NOX -- 现代网络操作系统

Thursday, February 24 2011, 2:44 PM

[注]本系列前面的三篇文章中，介绍了软件定义网络（SDN）的基本概念和相关平台。按照SDN的观点，网络的智能/管理实际上是通过控制器来实现的。本篇将介绍一个代表性的控制器实现——NOX。

现代大规模的网络环境十分复杂，给管理带来较大的难度。特别对于企业网络来说，管控需求繁多，应用、资源多样化，安全性、扩展性要求都特别高。因此，网络管理始终是研究的热点问题。

从操作系统到网络操作系统

早期的计算机程序开发者直接用机器语言编程。因为没有各种抽象的接口来管理底层的物理资源（内存、磁盘、通信），使得程序的开发、移植、调试等费时费力。而现代的操作系统提供更高的抽象层来管理底层的各种资源，极大的改善了软件程序开发的效率。

同样的情况出现在现代的网络管理中，管理者的各种操作需要跟底层的物理资源直接打交道。例如通过ACL规则来管理用户，需要获取用户的实际IP地址。更复杂的管理操作甚至需要管理者事先获取网络拓扑结构、用户实际位置等。随着网络规模的增加和需求的提高，管理任务实际上变成巨大的挑战。

而NOX则试图从建立网络操作系统的层面来改变这一困境。网络操作系统（Network Operating System）这个术语早已经被不少厂家提出，例如Cisco的IOS、Novell的NetWare等。这些操作系统实际上提供的是用户跟某些部件（例如交换机、路由器）的交互，因此称为交换机/路由器操作系统可能更贴切。而从整个网络的角度来看，网络操作系统应该是抽象网络中的各种资源，为网络管理提供易用的接口。

技术探讨

模型

NOX的模型主要包括两个部分。

一是集中的编程模型。开发者不需要关心网络的实际架构，在开发者看来整个网络就好像一台单独的机器一样，有统一的资源管理和接口。

二是抽象的开发模型。应用程序开发需要面向的是NOX提供的高层接口，而不是底层。例如，应用面向的是用户、机器名，但不面向IP地址、MAC地址等。

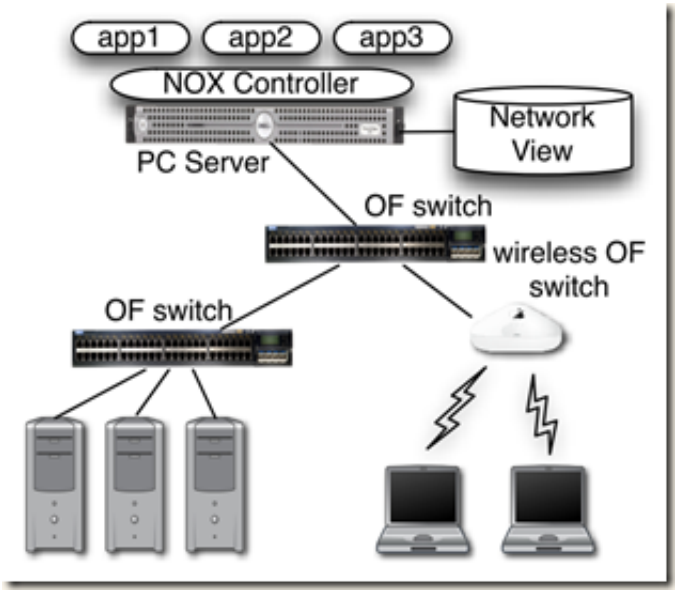
通用性

正如计算机操作系统本身并不实现复杂的各种软件功能，NOX本身并不完成对网络管理任务，而是通过在其上运行的各种“应用”（Application）来实现具体的管理任务。管理者和开发者可以专注到这些应用的开发上，而无需花费时间在对底层细节的分析上。为了实现这一目的，NOX需要提供尽可能通用（General）的接口，来满足各种不同的管理需求。

架构

组件

下图给出了使用NOX管理网络环境的主要组件。包括交换机和控制（服务）器（其上运行NOX和相应的多个管理应用，以及1个Network View），其中Network View提供了对网络物理资源的不同观测和抽象解析。注意到NOX通过对交换机操作来管理流量，因此，交换机需要支持相应的管理功能。此处采用支持OpenFlow的交换机。



操作

流量经过交换机时，如果发现没有对应的匹配表项，则转发到运行NOX的控制器，NOX上的应用通过流量信息来建立Network View和决策流量的行为。同样的，NOX也可以控制哪些流量需要转发给控制器。

多粒度处理

NOX对网络中不同粒度的事件提供不同的处理。

网包

对于10 Gbps链路，网包到达速度可以达到每秒百万级别。因此，大部分网包应该通过交换机来处

理。

网流

一般情况下，网流的更新速度要比网包的低至少数量级。网络需要通过管理应用来处理。

network view

网络自身的更新频度更低。

应用实现

NOX上的开发支持Python、C++语言，NOX核心架构跟关键部分都是使用C++实现以保证性能。代码可以从http://www.noxrepo.org获取，并遵循GPL许可。

开发接口

事件

事件包括流到达、结束、用户进入、离开、交换机进入、离开等等。

Network View和名称空间

通过检测DNS来管理用户、主机验证和名称映射，提供上层拓扑无关的编程接口。只有在改变的时候才需要。

控制

NOX对流量的控制是通过管理交换机来实现的。网络功能诸如DPI通过重定向网络流量到中间件来实现。

系统库

提供基本的高效系统库，包括路由、包分类、标准的网络服务（DHCP、DNS）、协议过滤器等

相关工作

NOX项目主页在http://noxrepo.org。

类似的项目包括SANE、Ethane、Maestro、onix、difane等