如何学好 Linux、C++，并搞定 BAT 面试

简介

本科的时候对Linux特别感兴趣，心中向往成为一名运维工程师，就开始没日没夜的看相关的书籍，到了大约2013年前后的时候发现DevOps开始流行起来了，就开始学习Python希望成为一名DevOps工程师，后来出去实习发现从事运维相关的工作并不是我的追求，苦于在合肥这样的城市真的很难找到一份专职做DevOps的地方（当时在科大讯飞的时候，做的就是纯运维的工作，该公司的DevOps也才有个雏形），所以我萌生了考研的想法，希望在Linux内核这个层面做深造，我选择了西安邮电大学，因为该校的陈莉君老师是我比较崇拜的对象，一直在拜读她的《深入理解Linux内核》一书。

后来上了研究生后我开始专注Linux内核、C/C++服务端开发等工作。至于为何选择C/C++，我的理由很简单，大家都去学Java了，我要是也去学Java，那么我的优势何在，而且C/C++也更偏向底层是我比较感兴趣的地方，所以我选择了C/C++。我的研究生三年所有的心思都投入到了C/C++还有Linux内核，不敢说对C/C++有什么独特的见解，至少我觉得我的学习经历还是可以给大家作为一个参考。

Linux从运维到DevOps

先简单花一些篇幅介绍下我的这段学习经历吧，我的Linux启蒙老师，还是要源于一些培训公司在学习做的免费培训，大学那会经常有一些培训公司会来我们学习做免费培训，想让我们花钱去培训。我记得当时最流行的两个证书一个是RHCE（RedHat相关的认证），另外一个则是CCNA、CCNP（思科网络相关的认证），当时的我深深被Linux吸引，opensource深深吸引了我，Linux那酷炫的黑框框吸引了我。我理想中的Hacker应该就是整天在黑框框中瞧着一堆看不懂的字符。就这样我开始一头扎到Linux的世界中，我的第一本入门书是大学图书馆借的一本linux用户指南，具体的书名已经记不起来了。后来开始读鸟哥的私房菜，这本书在当时真的很好，我没有想太多，只知道疯狂的读完整本书，一遍、二遍……，就这样我读了五遍，上面的实验不停的练习、命令不停的练习。

就这样我的Linux入门了，入门后我的开始迷茫，因为不知道下一步该学什么了，我又疯狂的开始寻找下一个目标“西安鹏程Linux网络服务视频“这个是带领我进入Linux最神秘的世界，在这里我发现Linux能做很多很多有趣的事情，我开始搭建Apache服务器了，我居然可以运行一个网站了，我还学会了用Linux做DHCP服务器、DNS服务器、VSFTP服务器，一时间我觉得我打开了一个新世界，后来开始接触网络、搭建路由器、防火墙等等。觉得Linux真的很奇妙，当时我还利用Linux搭建http代理服务器，然后在宿舍通过学校实验室部署的http代理服务器来免费上网。后来开始出去实习我就已经可以实现不看

任何文档，从头源码编译LNMP，并搭建discuz论坛，就靠这个本事我找到了我的第一份实习，工资是1800块。

后来发现工作了，就没有心思学习了，所以又回到了学校再好好巩固自己的基础，并下定决心开始考研深入学习Linux，在考研的期间我发现了马哥Linux，这又是我人生中另外一个起点，马老师讲课注重原理和实践，七分原理三分实践，通过他的课程我的linux水平提升了一个很大的档次，他的全套课程我完整的听过三遍，每一个课程上的实验我都做了至少五遍以上，这奠定了我的Linux基础和shell脚本的基础，此后在我的职业生涯中shell脚本一直是我最强有力的助手。在研究生阶段我还专门做过shell脚本相关的分享。

在2013到2014年这段时间开始流行自动化运维、Python，这个阶段我开始学Python，还有现在比较流行的Flask框架，我自认为我应该是最早的一批Flask框架的用户了，当时国外人出的一本《Flask Web开发：基于Python的Web应用开发实战》书，我也是第一时间从某宝上买到并阅读，只可惜后面转到C++后就没再看过了，这期间用Flask做过一个博客。在研究生阶段还帮同学和一些朋友运维过一些网站，做一些简单的调优和加固。到此为止我的DevOps之路终结了，从此走向了Linux C/C++的路上。通过上面我的这些经历我给大家简单的总结下:我觉得学好Linux运维需要做到以下几点:

1. 多做实验，实验环境完全可以通过VMware来模拟，模拟私有网络，模拟多台机器，要搞懂VMware提供的集中网络模式的工作原理(桥接网络、宿主机网络、NAT等)，这对整个Linux后续的学习帮助都非常之大，还可以通过虚拟机模拟Grub损坏并进行修复、模拟忘记密码并通过单用户模式修改密码等等。

2. LNMP、DHCP、DNS、MySQL等等这些网络服务需要完全基于源码来编译，这样更加有体感，对于编译的参数要理解，因为通过yum安装的软件包都是上一个稳定版本，并不是最新稳定版本，还有另外一点就是编译安装可以通过编译参数对软件进行一定的优化。

3. Linux基础要扎实，底层原理要理解，典型的文件系统的组成、inode和数据存放的位置、Linux进程是如何调度的、调度算法有哪些、磁盘调度算法有哪些、TCP/IP的三次握手和四次挥手的过程是如何的，网络中的数据是如何流向的（参考《构建高性能web站点》），iptables的三表五链、Nginx的网络IO模型(这个很重要，你要能讲清楚为什么Nginx要比Apache好)，马哥Linux对于这个部分的内容讲解是我最喜欢的，我强力推荐大家都去听一听马哥的视频。

4. 英文文档的阅读能力，阅读各类开源软件的官方文档是必经之路，这个也是了解一个开源软件最快的捷径，如果你没有一定的英文阅读能力，那么你只能等到有人把这些文章翻译成中文后你才能学习到。

5. 写博客，很多时候，看了视频和书后，如果你不能讲这些知识用你的语言表达出来，那么很大可能，你并没有真正的理解这些知识，通过写博客的方式会逼迫你回忆知识，然后总结出来，博客被大量人访问也会在一定程度上激励你，让你有一定的成就感，促使你把博客写的更好。

6. Shell脚本的要熟悉，运维这条路上脚本会帮我们省掉不少体力活，此外必须要学习一门编程语言Python、Golang等

C++从小白到入门

C++我是从研究生入学前的二个月开始学习，基本算是零基础吧，就大学那会学了一些C的知识，经常写一个程序一堆”烫烫烫”，真的是到了本科毕业还没搞懂C语言。然后在这个二个月我开窍了，突然发现对C语言融汇贯通了， 而这一切要归功于《C和指针》这本书，总结一下，我认为C语言要学好必须理解三个概念。

- 什么是指针?，指针和数组的关系。
- 程序分为哪几个段，能说清楚全局变量，局部变量，静态变量等分别属于哪个段，各个段的特点是什么?
- C语言的编译和链的接过程

真心不推荐在Windows上来学习C语言，因为它屏蔽了太多的细节，而这些细节却又是C程序员不可或缺的一部分。C语言这个阶段过去后，我开始学习Linux C系统编程这个部分，最开始接触的一本书就是《Unix/Linux编程实践教程》强力推荐给大家，这本书会给你介绍如何通过man手册来帮助编程，如何去实现who、cat、ls、ps等系统命令。通过这本书的学习会让你对Linux上很多的原理有一个深刻的认识。

这本书学完后我就开始看UNP和APUE，其中APUE我并不推荐给大家，我推荐给大家的是《Linux/UNIX系统编程手册》这本书的内容更全面，更新。建议大家在看这些书的时候可以做详细的笔记和代码练习，在我的博客上就有我总结的文章。系统编程ok后，就要重点看UNP了，看这本书的时候要找重点看，里面有的章节已经过时了，还有一些章节对于我们目前来说用途并不大，比如STCP的部分。对于这本书重点有三个部分。

- 各个socket API的对应到OS，做了哪些事情，比如connect后，做了哪些事情?，accept呢?，什么是RST报文?，什么是SIGPIPE，如何触发的?
- 网络IO模型，同步和异步，阻塞和非阻塞的概念，Linux上各种网络IO模型的优缺点对比，epoll、select、信号驱动IO等
- 服务器的网络编程模型，多线程、多进程、线程池等，各自优缺点

在我的博客上也有一篇文章介绍了相关的内容，学完这个后，剩下的就靠多实践和多读一些开源的项目来积累自己的经验了，这里推荐cjson、webbench、Tinyhttpd等，代码量都不大，很容易读懂，在读懂的基础上可以进行适当的改造和重写。

C语言和Linux系统编程这个部分结束后，就要开始踏入C++的世界了，自从C++11出来后，我觉得C++易学了，但是苦于现存的老的C++代码还是有很多，所以我们不得不去学习C++98相关的知识，这里我推荐《C++ primer》一书，注意是C++ primer，不是《C++ primer plus》我看书的方法都比较老套，第一遍力求看懂，第二遍开始抄代码，练习，第三遍开始总结写博客。所以这本书我前前后后看了大半年， 后面又看了C++编程思想上册，Effective C++、深度理解C++对象模型、Exceptional C++、深入理解C++11等经典书籍，看书的模式基本上都是二到三遍，通过抄书上的代码和写博客来加深记忆。看完这些书说真的，我觉得我的C++还只是一个小白，我真正蜕变要从读《Linux C++服务端编程》陈硕的这本书开始，通过这本书我觉得我的C++水平有了一个质的飞跃。我理解C++有以下几个要点(只是部分):

- RAII，这个很重要，是C++的核心，很多学习了C++的人都不知道RAII

- 值语义和对象语义，这个决定了你如何写好一个C++类
- 对象的生命周期，类的生命周期要清晰
- 智能指针，现代C++编程几乎不太可能出现delete语句，内存泄漏的问题真的很少会出现
- 各种STL和C++的一些坑，比如迭代器遍历过程中如何删除元素、std::list的size接口的复杂度居然是O(N)（C++11已经修正）等等
- 善用std::bind和std::function
- 基于对象编程和面向对象编程的区别
- 移动语义很重要
- Lambda的捕获表达式
- 搞清楚C++的三五法则
- std::string的实现方式，是否是线程安全的
- std::map和std::set的底层数据结构等

到了这个阶段后我就开始找工作了，上面的全部过程花费了研究生二年的时间，后来找到了某BAT实习的工作后，我就一边实习，一边读《Effective Modern C++》，这本书我重点推荐给大家，在我的博客上也有全部的总结，这本书讲了很多C++11种的一些实现细节、坑以及建议。最后推荐一些linux内核相关的书籍和学习方法

- 《深入理解Linux内核架构》
- 《Linux环境编程 从应用到内核》
- lwn.net
- 重点找自己感兴趣的模块来看，比如我就对文件系统
- 通过内核模块来探索，不能只看代码不练习

网络上有很多从头开始编写一个内存文件系统的文章，在我的博客上也有一个系列讲解Linux内核模块编程入门的文章，通过编写Linux内核模块可以做很多有趣的事情，比如系统调用拦截、网络拦截、做安全审计等等，通过编写内核模块可以提高对Linux内核学习的兴趣。

BAT求职之路

研究生阶段我主要面试了阿里巴巴、腾讯、网易都是C++研发工程师，只有网易拿到的是实习的Offer，其他的都是拿到了实习和正式的Offer，就C++这个岗位来说，阿里巴巴的要求明显高于腾讯和网易，网易的C++面试相对容易一些，问的很基础，感觉就是走个过场，都没问什么太难的问题，可能是因为是招实习吧，腾讯的C++面试偏基础从

OS、网络、编译原理、算法等。问题都不太难，问的比较广，阿里巴巴更侧重知识面、底层原理、解决问题的能力等。我阿里巴巴一共面了五面，问了很多C++、算法、Linux内核等知识，在整个求职过程中，我做了以下几件事:

- 刷leetcode的题目
- 从牛客网和google上搜集面试题，分门别类的进行整理，每天都回顾一下
- 拓宽自己的知识面，学习一些新的知识，比如当时流行的docker，更侧重学习其原理
- 加深自己对一些底层的OS知识的理解，比如epoll的原理、docker的cgroup机制和namespace机制的实现、文件系统VFS的实现、Linux信号、管道等的实现。

整理面试题和学习一些底层的OS知识对我整个面试过程中帮助很大，正常情况下如果你只学习Java或者C++是很难拿到阿里巴巴的Offer，阿里巴巴对应届生的知识面、知识的理解程度要求还是比较高的，在我的整个C++面试过程中，问的最多的就是IO复用、智能指针、内存泄漏的问题如何解决、如何排查Load高的问题、Linux内核相关的知识等，而这些问题对我来说早已得心应手，在搜集面试题的时候很多问题都是反复被提及到的，我只需要好好总结即可。

另外一点就是大家在准备面试题的时候不能只记一个结论，多问问为什么，举个例子，TCP/IP的三次握手和四次挥手的过程是什么样的，我们不光光要知道这个问题的结论，还要知道为什么是三次握手，四次挥手，为什么不是二次？当被问及到epoll、select的区别时，你应该从使用方法、可移植性、性能、优缺点、内核实现等多个方面分析和总结。而不是简简单单的就提及一个优缺点就完事了。你擅长的地方你应该多多引导你的面试官去问。

工作心得

在公司实习了大约1年，今年七月份正式入职，在这整个过程中我也零散的做了很多小需求，对C++的理解更加深刻，尤其是对软件工程有了一个新的理解，在此我想分享给大家。

- 要有造轮子的能力，但是不要轻易的去造轮子
- 单元测试的重要性，通过单元测试也可以提高程序员的信心，为了更好的写单测，会逼迫我们将模块之间的耦合降到最低，这样可以方便单测。
- 写易读的、可维护的代码
- Google的C++编程规范，每一条都值得细细品味
- Chromium开源项目有很多值得我们借鉴的地方
- 画UML图是程序员的基本素质，要有好的设计，设计要做评审
- CodeReview一定要有的
- 类名和变量名的易读性

可能对于很多人来说编码规范、CodeReview、UML、单测好像都是说说而已，我起初也是这么认为，感触并不是很深刻，直到我的同事开始带我的时候，我自己亲身去体验的时候才发现这其中奥妙无穷， 我们团队使用Chromium的base库作为自己的基础库，编

码规范、全都follow Chromium，至于为什么不用boost，我的理由则是，boost是一个我无法驾驭的怪兽，而chromium的base库是我可以驾驭的，可以打组合拳，代码的稳定性已经经过上亿人的考验，值得我们信赖。推荐大家看看chromium的base库，源代码很易读。通过阅读它的代码可以学习到很多知识。推荐一本书给大家《C++ API设计》，讲解了很多软件工程、设计层面的知识。

总结

上面说了很多，更侧重分享了我的学习经历，和一些感悟，可能我说的比较简单，实际上理解上面这些东西，是存在一个过程的，是一个从量变到质变的过程，在整个过程中我理解到，学习要坚持，不是一蹴而就的，要经历量变到质变的过程。