Open Tracing Support

OpenTracing is an open standard for Tracing in-process and out-of-process distributed transactions. At a high level, it performs shallow stitching of method executions involved in a business flow, and hence enables measuring it end to end.

AppDynamics Tracer is an implementation based on version 0.31.0 of the OpenTracing standards.

Prerequisite

For the OpenTracer to work correctly, use the Executor strategy in the Java agent. You can specify it in app-agent-config.xml or as node property:

```
<!--As part of app-agent.config.xml-->
<property name="async-instrumentation-strategy" value="executor"/>

Or, just specify the following node property in the Controller:

async-instrumentation-strategy = "executor"
```

Getting The AppDynamics OpenTracer

Installing the Dependency

The OpenTracer jar can be accessed directly, or downloaded from Maven Central, or it can be downloaded from the AppDynamics portal. The library version changes with each new OpenTracer release, and is not tightly coupled to the version of the underlying agent, which must be a minimum of version 4.5.13.

```
dependencies {
  compile group: 'com.appdynamics.agent', name: 'opentracer', version: '4.5.13.27526'
}
```
Enabling or Disabling the Tracer

By default when plugged in, OpenTracer is enabled. There are two ways to disable it:

- Programmatically you can enable and/or disable using:

```
AppdynamicsTracerConfiguration.appdTracingConfiguration().setTracingEnabled();
```

- Optionally, at startup, you can specify the system property to enable and/or disable using:

```
-Dappdynamics.opentracing.enabled=false
```

Plugging the Tracer into Lightbend Telemetry

The first OpenTracing use case validated by AppDynamics is its use with Lightbend telemetry (also known as the Cinnamon agent), which allows transaction tracing through applications built on the Lightbend reactive platform, for example those built on Akka HTTP. For more details, refer to the Lightbend documentation, in particular as it relates to OpenTracing integration.

Full instructions as to how to set up Lightbend Telemetry are beyond the scope of the AppDynamics documentation. At a high level, the necessary steps for configuration are:
1. Configure the Lightbend telemetry agent
   For Lightbend instructions, see Instructions.

2. Enable tracing for Akka HTTP endpoints you wish the AppDynamics agent to trace
   For Lightbend instructions, see OpenTracing Configuration.

3. Plug the AppDynamics OpenTracing implementation into Lightbend Telemetry as a custom OpenTracing tracer using
   the AppdynamicsTracerFactory as described in the Lightbend documentation. A sample AppDynamics tracer factory is provided below:

   ```java
   import com.appdynamics.opentracing.core.AppdynamicsTracerFactory;
   import com.lightbend.cinnamon.opentracing.TracerFactory;
   import io.opentracing.Tracer;
   public class AppdynamicsTracerPlugin implements TracerFactory {
   @Override
   public Tracer create() {
       return AppdynamicsTracerFactory.getTracer();
   }
   }
   ```

   Ensure the active sampler is set to `const-sampler` in your `cinnamon.opentracing` file. (The low-overhead nature of
   the AppDynamics Java agent means that you can safely ignore the comment in the sample configuration file that states this
   should only be used for non-production).