

HBase 工程中 protobuf 版本冲突解决

Protobuf（全称 Protocol Buffers）是 Google 开发的一种数据描述语言，能够将结构化数据序列化，可用于数据存储、通信协议等方面。在 HBase 里面用使用了 Protobuf 的类库，目前 Protobuf 最新版本是 3.6.1（参见[这里](#)），但是在目前最新的 HBase 3.0.0-SNAPSHOT 对 Protobuf 的依赖仍然是 2.5.0（参见 `protobuf.version`），但是这些版本的 Protobuf 是互补兼容的！

如果我们的业务系统里面既用到了 HBase，又用到了比较新的 Protobuf（比如 3.0.0），这时候我们的项目是无法运行的。因为项目中包含了多个版本的 Protobuf，保留最新的 Protobuf 或者最旧的 Protobuf 都会导致项目无法运行。那么遇到这种情况有什么好的办法来解决呢？本文将介绍两种办法来处理这个问题。

将客户端的 Protobuf 类库进行重命名

这里用到了 Maven 的一款工具 `maven-shade-plugin`，其可以在执行 `mvn package` 的时候将依赖全部打到一个 jar 包里面，同时我们还可以对依赖的包名进行重命名（Relocating Classes）。这个方法就是使用这个方法将 Protobuf 包名进行重命名，具体如下：下面的 `pom.xml` 文件打包的 jar 无法运行：

```
<dependencies>
  <dependency>
    <groupId>org.apache.hbase</groupId>
    <artifactId>hbase-client</artifactId>
    <version>1.4.8</version>
  </dependency>

  <dependency>
    <groupId>com.google.protobuf</groupId>
    <artifactId>protobuf-java</artifactId>
    <version>3.0.0</version>
  </dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-shade-plugin</artifactId>
      <version>3.0.0</version>
      <executions>
        <execution>
          <phase>package</phase>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
```

```

        <goals>
            <goal>shade</goal>
        </goals>
    </execution>
</executions>
</plugin>
</plugins>
</build>

```

我们将上面的 pom.xml 修改成下面的：

```

<dependencies>
    <dependency>
        <groupId>org.apache.hbase</groupId>
        <artifactId>hbase-client</artifactId>
        <version>1.4.8</version>
        <scope>provided</scope>
    </dependency>

    <dependency>
        <groupId>com.google.protobuf</groupId>
        <artifactId>protobuf-java</artifactId>
        <version>3.0.0</version>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-shade-plugin</artifactId>
            <version>3.0.0</version>
            <executions>
                <execution>
                    <phase>package</phase>
                    <goals>
                        <goal>shade</goal>
                    </goals>
                </execution>
            </executions>
            <configuration>
                <relocations>

```

```
<relocation>
  <pattern>com.google.protobuf</pattern>
  <shadedPattern>com.iteblog.google.protobuf</shadedPattern>
</relocation>
</relocations>
</configuration>
</plugin>
</plugins>
</build>
```

这样我们将 com.google.protobuf 替换为 com.iteblog.google.protobuf 了，可以通过查看打包的文件：

```
$ jar -tf hbase-demo-iteblog.jar | grep iteblog | less
```

输入如下：

```
com/iteblog/google/protobuf/AbstractMessageLite$Builder.class
com/iteblog/google/protobuf/AbstractMessageLite.class
com/iteblog/google/protobuf/AbstractParser.class
com/iteblog/google/protobuf/AbstractProtobufList.class
com/iteblog/google/protobuf/Any$1.class
com/iteblog/google/protobuf/Any$Builder.class
com/iteblog/google/protobuf/Any.class
com/iteblog/google/protobuf/AnyOrBuilder.class
com/iteblog/google/protobuf/AnyProto$1.class
com/iteblog/google/protobuf/AnyProto.class
com/iteblog/google/protobuf/Api$1.class
com/iteblog/google/protobuf/Api$Builder.class
com/iteblog/google/protobuf/Api.class
com/iteblog/google/protobuf/ApiOrBuilder.class
com/iteblog/google/protobuf/ApiProto$1.class
com/iteblog/google/protobuf/ApiProto.class
com/iteblog/google/protobuf/BlockingRpcChannel.class
com/iteblog/google/protobuf/BlockingService.class
```

如果想及时了

解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

然后运行程序的时候可以如下进行：

```
java -cp hbase-demo-iteblog.jar;.Whbase-1.4.8-binWlibW* com.iteblog.Test
```

使用 hbase-shaded-client

上面的方法我们是通过将 Google protobuf 包名进行了重命名。这个问题其实很常见，所以社区有人将 HBase 里面比较常见的依赖包进行了重命名，这个就是 hbase-shaded-client 了（详见 [HBASE-13517](#) 说明），其实他也是利用了 maven-shade-plugin，其中对 Google 的所有依赖（protobuf、Guava等）进行了重命名，如下（完整的参见 [这里](#)）

```
<relocation>
  <pattern>com.google</pattern>
  <shadedPattern>org.apache.hadoop.hbase.shaded.com.google</shadedPattern>
</relocation>
```

这样我们可以在项目里面只用使用 hbase-shaded-client.jar 来替换 hbase-client.jar，如下：

```
<dependencies>
  <dependency>
    <groupId>org.apache.hbase</groupId>
    <artifactId>hbase-shaded-client</artifactId>
    <version>1.4.8</version>
  </dependency>

  <dependency>
    <groupId>com.google.protobuf</groupId>
    <artifactId>protobuf-java</artifactId>
    <version>3.0.0</version>
  </dependency>
</dependencies>

<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-shade-plugin</artifactId>
      <version>3.0.0</version>
      <executions>
        <execution>
          <phase>package</phase>
          <goals>
            <goal>shade</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
```

```
</executions>  
</plugin>  
</plugins>  
</build>
```

然后我们就可以正常运行程序了。本文只是以 HBase 为例说明如何解决 Java 工程中的冲突依赖，大家可以举一反三，比如 Guava 冲突解决，这里就不再介绍了。

本博客文章除特别声明，全部都是原创！
原创文章版权归过往记忆大数据（[过往记忆](#)）所有，未经许可不得转载。
本文链接: [【】（）](#)