后端开发之路

现在是2020.9.12，秋招进入尾声了，回顾一年前，自己连《C++ Primer》都没看过，现在经过春招、秋招的面试锻炼，在腾讯、阿里巴巴的两段大厂实习经历，有一些经验可以分享给学弟学妹们

后端开发所需要的技术栈可以分为以下几个方面（重要性从高到低排序）：

- 一种后端语言：C/C++ or Java
- 操作系统：后端服务器一般都部署于Linux机器上，所以后端开发人员需要对操作系统有一定理解
- 算法与数据结构：程序员必须要会的，计算机性能有限，消耗资源越低越好，同时算法题也是笔试、面试时必须要跨过的一道坎
- 计算机网络：没有网络无法提供服务，后端不需要会组网（比如OSPF），但是需要掌握网络的基础知识，常见的如IP、TCP、HTTP等等，TCP三次握手四次挥手是面试高频考点
- 网络编程/并发编程：这里指socket、多线程等知识，对于C++开发也是需要掌握的，推荐陈硕写的muduo网络库以及随之出版的书
- 数据库：掌握基本的SQL语法，关于索引、B+树等知识，我对数据库研究不深，侥幸面试较少被问到
- 一种脚本语言：Shell or Python，很多时候在纯命令行上进行一些简单的开发与调试，掌握脚本语言会方便很多

没事多刷刷牛客，刷刷牛客的选择题，看看面经，看看别人的成长路线，大有裨益

我的经历

本科课程学的是C，自学过一点Java的皮毛，科研时都是用Python，在研一研二时觉得要走C++后端开发的路，原因有以下几点：

1. 实验室大团队传统都是C++，很少Java，认识的人用Java的很少
2. C++偏性能，Java偏业务，所以电商公司（阿里、美团、拼多多等）大多用Java
3. C++和底层技术，如操作系统、计算机组成原理，关联性更大，比如操作系统就是用C写的
4. C++涉及的知识面需要比较深入（必看书目不到十本），Java涉及的知识面较广（这意味着你要看很多书）
5. C++博大精深，从C++转换为其他语言要更轻松

在研二之前，没有系统复习过后端开发，看过各种杂七杂八的开发书籍，与面试直接相关的是《算法导论》、《现代操作系统》

2019下学期开始系统备战C++后端开发

9.1开始看C++ Primer（用时一个月，其实不需要花那么多时间

11.10 刷力扣

12.10 csapp

12.24 算法与数据结构

1.1 STL源码剖析（刚看完算法，所以速度很快）

1.26 刷完剑指offer，中间还有一个星期在看面经

2.10 unp重点章节

2.18 Linux多线程服务器编程by陈硕、Linux高性能服务器编程by游双（相当于面经）

中间看了极客时间的一些课程，后面就是刷题、刷面经了

推荐书目

先搬上牛客两个大神的推荐：

我的C++后台/基础架构岗位学习路线(offer大多是ssp

通信专业转CS收获BAT等20+offer学习路线分析

我十分赞同特立独行MVP的一句话：我认为没有最好的学习方案，只有最适合自己的方案。

我先把书目列在这里，给出主观的评价与建议，任君挑选

C++

建议按照顺序阅读

- 《C++ Primer》：如果你之前没学过C++，这本书是最好的入门导览，唯一缺点是太厚，新手容易迷失在细枝末节中，我的建议是找到重点，争取在一个月内看完
- 《STL源码剖析》：侯捷的书都是精品，遣词造句非常讲究，STL是C++的标准库，每个C++开发人员都应该掌握，看完这本书，你会知道STL的一些巧妙之处
- 《Effective C++》：很多面试题的出处，但新手看可能不知所以然，当有过一定开发经验在回过头看这本书，相信会有新的收获
- 《深度探索C++对象模型》：C++的一大难点就是其内存模型，这本书是很好的读物，也是很多面试题的出处

操作系统

建议按照顺序阅读

- 《深入理解计算机系统》：包罗万象，对于底层技术都有涉及，包括操作系统、计算机网络、网络编程、汇编等等，读完后对面试没有太大直接帮助，但是对于建立自己的计算机体系结构是大有帮助的
- 《现代操作系统》：操作系统原理书，很教材，但是有些东西与Linux不太一样，建议重点关注进程、内存、文件系统、死锁等章节
- 《Linux内核设计与实现》：Linux实践书，我没看过，但口碑不错，建议与《现代操作系统》同步阅读
- 《鸟哥的Linux私房菜》：教你如何使用Linux，如果你之前仅用过Windows，那面对一个全新的操作系统肯定是痛苦万分的，更别说还没有UI交互了，老手可忽略这本书

算法与数据结构

这里的分歧较大，如果要追求完备，《算法导论》是最优选择，但那太学术了；如果要追求快捷，《大话数据结构》是很好的入门读物，但那没啥代码实现；建议选一本基于C++的算法书，这里我推荐《数据结构与算法分析——C++语言描述》，里面有很充实的代码实现，唯一缺点是有一些高级数据结构用的不多

当然，有时间的话，三本可以一起看:)

对了，建议先看《数据结构与算法分析——C++语言描述》再看《STL源码剖析》，因为STL里有很多泛型算法是前一本书里出现过的，读起来很连贯

看完就应该要刷题了，建议在春招找实习前，剑指offer至少刷一遍，力扣至少刷高频100（200更佳），多多益善

计算机网络

计算机网络的知识点比较多，推荐《计算机网络原理：自顶向下方法》，其他的没怎么看过，可以看看两位大神的推荐，同类型的书，看两本肯定比看一本好，看三本肯定比看两本好

网络编程

- 《Linux多线程服务器端编程-muduo》：必看，如果你想做一个webserver的项目，陈硕的这本书每逢春秋之际就会销量大增，正好符合春招秋招的周期，由此可知这本书的重要性
- 《Linux高性能服务器编程》：必看，非常面试向的一本书
- 《UNIX网络编程卷1》：简称unp，关于TCP/UDP编程非常详细，但书中有些知识过时了，可以跳过
- 《UNIX 环境高级编程》：简称apue，大部头，工具书，挑着看

数据库

- 《MySQL必知必会》：新手上路
- 《高性能MySQL》：高手进阶，比较底层
- 《Redis设计与实现》：进阶可以看一下这本书，大厂都在用Redis

推荐博客

阮一峰的个人网站：很多关于计算机技术的新闻、技术、讨论

C++之父的博客

酷壳：左耳朵听风/陈皓/皓子哥的博客，内容很干

结构之法、算法之道：讲解算法与数据结构，很精彩

labuladong的算法小抄：活跃于力扣的大神，公众号我也关注了，推荐

Linux Tools Quick Tutorial

各种技术博文

基础的Git、Python可以看廖雪峰的博客

后记

应世昌邀约，要给师弟师妹们一些关于面试研发岗的建议。回头一看，正式入职阿里巴巴钉钉已有四月有余，虽然去年在这个组实习过，原以为可以轻松handle工作内容，但毕竟是正式员工，有压力，有挑战，在这期间我也成长飞快，自主学习能力、排查&解决问题能力大大提升。在我不断学习与成长的过程中，我深深赞同一个道理：学的越多，不知道的越多。特别是在阿里巴巴这个技术底蕴极其深厚的公司，我每天都感觉还有很多新东西要学习，这正是因为在学习的过程中眼界更加开阔了。我有时在想，为什么研究生的时候没有学到这些知识？所以，写个后记，权当对前文的调整与补充。

语言

Java：《Java核心技术：卷1》适合有C/C++经验的人快速入门Java，因为在讲解Java知识点的时候会对比C++的实现；《深入理解JVM虚拟机》是找Java研发岗的必备书籍，这本书也同样解答了我的一个疑惑：Java可以write once, run everywhere，因为JVM屏蔽了操作系统与硬件架构的细节，而C/C++是与操作系统相关的，比如Windows C++研发工程师与Linux C++研发工程师不太一样，从工作内容上来区分，前者写Windows软件、写Qt界面可能更多一点，后者写跑在Linux上的后端服务程序更多一点，当然，我们现在说的"后端研发工程师"一般都是指Linux C++研发，因为服务器一般都是Linux操作系统的

C++：老本行了，C语言、STL、模板元编程，都需要很熟悉，C++11必须会，C++14、17、20了解。还需要说明一点的是：学C++不能只会语法，一定要深入操作系统与编译原理，在工作中，你肯定会碰到编译与链接的问题，肯定会碰到coredump或者内存泄露，这就需要你掌握gdb，会用内存分析工具

Python：这个没什么好说的，必须会的脚本语言，方便快捷

Shell：在Linux中，Shell比Python更广泛使用，高频使用的Shell命令：grep, vim, tail, sed, wc ...，我可以这样说：如果一个服务端研发工程师的黑屏操作（纯终端模式）不熟练，那他排查问题的能力肯定是非常弱的。对于互联网从业者来说，"效率"一定是非常重要的东西，比如：技术大牛凭借熟练的shell命令，只用了三分钟就从日志中定位到了问题，而菜鸟可能需要花上半天时间去排查定位。一切能提升效率的学习与折腾都是值得的。

代码风格与设计模式

代码风格是体现程序员专业性的最直接证明，如果你经常写那种成百上千行的函数，命名晦涩难懂且混乱，那真的要好好锻炼自己的抽象、整合、拆分等能力。最直接的学习路径：C++开发看Google的C++编码规约，Java开发看阿里巴巴编码规约。

设计模式在面试时出现频次较少，刷力扣只能考验程序员的抽象逻辑能力与基本STL知识，但程序员的工作不是"请你打印出三角形"，而是参与到项目构建中，实现各种不同的目标与需求，写完代码会有code review环节，mentor/师兄们在review时，不仅会考虑代码风格、功能正确性，还会考虑代码的可读性与可扩展性，这就需要程序员有良好的设计模式的概念。我们说某个人的代码水平很高，除了他很少写bug外，代码写的是否"优雅"也是评价的重要因素，而代码的优雅很大程度就取决于设计模式。编程经验不多的人光看设计模式可能没啥体感，建议读一些源码，思考代码为啥要这样写。

中间件

各大公司会广泛使用性能优异的中间件来满足业务需求，比如消息队列（rocketmq、kafka）可以对流量洪峰起到削峰填谷的作用；Redis作为高性能kv存储可以用作缓存，从而降低后端存储压力。如果你学过这些中间件，面试会很加分，如果读过源码，那会更让人眼前一亮。

其他

现在回头看，前文写的学习路线基本正确，这个后记仅是有感于正式工作后发现的技能短板，对于面试者来说，简历是敲门砖，项目是自我介绍，刷题是门槛，八股文是流程，如何在八股文流程中打动面试官？在保证知识广度足够的情况下，一定要深挖细节，这才是硬本领，面试官也更青睐这种有知识深度的人。

最后，如果你有志于在技术上有所造诣，一定要敢于折腾，我记得我第一次装Linux虚拟机的时候花了整整两天时间，那些晦涩的报错信息让人感到沮丧，但所幸自己坚持了下来。程序员很多时候都在和机器斗智斗勇，要学会享受折腾的过程。

from 一个廖少少 于2021.12.2