只能从 Channel 中取走事件。

Flume 自带的 Channel

Flume 自带两种 Channel：*Memory Channel* 和 *File Channel*。这里介绍的两种 Channel 以相同的基本原理工作。两种 Channel 都是完全线程安全的，并且可以操作多个 Source 和 Sink。顾名思义，Memory Channel 在主存中存储提交的事件，而 File Channel 将事件写出到磁盘上的文件。在本节中我们将讨论这两种 Channel 和在两者之间选择时不同的考虑因素。

Memory Channel

Memory Channel 是内存中的 Channel，在堆上存储写入的事件。实际上，Memory Channel 是内存中的队列——Source 从它的尾部写入，Sink 从它的头部读取。Memory Channel 支持很高的吞吐量，因为它在内存中保存所有的数据。如前面所说，Channel 是线程安全的，可以同时处理几个 Source 的写入操作和几个 Sink 的读取操作。Memory Channel 在不需要关心数据丢失的情景下适用，因为该类 Channel 没有将数据持久化到磁盘。如果需要关心数据丢失，那么 Memory Channel 就不应该使用，因为程序死亡、机器宕机或重启都会导致数据丢失。

83

Memory Channel 支持 Flume 的事务性模型，并为每个程序中的事务维护单独的队列。一旦一个 source-side 事务提交，该事务队列中的事件就被自动移入到 Channel 的主队列中。如果提交完全成功，事务中的事件对 Sink 是可用的。如果失败了，Source 将回滚事务且事件被忽略。对于 sink-side 事务，每当 Sink 完成一个读操作事件就会被移入到事务的队列中。这保证 Sink 正确地读取了事件。当 Sink 提交事务时，事务队列被忽略，事件被废弃，等待垃圾回收。因此，Sink 的实现必须谨慎，只有当事件被成功写入目的地时，Sink 才能提交事务。

如果事务失败，事件将以相反的顺序被重新插入到 Channel 的头部，所以事件将以相同的顺序被再次读取，就像它们当初被插入时一样。用这种方法，尽管 Flume 不能保证顺序性，但是 Memory Channel 能保证事件以它们被写入的顺序进行读取。然而，当某些事务回滚，后写入的事件有可能更早写出到目的地（因为另一个 Sink 可能已经提交包含事件的事务，这些事件比回滚事务中的事件“更新”）。

Memory Channel 不需要太多精力就可以配置好，它是最容易配置的 Flume 组件之一。表 4-1 列出了 Channel 的配置参数。