使用 flume.avro.schema.hash 的关键词将模式指纹插入到 header 中。

如果整个 JSON 化模式应该写入每个事件，设置 schemaType 参数的值为 flume.avro.schema.literal。在这种情况下，整个模式使用关键 flume.avro.schema.literal 来写。在每一个事件的 header 中写模式是非常低效的，因为它增加了事件的大小，特别是如果存在数量有限的模式类型。 <58>

要从目录中读取文件作为二进制大对象 (BLOB)[blob]，则要使用 blob 反序列化器。blob 反序列化器的 FQCN 是 org.apache.flume.sink.solr.morphline.BlobDeserializer。这类反序列化器有 blob 能接受的最大大小，每个文件会试图以尽可能多的字节从 blob 中文件读取数据。这类反序列化器只有一个配置参数，maxBlobLength，是每个 blob 以字节为单位的最大容量。如果文件比该值大，就要分为几个 blob，每个 blob 的大小应小于或等于配置的最大值。反序列化器在内存中分批缓存所有的 blob，所以批量的大小和 blob 容量的最大值应该要配置，来确保反序列化器不会比预期使用更多内存导致终止。

因为 Spooling Directory Source 是 flume-ng-core 工件的一部分，确保你在序列化器的 pom.xml 文件中添加了 flume-ng-core 的工件，在例 3-6 中提及过。自定义反序列化器可以通过使用第 8 章“部署自定义代码”一节中提到的 *plugins.d* 框架部署到 Flume Agent。

Spooling Directory Source 性能

Spooling Directory Source 是 I/O 密集型的。为了避免复杂的反序列化器实现，Source 被专门设计成单线程的。这意味着有可能通过使用多个线程读取数据、更多地使用可用的 CPU 来提高性能。提高文件读取的性能的一种方法是轮流写文件到不同的目录，并有一个 Spooling Directory Source 处理每一个目录 (如果所有数据传到相同的目的地，则写入到相同的 Channel)。这意味着更多从磁盘读取数据的线程和更多的可以用来反序列化的 CPU。

Syslog Source

Syslog 是很多应用写日志消息的众所周知的格式。集成了 syslog 的 Flume 可以接收 TCP 和 UDP 消息。Flume 提供了两种 syslog Source: Syslog UDP Source 和 Multiport Syslog Source。Syslog UDP Source 用 UDP 接收 syslog 消息，而 Multiport Syslog Source 可以在多个端口用 TCP 接收 syslog 消息。这两种 Source 都能解析 syslog 消息，抽取几个字段到 Flume 事件的 header，可以用 HDF Sink 分桶。如果 syslog 消息不符合 Syslog RFCs，RFC-3164 或 RFC-5424，事件将包含一个带有值为 Invalid flume.syslog.status 的 header。 <59>