foundry - Lounge #2

More info:
docs.cloudfoundry.org/services
github.com/cloudfoundry-incubator/routing-release
Router Groups

- Horizontally scalable group of identically configured routers
- Each router group is configured independently, enabling port reuse
- Only the TCP router group is supported now, but many in the future
- Router Groups have a type that describes their capabilities, e.g. "tcp"

A router group is associated with one or more domains

- Developers work with Domains and Routes; abstracted from Router Groups
- Developers choose a domain to create a route from that offers the capabilities they need
User Experience

- Or, add a TCP route to your app later
  
  ```
  $ cf map-route myapp tcp1.cf.example.com --random-port
  ```

- Optionally request a specific port; it may already be reserved by another route
  
  ```
  $ cf map-route myapp tcp1.cf.example.com --port 60089
  FAILED
  The port is taken: Port 60089 is not available
  ```
Admin Preparation

1. Configure a load balancer to forward a range of ports to the TCP routers
2. Configure the Routing Service with the same port range for reservation
3. Configure DNS to point a new domain at the load balancer
4. Add the domain to CF
5. (Optional) Configure quotas to control how many ports can be reserved by orgs and spaces