[返回主页](http://www.cnblogs.com/charlesblc/)

[**笨鸟居士的博客**](http://www.cnblogs.com/charlesblc/)

* [博客园](http://www.cnblogs.com/)
* [首页](http://www.cnblogs.com/charlesblc/)
* [新随笔](https://i.cnblogs.com/EditPosts.aspx?opt=1)
* [联系](https://msg.cnblogs.com/send/blcblc)
* [管理](https://i.cnblogs.com/)
* [订阅](http://www.cnblogs.com/charlesblc/rss) [订阅](http://www.cnblogs.com/charlesblc/rss)

随笔- 1548  文章- 0  评论- 16

[**两军问题与拜占庭将军问题**](http://www.cnblogs.com/charlesblc/p/6271472.html)

下面这篇文章，感觉讲的非常好：

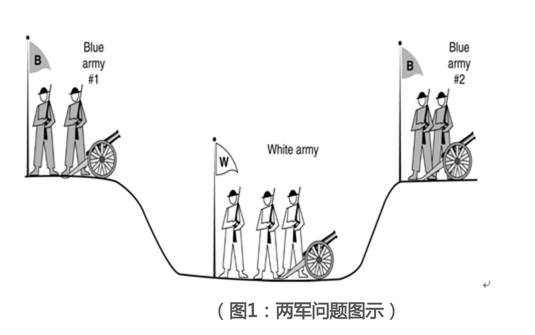
<http://www.8btc.com/baizhantingjiangjun>

拜占庭将军问题是一个共识问题: 首先由Leslie Lamport与另外两人在1982年提出，被称为The Byzantine Generals Problem或者Byzantine Failure。核心描述是军中可能有叛徒，却要保证进攻一致，由此引申到计算领域，发展成了一种容错理论。随着比特币的出现和兴起，这个著名问题又重入大众视野。

应该明确的是，拜占庭将军问题中并不去考虑通信兵是否会被截获或无法传达信息等问题，即消息传递的信道绝无问。Lamport已经证明了在消息可能丢失的不可靠信道上试图通过消息传递的方式达到一致性是不可能的。所以，在研究拜占庭将军问题的时候，我们已经假定了信道是没有问题的，并在这个前提下，去做一致性和容错性相关研究。如果需要考虑信道是有问题的，这涉及到了另一个相关问题：两军问题。

**1.2.与拜占庭将军相关问题：两军问题**

正如前文所说，拜占庭将军问题和两军问题实质是不一样的。国内大量解释拜占庭将军问题的文章将两者混为一谈，其实是混淆了两个问题的实质，由此造成了许多误解。这两个问题看起来的确有点相似，但是问题的前提和研究方向都截然不同。

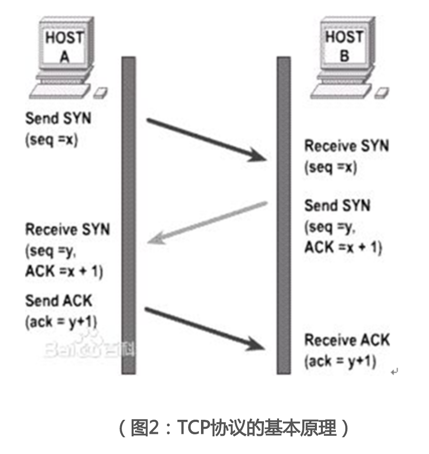


看到这里您可能发现两军问题和拜占庭将军问题有一定的相似性，但我们必须注意的是，通信兵得经过敌人的沟渠，在这过程中他可能被捕，也就是说，两军问题中信道是不可靠的，并且其中没有叛徒之说，这就是两军问题和拜占庭将军问题的根本性不同。由此可见，大量混淆了拜占庭将军问题和两军问题的文章并没有充分理解两者。

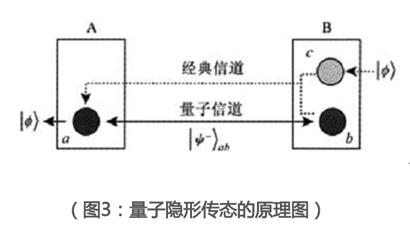
两军问题的根本问题在于信道的不可靠，反过来说，如果传递消息的信道是可靠的，两军问题可解。然而，并不存在这样一种信道，所以两军问题在经典情境下是不可解的.

但我们能不能通过一种相对可靠的方式来解决大部分情形呢？这需要谈到TCP协议。事实上，搜索“两军问题与三次握手”，您一定可以找到大量与TCP协议相关的内容。

权当笔者是班门弄斧，这里仅用最浅显易懂的方式科普TCP协议的原理和局限，可能存在一些毛刺，请多包涵。



那么，是否能够找到一个理论方法来真正的破解两军问题呢？答案是有的，量子通讯协议，笔者并没有能力弄清这个颇为高深的问题。据我的理解，处于量子纠缠态的两个粒子，无论相隔多远都能够彼此同步，光是直观的来看，这个效应可以用来实现保密通讯。



但是由于测不准原理，一测量粒子状态就会改变其状态，所以通讯时还必须通过不可靠信道发送另一条信息。尽管这个“另一条信息”是不可靠的，但是由于已经存在了一条绝对可靠的信道（量子纠缠），保证了另一条信道即使不可靠也能保证消息是可靠的，否则至少被窃取了一定能够被发现。

**2.1. 拜占庭将军问题实质**

至此，我们将拜占庭将军问题简化成了，所有忠诚的将军都能够让别的将军接收到自己的真实意图，并最终一致行动；而形式化的要求就是，

“一致性”与“正确性”。

**2.2.形式化条件的推演**

拜占庭问题，关键要达到下面两个目标：

**IC1：**所有忠诚的副官遵守一个命令，即一致性。

**IC2：**若司令是忠诚的，每一个忠诚的副官遵守他发出的命令，即正确性。

在经典的情形下，我们可以找到两种办法，口头协议和书面协议。笔者将会逐一探讨两种算法的推演和证明，其中证明部分并不会采用纯推理，而以介绍证明思路为主。

**Part3：口头协议**

首先，我们明确什么是口头协议。我们将满足以下三个条件的方式称为口头协议：

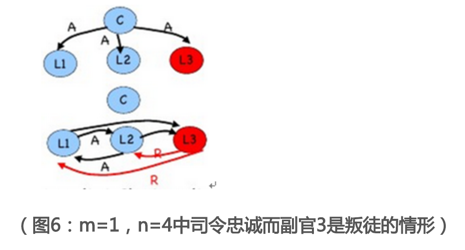
**A1：**每个被发送的消息都能够被正确的投递

**A2：**信息接收者知道是谁发送的消息

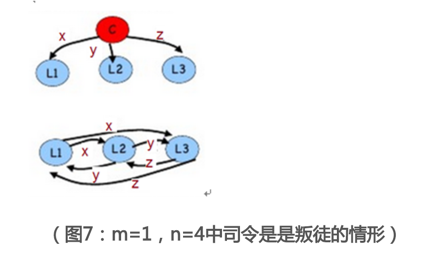
**A3：**能够知道缺少的消息

先告知结论：采用口头协议，若叛徒数少于1/3，则拜占庭将军问题可解。也就是说，若叛徒数为m，当将军总数n至少为3m+1时，问题可解（即满足了IC1和IC2）。

**3.1.口头协议算法OM(m)**



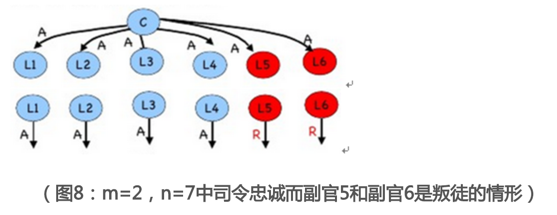
倘若司令在OM(1)中给各副官发送的消息都是进攻（A），之后OM(0)时，叛徒副官3给副官1和副官2说他收到的消息是撤退（R）。那么对于副官1（或副官2）来说，他综合司令、副官3和副官2（或副官1）后得到的消息向量都将会是(A,A,R)，利用majority函数之后，将会采用A，满足了IC1和IC2（回顾IC1：所有忠诚的副官遵守一个命令，IC2：若司令是忠诚的，每一个忠诚的副官遵守他发出的命令）。



倘若司令是叛徒，那么我们已经不需要满足IC2。为方便，我们假设叛徒司令在OM(1)会给三个副官发送的信息是(x,y,z)，其中x，y，z都可以是A或R的任意一种。之后，三位忠诚的副官将会按照OM(0)要求的那样，交换他们收到的信息。

对于副官1，他综合司令、副官2和副官3后得到的消息向量将会是(x,y,z)，可以发现对于其他两个忠实的副官，他们得到的消息向量也将是(x,y,z)。不管x，y，z如何变化，majority(x,y,z)对于三人来说都是一样的，所以三个副官将会采用一致的行动。

（3）m=2，n=7的情形 接下来，我们将讨论m=2，n=7的情形，虽然只是多了一个叛徒，但是这里会出现递归过程，所以会复杂很多。



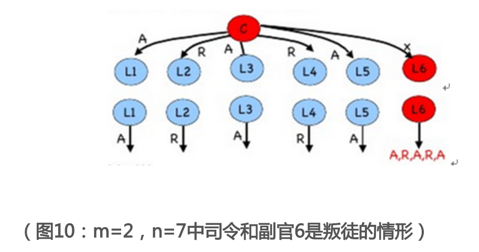
在OM(2)中，司令将攻击命令（A）传给各个副官。在OM(1)中，忠诚的副官们将会发送他们收到的消息（A），但由于副官5和副官6是叛徒，他们将会发送别的信息（比如R）。这时，忠诚的副官们将会采用使用OM(1)中的方法来确定各个v1~v6。例如，对于副官1，他收到了司令传来的命令，他会直接采用majority函数综合司令和其他将军传来的信息吗？他不会，因为这还在OM(1)中，他并不知道司令是不是叛徒，他会利用询问别人的方式来确认将军的命令，但是按照算法他会把司令的命令作为v1（即v1=A）并传给其他人。

接下来他会努力取得其他的v2~v6的值，这时已经在OM(1)中了，副官1绝不会轻易相信别人传来的消息，比如副官2给他传来了命令A，但是他会怀疑副官2传来的消息，所以他用OM(1)大法，问其他人副官2传给了他们什么，副官3和副官4诚实的告诉副官1（也就是所有人都要问到，都要听到），副官2给他们传的是A，而这时副官5和副官6又要撒谎了，他们又乱说，我们姑且假定他们传来的是x’和y’吧。这样，终于进入到了OM(0)，这时副官1将会综合其他副官对于v2的反馈，得到向量(A,A,A,x’,y’)，再利用majority函数，得到了v2=A。如下图，这是副官1在OM(1)中得到的信息（x，y等表示了不确定的命令）。



所以，我们可以发现忠诚的副官采用的命令都是A（满足IC1），并且和忠诚的将军的命令一致（满足IC2）。至此，您应该已经明白了这个算法是怎么递归的，不管m等于多少，都只是一个算法步骤多寡的问题。

至于司令是叛徒的情形，其实是相似的，这里简单的再提一下便于理解。若您已经明白了算法过程，完全可以跳过。



最终他们采用的行动都会是A（满足了IC1），而司令是叛徒不需要满足IC2。

**我们回顾一下命题：**

**将军总数为n，叛徒数量为m，OM(m)可以实现，在n>3m（至少3m+1）的情况下，使得：**

**IC1：**所有忠诚的副官遵守一个命令。

**IC2：**若司令是忠诚的，每一个忠诚的副官遵守他发出的命令。

**Part4：书面协议**

除了A1，A2和A3以外，我们在口头协议之上添加一个条件A4，使之成为书面协议

**A4**：（a）签名不可伪造，一旦被篡改即可发现，而叛徒的签名可被其他叛徒伪造；（b）任何人都可以验证签名的可靠性。

回顾A1-3：

**A1：**每个被发送的消息都能够被正确的投递

**A2：**信息接收者知道是谁发送的消息

**A3：**能够知道缺少的消息

那么，我们先说结论：对于任意m，最多只有m个背叛者情况下，算法SM(m)能解决拜占庭将军问题。也就是说，在使用签名的情况下，书面协议可以打破三模冗余的僵局，使用了签名的情况下，只要知道了叛徒数量，我们就可以利用SM(m)算法解决拜占庭将军问题。

**4.1.书面协议算法SM(m)**

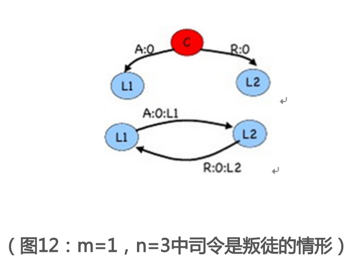
笔者对书面协议尽量简短的介绍。回顾

**IC1：**所有忠诚的副官遵守一个命令，即一致性。

**IC2：**若司令是忠诚的，每一个忠诚的副官遵守他发出的命令，即正确性。

**4.2.书面协议实例推演**

举个例子，n=3，m=1，其中司令是叛徒，这是口头协议不能解决的状况。



很显然，副官1得到的V1={A,R}，副官2得到相同的V2={A,R}。他们采用choice函数后得到的命令一定相同。

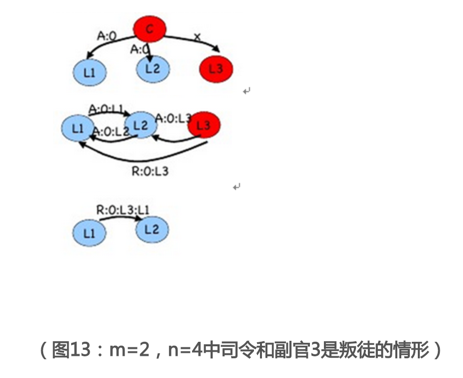
回顾之前的口头协议，m=1, n=3是无法解决的。因为：

如果司令是叛徒，两个副官忠诚，司令会发送两个不同的命令。当两个副官对照命令时，他们又凌乱了，无法判断司令是叛徒或者对方是叛徒，从而又无法判断。这个情形非常简易的说明了三模冗余是无法动态容错的。

可以看出，OM无法处理三模冗余，是因为无法判断消息本来是司令乱发的，还是另一个副官篡改的。 而SM书面协议能够保证消息不被篡改，所以就OK了。

**关键点就在于签名不可伪造。**

相似的，n=4，m=2，其中司令是叛徒，这同样是口头协议不能解决的状况。



书面协议的本质就是引入了签名系统。

**结论**

书面协议的结论非常令人兴奋，这不是解决了拜占庭将军问题了吗？但请注意我们在A1~A4中实际上是添加了一些条件的，这使得拜占庭将军问题在这些假设下能够解决，但是在实际状况中却会有一些问题。观察A1~A4，我们做了一些在现实中比较难以完成的假设，比如没考虑传输信息的延迟时间，书面协议的签名体系难以实现，而且签名消息记录的保存难以摆脱一个中心化机构而独立存在。

回顾A1-A4：

**A1：**每个被发送的消息都能够被正确的投递

**A2：**信息接收者知道是谁发送的消息

**A3：**能够知道缺少的消息

**A4**：（a）签名不可伪造，一旦被篡改即可发现，而叛徒的签名可被其他叛徒伪造；（b）任何人都可以验证签名的可靠性。

事实上，存在能够完美解决书面协议实际局限的方法，这个方法就是区块链。可以看我的下一篇文章：

[http://www.cnblogs.com/charlesblc/p/6272785.html](http://www.cnblogs.com/charlesblc/p/6272785.html" \t "_blank)

分类: [Hadoop&Spark&数据处理](http://www.cnblogs.com/charlesblc/category/915501.html),[Linux网络编程](http://www.cnblogs.com/charlesblc/category/820189.html),[并发编程](http://www.cnblogs.com/charlesblc/category/914733.html)

[**好文要顶**](javascript:void(0);) [**关注我**](javascript:void(0);) [**收藏该文**](javascript:void(0);) **[https://common.cnblogs.com/images/icon_weibo_24.pnghttps://common.cnblogs.com/images/wechat.png](javascript:void(0);)**

[https://pic.cnblogs.com/face/sample_face.gif](http://home.cnblogs.com/u/charlesblc/)

[blcblc](http://home.cnblogs.com/u/charlesblc/)  
[关注 - 9](http://home.cnblogs.com/u/charlesblc/followees)  
[粉丝 - 45](http://home.cnblogs.com/u/charlesblc/followers)

[+加关注](javascript:void(0);)

0

0

[«](http://www.cnblogs.com/charlesblc/p/6267329.html) 上一篇：[Storm学习](http://www.cnblogs.com/charlesblc/p/6267329.html" \o "发布于2017-01-10 00:25)  
[»](http://www.cnblogs.com/charlesblc/p/6272785.html) 下一篇：[区块链、比特币与拜占庭将军问题](http://www.cnblogs.com/charlesblc/p/6272785.html" \o "发布于2017-01-11 11:04)

posted @ 2017-01-11 00:29 [blcblc](http://www.cnblogs.com/charlesblc/) 阅读(1415) 评论(1) [编辑](https://i.cnblogs.com/EditPosts.aspx?postid=6271472) [收藏](https://www.cnblogs.com/charlesblc/p/6271472.html)

**评论**

[#1楼](https://www.cnblogs.com/charlesblc/p/6271472.html#3661895)36618952017/4/7 18:44:09 2017-04-07 18:44 | [[\_笑爺\_]](http://home.cnblogs.com/u/1142060/)

关于 4.2.书面协议实例推演 的地方，有类似图九的图来讲解吗？

[支持(0)反对(0)](javascript:void(0);)

[刷新评论](javascript:void(0);)[刷新页面](https://www.cnblogs.com/charlesblc/p/6271472.html)[返回顶部](https://www.cnblogs.com/charlesblc/p/6271472.html#top)

**注册用户登录后才能发表评论，请** [**登录**](javascript:void(0);) **或** [**注册**](javascript:void(0);)**，**[**访问**](http://www.cnblogs.com/)**网站首页。**

[【推荐】超50万VC++源码: 大型工控、组态\仿真、建模CAD源码2018！](http://www.ucancode.com/index.htm)  
[【活动】杭州云栖·2050大会-追逐早上七八点钟的太阳-源点](https://www.cnblogs.com/cmt/p/8462669.html" \t "_blank)  
[【推荐】微信小程序一站式部署 多场景模板定制](https://cloud.tencent.com/solution/la?fromSource=gwzcw.766790.766790.766790)

[](http://www.gcpowertools.com.cn/products/spreadjs/?utm_source=cnblogs&utm_medium=blogpage&utm_term=bottom&utm_content=SpreadJS2&utm_campaign=community)

**最新IT新闻**:  
· [Guido van Rossum 获得计算机历史博物馆的会员奖](https://news.cnblogs.com/n/590269/)  
· [以德跑路、被攻击？一文详解去中心化交易所以德的底层运作协议](https://news.cnblogs.com/n/590268/)  
· [马斯克卫星互联网计划：高速上网，没有中间商赚差价](https://news.cnblogs.com/n/590266/)  
· [全资收购探探 陌陌正在打造中国陌生社交帝国](https://news.cnblogs.com/n/590264/)  
· [特斯拉推出工作场所充电计划 为企业客户免费提供充电站](https://news.cnblogs.com/n/590263/)  
» [更多新闻...](http://news.cnblogs.com/)

[](http://click.aliyun.com/m/34770/)

**最新知识库文章**:

· [作为一个程序员，数学对你到底有多重要](http://kb.cnblogs.com/page/590141/)  
· [领域驱动设计在互联网业务开发中的实践](http://kb.cnblogs.com/page/586236/)  
· [步入云计算](http://kb.cnblogs.com/page/585502/)  
· [以操作系统的角度述说线程与进程](http://kb.cnblogs.com/page/531409/)  
· [软件测试转型之路](http://kb.cnblogs.com/page/141729/)

» [更多知识库文章...](http://kb.cnblogs.com/)

**公告**

朋友，任性地打赏我吧！ 

昵称：[blcblc](https://home.cnblogs.com/u/charlesblc/)  
园龄：[1年11个月](https://home.cnblogs.com/u/charlesblc/" \o "入园时间：2016-02-28)  
粉丝：[45](https://home.cnblogs.com/u/charlesblc/followers/)  
关注：[9](https://home.cnblogs.com/u/charlesblc/followees/)

[+加关注](javascript:void(0);)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | [**<**](javascript:void(0);) | **2018年2月** | [**>**](javascript:void(0);) | | | | | | | |
| **日** | **一** | **二** | **三** | **四** | **五** | **六** |
| 28 | 29 | 30 | 31 | [**1**](http://www.cnblogs.com/charlesblc/archive/2018/02/01.html) | [**2**](http://www.cnblogs.com/charlesblc/archive/2018/02/02.html) | [**3**](http://www.cnblogs.com/charlesblc/archive/2018/02/03.html) |
| [**4**](http://www.cnblogs.com/charlesblc/archive/2018/02/04.html) | [**5**](http://www.cnblogs.com/charlesblc/archive/2018/02/05.html) | [**6**](http://www.cnblogs.com/charlesblc/archive/2018/02/06.html) | [**7**](http://www.cnblogs.com/charlesblc/archive/2018/02/07.html) | [**8**](http://www.cnblogs.com/charlesblc/archive/2018/02/08.html) | [**9**](http://www.cnblogs.com/charlesblc/archive/2018/02/09.html) | [**10**](http://www.cnblogs.com/charlesblc/archive/2018/02/10.html) |
| [**11**](http://www.cnblogs.com/charlesblc/archive/2018/02/11.html) | [**12**](http://www.cnblogs.com/charlesblc/archive/2018/02/12.html) | [**13**](http://www.cnblogs.com/charlesblc/archive/2018/02/13.html) | [**14**](http://www.cnblogs.com/charlesblc/archive/2018/02/14.html) | [**15**](http://www.cnblogs.com/charlesblc/archive/2018/02/15.html) | [**16**](http://www.cnblogs.com/charlesblc/archive/2018/02/16.html) | 17 |
| 18 | [**19**](http://www.cnblogs.com/charlesblc/archive/2018/02/19.html) | 20 | [**21**](http://www.cnblogs.com/charlesblc/archive/2018/02/21.html) | [**22**](http://www.cnblogs.com/charlesblc/archive/2018/02/22.html) | [**23**](http://www.cnblogs.com/charlesblc/archive/2018/02/23.html) | 24 |
| 25 | 26 | 27 | 28 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 |

**搜索**





**常用链接**

* [我的随笔](http://www.cnblogs.com/charlesblc/p/)
* [我的评论](http://www.cnblogs.com/charlesblc/MyComments.html)
* [我的参与](http://www.cnblogs.com/charlesblc/OtherPosts.html)
* [最新评论](http://www.cnblogs.com/charlesblc/RecentComments.html)
* [我的标签](http://www.cnblogs.com/charlesblc/tag/)
* [更多链接](https://www.cnblogs.com/charlesblc/p/6271472.html)

**我的标签**

* [Linux](http://www.cnblogs.com/charlesblc/tag/Linux/)(2)
* [网络编程](http://www.cnblogs.com/charlesblc/tag/%E7%BD%91%E7%BB%9C%E7%BC%96%E7%A8%8B/)(1)
* [leetcode](http://www.cnblogs.com/charlesblc/tag/leetcode/)(1)

**随笔分类**(2977)

* [C++(186)](http://www.cnblogs.com/charlesblc/category/821418.html)
* [git&项目管理(14)](http://www.cnblogs.com/charlesblc/category/908897.html)
* [Hadoop&Spark&数据处理(173)](http://www.cnblogs.com/charlesblc/category/915501.html)
* [Javascript(33)](http://www.cnblogs.com/charlesblc/category/896278.html)
* [Java学习(137)](http://www.cnblogs.com/charlesblc/category/886919.html)
* [leetcode(258)](http://www.cnblogs.com/charlesblc/category/817936.html)
* [libevent学习(26)](http://www.cnblogs.com/charlesblc/category/820187.html)
* [Linux命令&系统(138)](http://www.cnblogs.com/charlesblc/category/820196.html)
* [Linux网络编程(172)](http://www.cnblogs.com/charlesblc/category/820189.html)
* [MySQL&Redis&MQ(65)](http://www.cnblogs.com/charlesblc/category/890571.html)
* [PHP&Python学习(157)](http://www.cnblogs.com/charlesblc/category/891846.html)
* [ToDo(112)](http://www.cnblogs.com/charlesblc/category/891212.html)
* [安装部署(90)](http://www.cnblogs.com/charlesblc/category/886676.html)
* [并发编程(81)](http://www.cnblogs.com/charlesblc/category/914733.html)
* [代码备份(25)](http://www.cnblogs.com/charlesblc/category/1089669.html)
* [读书笔记(65)](http://www.cnblogs.com/charlesblc/category/917976.html)
* [机器学习&数据挖掘(472)](http://www.cnblogs.com/charlesblc/category/838119.html)
* [计算广告(94)](http://www.cnblogs.com/charlesblc/category/992678.html)
* [建站&大型网站建设(78)](http://www.cnblogs.com/charlesblc/category/896494.html)
* [面试题分析(165)](http://www.cnblogs.com/charlesblc/category/917925.html)
* [前端(69)](http://www.cnblogs.com/charlesblc/category/886988.html)
* [数学知识(5)](http://www.cnblogs.com/charlesblc/category/1160027.html)
* [算法(276)](http://www.cnblogs.com/charlesblc/category/820186.html)
* [消息队列(6)](http://www.cnblogs.com/charlesblc/category/906750.html)
* [行业&其他&电商(75)](http://www.cnblogs.com/charlesblc/category/893414.html)
* [要填的坑(2)](http://www.cnblogs.com/charlesblc/category/922785.html)
* [游戏开发(3)](http://www.cnblogs.com/charlesblc/category/931471.html)

**随笔档案**(1551)

* [2018年2月 (97)](http://www.cnblogs.com/charlesblc/archive/2018/02.html)
* [2018年1月 (60)](http://www.cnblogs.com/charlesblc/archive/2018/01.html)
* [2017年12月 (49)](http://www.cnblogs.com/charlesblc/archive/2017/12.html)
* [2017年11月 (20)](http://www.cnblogs.com/charlesblc/archive/2017/11.html)
* [2017年10月 (17)](http://www.cnblogs.com/charlesblc/archive/2017/10.html)
* [2017年9月 (32)](http://www.cnblogs.com/charlesblc/archive/2017/09.html)
* [2017年8月 (23)](http://www.cnblogs.com/charlesblc/archive/2017/08.html)
* [2017年7月 (63)](http://www.cnblogs.com/charlesblc/archive/2017/07.html)
* [2017年6月 (64)](http://www.cnblogs.com/charlesblc/archive/2017/06.html)
* [2017年5月 (63)](http://www.cnblogs.com/charlesblc/archive/2017/05.html)
* [2017年4月 (98)](http://www.cnblogs.com/charlesblc/archive/2017/04.html)
* [2017年3月 (244)](http://www.cnblogs.com/charlesblc/archive/2017/03.html)
* [2017年2月 (165)](http://www.cnblogs.com/charlesblc/archive/2017/02.html)
* [2017年1月 (101)](http://www.cnblogs.com/charlesblc/archive/2017/01.html)
* [2016年12月 (116)](http://www.cnblogs.com/charlesblc/archive/2016/12.html)
* [2016年11月 (88)](http://www.cnblogs.com/charlesblc/archive/2016/11.html)
* [2016年10月 (163)](http://www.cnblogs.com/charlesblc/archive/2016/10.html)
* [2016年9月 (38)](http://www.cnblogs.com/charlesblc/archive/2016/09.html)
* [2016年8月 (15)](http://www.cnblogs.com/charlesblc/archive/2016/08.html)
* [2016年7月 (5)](http://www.cnblogs.com/charlesblc/archive/2016/07.html)
* [2016年6月 (14)](http://www.cnblogs.com/charlesblc/archive/2016/06.html)
* [2016年5月 (10)](http://www.cnblogs.com/charlesblc/archive/2016/05.html)
* [2016年4月 (6)](http://www.cnblogs.com/charlesblc/archive/2016/04.html)

**积分与排名**

* 积分 - 210358
* 排名 - 1094

**最新评论**

* [1. Re:Kafka安装与实验](http://www.cnblogs.com/charlesblc/p/6046023.html#3891881)
* comuser端启动的命令过时了，应该用bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic test --f......
* --雪中鱼01
* [2. Re:【转载】协同过滤 & Spark机器学习实战](http://www.cnblogs.com/charlesblc/p/6165201.html#3857410)
* 楼主您好，您代码中的oneratings这个data怎么得到的
* --小白驾不到
* [3. Re:libevent总结学习](http://www.cnblogs.com/charlesblc/p/6286685.html#3852163)
* thank you：关于libevent的写操作我一直没有搞明白，读数据流程相对很好理解，就是对端有数据到达了，从而触发了读事件，继而调用读回调函数，这个读事件的触发，和读回调函数的调用是libeve......
* --cmlw001
* [4. Re:检验开发团队好不好的12个问题](http://www.cnblogs.com/charlesblc/p/7074637.html#3815125)
* “走廊可用性测试” 指的是什么
* --jsw
* [5. Re:国内maven库镜像（阿里云）](http://www.cnblogs.com/charlesblc/p/6104804.html#3801997)
* 好用
* --韦驮天

**阅读排行榜**

* [1. 国内maven库镜像（阿里云）(30653)](http://www.cnblogs.com/charlesblc/p/6104804.html)
* [2. Hadoop与Spark比较(14402)](http://www.cnblogs.com/charlesblc/p/6206198.html)
* [3. C++的一些不错开源框架，可以学习和借鉴(11960)](http://www.cnblogs.com/charlesblc/p/5703557.html)
* [4. 消息队列-推/拉模式学习 & ActiveMQ及JMS学习(9313)](http://www.cnblogs.com/charlesblc/p/6045238.html)
* [5. 什么是HotSpot VM & 深入理解Java虚拟机 JVM(8652)](http://www.cnblogs.com/charlesblc/p/5993804.html)

**评论排行榜**

* [1. 【转载】Java中的锁机制 synchronized & 偏向锁 & 轻量级锁 & 重量级锁 & 各自优缺点及场景 & AtomicReference(2)](http://www.cnblogs.com/charlesblc/p/5994162.html)
* [2. 【特别好】【位运算】maximum-xor-of-two-numbers-in-an-array(2)](http://www.cnblogs.com/charlesblc/p/5966368.html)
* [3. 国内maven库镜像（阿里云）(2)](http://www.cnblogs.com/charlesblc/p/6104804.html)
* [4. Kafka安装与实验(1)](http://www.cnblogs.com/charlesblc/p/6046023.html)
* [5. 【转载】全栈工程师-Hadoop, HBase, Hive, Spark(1)](http://www.cnblogs.com/charlesblc/p/6014158.html)

**推荐排行榜**

* [1. 【转载】Java中的锁机制 synchronized & 偏向锁 & 轻量级锁 & 重量级锁 & 各自优缺点及场景 & AtomicReference(4)](http://www.cnblogs.com/charlesblc/p/5994162.html)
* [2. 消息队列-推/拉模式学习 & ActiveMQ及JMS学习(4)](http://www.cnblogs.com/charlesblc/p/6045238.html)
* [3. Hadoop与Spark比较(3)](http://www.cnblogs.com/charlesblc/p/6206198.html)
* [4. 【转】Spring框架深入理解(2)](http://www.cnblogs.com/charlesblc/p/6124224.html)
* [5. Java并发学习 & Executor学习 & 异常逃逸 & 同步互斥Best Practice & wait/notify, conditon#await/signal(1)](http://www.cnblogs.com/charlesblc/p/6097111.html)

Copyright ©2018 blcblc