**我与SDN的缘分 一名初学者的学习之路与心得**

去年十一月，我在大三的计算机网络课程上与SDN初识。今年三月中旬，我有幸得到老乡学长北邮– 李呈的指引，真正地与SDN结缘，悄然走上学习之路。

[](http://s4.51cto.com/wyfs02/M02/6F/A7/wKiom1WkZzSggVwIAAB4rQTA4Rk747.jpg)

SDN，Software Defined Network，是对传统网络架构的一次革新。经过短短三四个月的学习和实践，我本着授人以渔的理念，辅以我的一些理解，将我的学习历程和心得叙写出来，送给各位想要入门的或跟我一样刚刚入门的朋友们。文中有理解不到位的地方，还望各位朋友不吝赐教，非常感谢!SDN，软件定义网络，我们关键就是弄清楚三件事：网络、软件、软件与网络怎么结合。

**一、 走进网络**

既然我们要用SDN来改造网络，当然得先了解一下网络是何物，磨刀不误砍柴工。

我对网络的了解，是从高中开始的。从OSI七层模型，到五层模型;从家庭组网，再到Socket编程实践，我对网络的兴趣不断增长。直到大二学了《计算机通信与网络》这门课，才算是对过往三四年积累的零星知识的一次大梳理，让我对网络有了一个系统性的了解。

**(1)传统网络**

传统网络，我的老师用它代指我们一直以来都在使用的网络，用以跟SDN网络区别。我是跟随着谢希仁前辈的《计算机网络》这本书学习的，也推荐给各位朋友。

跟随着大二的课程，我把五层模型的低四层学了个遍，主要是从物理层的拓扑、集线器，到数据链路层的网桥、MAC、CSMA/CD、CSMA/CA，再到网络层的路由器、最长前缀匹配、IP、ARP、OSPF、RIP、BGP，最后到传输层的UDP、TCP，掌握了这些，对我们网络的理解大有裨益。

根据我的SDN实践经验，深入理解一下最长前缀匹配，TCP的反馈重传、滑动窗口、三次握手、四次挥手，是非常有好处的。

**(2)SDN网络**

在这里，我们需要弄清楚三个问题：

① SDN是什么?

② 我们为什么需要SDN?

③ SDN可以用在何处?

学习SDN伊始，我阅读了一些介绍SDN的文献资料，还有一些控制器的白皮书。比较推荐大家从Open Network Fundation(ONF)组织的SDN白皮书入手，再辅以其他的介绍资料，了解SDN的架构是什么样，数据、控制、管理面，南向、北向、东西向，以及传统网络存在哪些不能适应新需求的问题、SDN针对这些问题有什么样的特性去应对。

对于控制器的白皮书，我看了ONOS的白皮书，还有一些OpenDayLight和Floodlight的介绍。通过这些，我们可以了解SDN网络的工作模式是什么，以及不同的应用场景对SDN网络有什么不同的要求。

**二、 编程实践**

通过上面阶段的学习，我们已经知道我们面前的SDN能做些什么，我们为什么要去用它了，这两个要点将支撑着我们继续深入钻研。

**(1)Openflow**

Openflow，是南向协议的一种，当下比较主流。通过对它的学习，我们可以搞清楚文章开头提到的“软件与网络怎么结合”这个问题。

Openflow以“flow”(“流”)去看待网络中的连接，它只负责对流的管理，不涉及交换设备属性的管理，比如给每个端口配置多少条队列等等，这些由其伴侣协议OF-Config去做。

对协议的学习，是很让人兴奋的一件事，可以先通过Openflow白皮书对其工作模式、流和流表、优点和性能局限有个理解，然后在SDNLAB、ONF等网站上下载到协议的细则说明书，具体学习。先从1.0版本入手，然后可以进行一些编程实践以加深理解，在掌握1.0的基础上，再去学习1.3版本。协议细则在我们后续的编程实践中也很有参考价值。

**(2)控制器**

控制器有RYU、NOX、POX、Floodlight(FL)、OpenDayLight(ODL)、ONOS等等，不同的控制器设计思路不同、消息/事件机制不同、性能不同、编程语言不同，以致于适用的场景场合不同、学习难度不同。大家可以多方面权衡之后，选一个作为SDN入门学习。

我一开始接触的是Floodlight，也尝试使用过ODL，最后，我选择了RYU这个小巧精干的控制器作为科研阶段的使用。对于学习者来说，控制器只是一个实现SDN的工具，关键在于跑在控制器上的模块，也就是需要我们根据应用需求去设计、编写的东西。

选好控制器之后，先通过官方的介绍或者编程手册了解控制器自身的代码文件组织，再花一点时间了解相应的编程语言，然后再看一下控制器编程手册里官方给的一个最简单模块的示例代码，了解一下一个模块在代码上有什么固定结构。如果官方的控制器代码包中含有已经写好可以直接使用的模块，也可以拿它们的代码来学习，这些在后续的实践中也很有参考价值。

**(3)Mininet**

在跟一些小伙伴的交流中，我发现有些童鞋还不是很了解Mininet这个东西，我在这里简单地说一下我的理解吧。Mininet是一个拓扑仿真工具，对我们来说，它就是帮我们虚拟地搭建了一个硬件网络，网络中有交换机，有主机，有相互之间的线路连接，通过它我们就得到了一张网，能了解到这里就差不多够了。

更深入一点的话，可以把Mininet看成一个助手或者脚本，我们用参数的形式，输入所要拓扑和网络的参数，它就帮我们调用起安装在Linux中的OVS，使用Linux提供的虚拟化技术KVM虚拟出主机host，然后再把它们连了起来，当然目前host之间的隔离性做得还不是很好。更进阶一步，还可以通过Mininet的Intf类或者直接使用OVS的命令，实现Mininet中交换机网口与Linux网口的对接，从而使我们这张网可以跟外部通信。

具体的学习过程，可以跟着mininet.org官网的WalkThrough页面做一遍，就算基本掌握Mininet这个工具了。

**(4)Need &Design& Coding**

本文假设大家都具有编程、调试方面的基本功，如果这方面还有待加强的话，可以找一些编程语言的书籍和调试技巧的文章看一看，然后做一些类似文件存取、网络聊天、数据库管理这样的小项目来练练手，编程能力自然会提高。

一个好的程序，比编码更关键的是设计，比设计更关键的是需求分析，这是我多年编程实践的感悟。SDN的编程实践和开发一款软件的软件工程实践是相通的。

首先，是对应用场景的需求分析，需要实现什么效果，需要支撑多大的规模，需要适应什么样的拓扑结构等等。这个阶段，最好尽可能地详细，特别是不能遗漏那些最基本的需求，否则可能会导致后续程序架构的大改。小的需求可以在后续进行快速迭代。

然后，就是设计。面对这些需求，首先要想，我用什么样的管理策略去实现这些需求。然后根据制定好的策略去想，我作为控制器中的模块，需要得到网络中的什么信息，包括topology、switch、link、host等的信息;需要处理哪些协议、地址、端口、字段的数据包。再接着，我需要什么样的数据结构去存储这些信息，需要设计什么样的辅助算法。设计过程中可能需要参考OpenFlow协议和控制器编程手册，看看自己是否能获取到所需的信息。

设计的过程是一个闭环反馈的过程。

最后，就是Coding实现。好的代码风格可以改善我们Coding的心情，也能提高我们Debug和Upgrade的效率。

**(5)抓包分析**

Wireshark，想必做网络的朋友都听过它的大名。它通过监听网卡，把收发的数据包全部列出来供我们查看。如果程序模块Debug确定没问题，但是功能效果就是不理想，甚至无法实现，那么就要祭出我们这张王牌了。

最近正值大三的期末，我的SDN课程的期末作业就好好地用了一把wireshark，通过添加过滤条目，可以细粒度地看到TCP三次握手、四次挥手的全过程，还有ACK、重传包、RST报文。它还可以解析应用层HTTP、Openflow等协议的数据包。

我在另一个SDN智能组播树的项目中，通过对IGMP的过滤查看，找到了能被利用的协议包，验证了组播协议的工作流程。

有时候，并不是我们的模块做得不好，而是我们对协议的了解不够。

新手上路，一是很难透彻理解通信协议和兼顾通信的各个环节，二是可能遇到网络应用中一些不可预知的工程问题。

对于第一条，我们可以通过抓包，找出是哪个环节出了问题，进而可以发现是我们在Openflow上的操作不对，还是我们对传统协议的兼容出了问题，亦或是我们在传统的通信环节上出了岔子。

我的期末作业就遇到一个典型问题例子：功能测试过程中，不同网段的两个主机互ping，明明已经收到了ICMP响应包(wireshark解析包后会有所指示)，但还是报告“主机不可达”，结果发现是主机上的默认网关没有设置好。正常情况下，收到的ping包应该是(IPa->IPb)，但是不同网段下，当网关没有设置成自己时，收到的ping包也许应该是(IPgateway->IPb)。

我的一点理解是，Openflow是对流的控制，如果一条流已经被我们成功引过去了，应该就表明我们的模块没有问题了。供大家参考。

而对于第二条，还是举我期末作业遇到的例子吧，访问HTTP时，用wget命令只会产生一个TCP连接，但是用Firefox则会产生两个TCP连接，神奇吧?如果这时候我们是对访问次数或者访问间隔做限制的话，那wget就会成功，而Firefox就会说网页打不开。这时候就不得不抓包分析了。对于这一点，我们可以设置相应的控制时延或者更高级的办法来解决。

对于这个例子，我还遇到了这一种情况，缘起于TCP的反馈重传机制。网络情况是多变的，即使是非常通畅的情况，通过抓包我发现，也会有各种原因导致TCP的意外重传，同理，一发生重传，就会导致上述访问失败的情况。

**三、 总结**

阿里巴巴的曹捷前辈在今年的DCD大会上宣布愿意率先跨出SDN的第一步，华为也在ONOS和ODL方面双管齐下，其IP部门今年的宣传片中也新加入了SDN的部分。相信SDN这个朝气蓬勃的新架构，会在未来向世人展示它的巨大威力。SDN现在大体还处于实验室和业界的研究阶段，很多东西也还没有定论，虽然已经有Google B4这样强大的成功案例，以及华为和阿里这样的高度热情，但也许一年、两年、五年、十年都无法看到它的遍地开花。所以说，学习SDN，没有足够的兴趣和毅力是无法深入下去的。

我有幸得到老乡学长北邮–李呈的指引，他带我走进了这一片新天地，真的是非常的感谢!同时也要感谢学习路上跟我一起交流的小伙伴们，相互学习，共同进步，相信美好的未来就在不远的前方!

【编辑推荐】

[SDN领域的七大创新](http://network.51cto.com/art/201507/483432.htm)

[官方数据：5次SDN大会的背后](http://network.51cto.com/art/201507/483452.htm)

[利用SDN&NFV优化数据中心](http://network.51cto.com/art/201507/483726.htm)

[SDN落地应是一场非暴力性革命](http://network.51cto.com/art/201507/483997.htm)

[一个大学生的SDN修炼记](http://network.51cto.com/art/201507/484013.htm)