Proposal UX for Routes in App Manifest

Motivation

When the application manifest specification was designed, apps on CF had only a single route, consisting of a host and a domain. The corresponding attributes in the manifest were overloaded with “hosts” and “domains” to support multiple routes to an app.

This can already cause confusing situations such as the following (from github issue 765):

```
---
  ...
  hosts:
  - app_host1
  - app_host2
  domains:
  - domain-example1.com
  - domain-example2.org
```

Does this mean four routes for this application?

1. “app_host1.domain-example1.com”
2. “app_host2.domain-example1.com”
3. “app_host1.domain-example1.org”
4. “app_host2.domain-example1.org”

What if we only wanted routes 1 and 4?

With recent new routing features such as Routes with Path, TCP routing and mapping to Multiple App Ports, adding similar attributes for each would lead to a multitude of combinations.

This document is a proposal for a new UX to address this.

Refer to http://docs.cloudfoundry.org/devguide/deploy-apps/manifest.html for the current app manifest specification.

Goals

1. Allow a user to express the routes they want in the app manifest, without creating additional unintended routes.
2. Allow the `cf create-app-manifest` command to generate an app manifest from a deployed application that uses the various route components, and which represents the mapped routes correctly.
3. Ensure the compatibility with the current app manifest specification: existing app manifests will remain supported.

Non-goals

1. Use the same options in app manifests and CLI commands that specify routes and route mappings.

Fundamentals

Route Composition

In Cloud Foundry, an HTTP route is composed as follows:

app1.example.com/path	
app1	Hostname; optional
example.com	Domain
path	Route path; optional

A TCP route is composed as follows:

example.com:1234	
example.com	Domain
1234	Route port

Route Example	Route Type	App
app1.example.com/path	HTTP	app1
app1.example.com/some-other-path	HTTP	app1
example.com:1234	TCP	app1
example.com:1235	TCP	app1

Proposed Solutions

Two different solutions are proposed and proposal B is currently favored.

Proposal A: Move route definitions into a new section under the application;
Proposal B: Move route definitions into a new section **and** specify routes in their entirety, not split up into their components.

The major differences between the two proposed and the current way of expressing routes are as follows:

- With both proposals, each route will be declared in a YAML mapping, allowing additional attributes to be defined for each route (e.g. the app port the route is mapped to).
- With proposal A, each route will be specified in the same familiar way as with the `cf create-route` and `cf push` commands: for each route the host, domain, port and/or path is specified separately.
  With proposal B, each route will be specified in the same way a client would access it, as a full route, making it easier to recognize and copy & paste into a browser, curl command or application client.

Examples

Before explaining the proposed new syntax, first some examples:

The first example shows a full app manifest for an application named “myapp” with two HTTP routes mapped to it. It is the equivalent to executing the following two commands:

```
$ cf push myapp --hostname www -d example.com --route-path foo
$ cf map-route myapp example.com --hostname my --path /bar
```

HTTP route example with proposal A:

```
---
applications:
- name: myapp
  routes:
  - host: www
    domain: example.com
    path: /foo
  - host: my
    domain: example.com
    path: /bar
```

HTTP route example with proposal B:

```
---
applications:
- name: myapp
  routes:
  - route: www.example.com/foo
  - route: my.example.com/bar
```

The next example shows a full app manifest for an application named “myapp” with an HTTP route and a TCP route. (This example makes more sense once multiple application ports are supported.)

```
$ cf push myapp -d example.com --route-path /foo
$ cf map-route myapp tcp-example.com --port 1234
```

HTTP & TCP route example with proposal A:

```
---
applications:
- name: myapp
  routes:
  - domain: example.com
    no-hostname: true
    path: /foo
  - domain: tcp-example.com
    port: 1234
```

HTTP & TCP route example with proposal B:

```
---
applications:
- name: myapp
  routes:
  - route: example.com/foo
  - route: tcp-example.com:1234
```

Syntax

Routes are defined under a new “routes” mapping in the application definition.
HTTP and TCP routes can be declared together. The order in which routes are defined has no significance.

Route declaration syntax with proposal A:

```
---
applications:
- name: my-app
  routes:
  - domain: HTTP_DOMAIN
    host: HOSTNAME
    path: PATH
    no-hostname: true | false
  - domain: TCP_DOMAIN
    port: PORT | random
  - …
```

An empty “routes” mapping is equivalent to setting the current “no-route” attribute.
When attributes are omitted, the following defaults apply:

Attribute	**Default value**
domain	Default shared domain
host	APP_NAME (unless no-hostname: true)
path	/
no-hostname	false

Route declaration syntax with proposal B:

```
---
applications:
- name: my-app
  routes:
  - route: HOSTNAME.HTTP_DOMAIN/PATH
  - route: TCP_DOMAIN | TCP_DOMAIN:PORT
  - …
```

A “routes” attribute with no mappings is ignored (i.e., it does not imply “no-routes: true”). When no port number is specified in a TCP route, an available port will be allocated and assigned.

If a manifest with a routes declaration is used to push an existing app that has routes mapped already that are not defined in the manifest file, these routes are unmapped (but not deleted).

Interaction of New Manifest Attributes with `cf push` Options

The current specification allows the user to override most app manifest attributes by `cf push` options. This is useful when testing an app manifest that was created for another environment (with e.g. domains or port numbers not available in the test environment) without having to modify the original app manifest or create a new one.

This use case will remain to be supported with the new attributes.
The following table describes the resulting behaviour for each of the applicable `cf push` options and manifest attributes.

`cf push` options interaction with attributes of proposal A:

`cf push` Option	**Manifest Attribute**	**Resulting Behaviour**
--no-route	any	All declared routes are ignored.
-d	domain	Option overrides “domain” attribute in all routes.
--hostname, -n	host, no-hostname	Option implies “no-hostname” false and overrides “host” attribute in all HTTP routes. It has no impact on TCP routes.
--no-hostname	no-hostname	Option overrides “no-hostname” attribute in all HTTP routes. It has no impact on TCP routes.
--route-path	path	Option overrides “path” attribute in all HTTP routes. It has no impact on TCP routes.
--random-route	host, port	Option implies “no-hostname” false and a random value for the “host” attribute in all HTTP routes. Option implies “port” to be “random” in all TCP routes.
--route-port	port	Option overrides “port” attribute in all declared TCP routes. It has no impact on HTTP routes.

`cf push` options interaction with attributes of proposal B:

`cf push` Option	**Manifest Attribute**	**Resulting Behaviour**
--no-route	any	All declared routes are ignored.
-d	route	Option overrides DOMAIN part of all declared HTTP and TCP routes.
--hostname, -n	route	Option sets or overrides HOSTNAME part in all HTTP routes. It has no impact on TCP routes.
--no-hostname	any	Option sets or overrides “no-hostname”, which is not allowed with the new attributes, so results in an error. (The option’s purpose is to suppress creation of a default hostname, which doesn’t happen with the new attributes.)
--route-path	route	Option sets or overrides the PATH part in all HTTP routes. It has no impact on TCP routes.
--random-route	route	Option sets or overrides the HOSTNAME part in all HTTP routes with a random name. one.
--route-port	url	Option sets or replaces the PORT part of TCP routes. It has no impact on HTTP routes.

Compatibility With Current Manifests

App manifests with the current route attributes remain to be supported.
However, to keep things simple, usage of current application level “domain”, “domains”, “host”, “hosts” and “no-hostname” attributes is not allowed together with the new “routes” attribute in the same app manifest file.

`cf create-app-manifest` will only use the new attributes in generated manifests.