Title: SIngle jar startup failed after upgrade from Spring Boot 1.3.7 to 1.4.0

Post Body:

After upgrading from Spring Boot 1.3.7 to 1.4.0 we can no longer start our application as a single jar build with the Spring Boot Maven plugin. Our application is a small REST interface using Jersey and Jetty. We use Maven and our pom file is pretty standard Spring Boot.

We can still run the application using `mvn spring-boot:run` and from within Eclipse, but when run as a single jar Jersey `ResourceFinder` complains that it cannot find `.jar!/BOOT-INF/classes`.

When I unpack the jar the folder `BOOT-INF/classes` is present and contains the expected classes and resources.

Any help appreciated.

```
2016-08-10 14:58:31.162 ERROR 16071 --- [           main] o.s.boot.SpringApplication               : Application startup fail
```

Accepted Answer:

From the Spring Boot 1.4 release notes:

> The change to the layout of executable jars means that a limitation in Jersey's classpath scanning now affects executable jar files as well as executable war files. To work around the problem, classes that you wish to be scanned by Jersey should be packaged in a jar and included as a dependency in `BOOT-INF/lib`. The Spring Boot launcher should then be configured to unpack those jars on start up so that Jersey can scan their contents.

Highest Rated Answer:

Just another solution:

Although Jersey cannot scan your classes inside the new version of the fat boot jar, you can achieve the same effect using Spring classpath scanning facilities. This way you can scan a package similarly to `ResourceConfig.packages()`:

```
ClassPathScanningCandidateComponentProvider scanner = new ClassPathScanningCandidateComponentProvider(false); scanner.addInclu
```

Note: please have a look at the source of `org.glassfish.jersey.server.internal.scanning.AnnotationAcceptingListener`. This is the stock solution and you can see that it does the same: it scans for classes annotated with `@Path` or `@Provider` (but doesn't manage to find anything because of the broken scanning mechanism).

By the way the bean based method posted by lanwen may be clearer :) Just add `@Provider` to that too.