NOX: Sistema Operacional para Redes

Gustavo Henrique dos Santos Marcello
Universidade Federal de São Carlos, Campus Sorocaba
Sorocaba, São Paulo
Email: gustavohenriquesm@gmail.com

***Abstract*— This paper gives an overview of the concepts of a network operating system, showing its importance on managing networks, and illustrating with a new growing technology inside this context: NOX . Besides introducing to this tool, this paper shows examples of applications developed on it, providing a scalable and flexible solution to the complexity of managing and controlling networks at low level.**

***Index Terms*— Sistemas operacionais de redes, NOX, OpenFlow, fluxos, escalabilidade, segurança**

I. INTRODUÇÃO

As redes de computadores estão crescendo em ritmo acelerado, e é notável que gerenciar uma rede não é uma tarefa tão simples. Não faz muito tempo em que a programação de aplicações era feito em baixo nível, apresentando uma interface de programação pouco amigável.

Hoje existem sistemas operacionais que são capazes de prover abstração de alto-nível dos recursos, porém, o gerenciamento das redes ainda depende da configuração de componentes individuais em baixo-nível. A necessidade do conhecimento desses elementos dificulta e torna o processo de programação de aplicações para gerencimento da rede complicado. Por exemplo, para bloquear o acesso de um usuário com uma ACL requer saber o IP atual dele.

É evidente a necessidade de um sistema operacional para redes que providencie uma interface de programação uniformizada e centralizada para toda rede. Aplicações seriam implementadas em cima desse sistema operacional de rede, que, de fato, faria o gerenciamento da rede. Esses programas funcionariam como se toda a rede estivesse presente em uma única máquina, e não precisaria de elementos de baixo-nível (exemplo: utilizar-se de user e host name no lugar de IP e MAC) .Entretanto, para que esse sistema funcione, é necessário que tenha um mapeamento entre essas abstrações e as configurações de baixo-nível.

Nesse contexto, uma questão é levantada, seria possível a construção de um sistema operacional de rede capaz de centralizar toda a rede, atendendo todas necessidades e sem prejudicar a flexibilidade e escalabilidade desejada nos sistemas atuais?

Esse artigo visa dar uma visão geral sobre sistemas operacionais de rede e explorar uma solução para o problema que vem se mostrando eficaz e mas que ainda se encontra em processo de desenvolvimento, portanto, ainda não consolidada: o NOX. Será mostrada uma visão geral, seguida de uma explicação de seus components básico e do seu funcionamento. Também foi estudada a questão da escalabilidade em um sistema centralizado como o NOX. Por fim, alguns exemplos de utilização do NOX são mostrados.

II .Network Operating System (NOS)

A. Overview

Sistemas operacionais para rede são usados para fazer com que computadores ajam como servidores. Eles são softwares que controlam outros softwares e hardwares que rodam em uma rede, além de permitirem que vários computadores (computadores da rede) se comuniquem com um computador principal e entre si, compartilhem recursos, rodem aplicação, enviem mensagem, entre outras funcionalidades. Uma rede de computadores pode ser uma rede sem fio (wireless), rede local (LAN), rede de longo alcance (WAN), ou então uma pequena rede de alguns computadores. O "coração" de uma rede é o sistema operacional da rede.

Esse computador central tem uma interface de administração orientada a menus, na qual o administrador da rede pode realizar uma variedade de atividades, como, por exemplo, formatar discos rígidos, configurar restrições de segurança, estabelecer informações de usuários (log-ins, acessos, etc), anexar impressoras compartilhadas à rede, configurar o sistema para realizar back up automático dos dados, entre outras funcionalidades.

Outro componente de uma rede é o servidor de arquivos, um dispositivo utilizado para armazenar dados usados pelos computadores da rede. Ele pode ser um computador ou um cluster de discos rígidos externos. O sistema operacional da rede auxilia no gerenciamento do fluxo de informações entre esse servidor de arquivos e a rede de computadores. Como exemplos consolidados de sistemas operacionais de redes, podemos citar UNIX, Windows 98, Windows 2000 Server, MacIntosh e Netware.

B. *Tipo de Sistemas operacionais de rede*

Existem dois tipos principais de sistemas operacionais de rede: Peer-to-Peer e Cliente/Servidor.

Sistemas operacionais de rede peer-to-peer permitem que os usuários compartilhem recursos e arquivos em seu computador e acessem recursos e arquivos em outros computadores. Nesses sistemas, não há um servidor de arquivos ou uma fonte centralizada de gerenciamento. No peer-to-peer, todos computadores são considerados iguais e podem utilizar os recursos disponíveis na rede da mesma maneira. A figura 1 ilustra uma rede peer-to-peer.

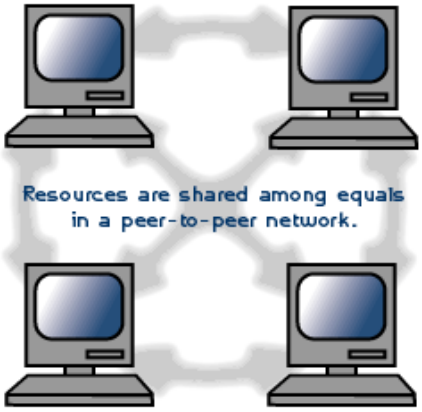

Figura 1. Rede peer-to-peer

Sistemas operacionais de rede Cliente/Servidor permite que as aplicações e funções sejam centralizadas em um ou mais servidores dedicados. Como visto na sessão anterior, em NOS Cliente/Servidor, os servidores do sistema operacional da rede são o "coração" da rede e proveem o acesso aos recursos e a segurança na rede. A figura 2 ilustra uma rede Cliente/Servidor.

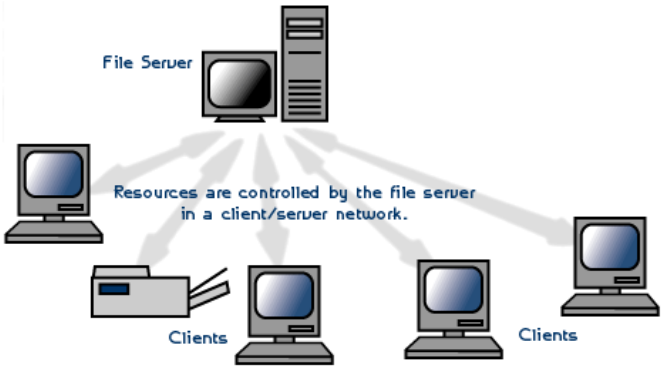

Figura 2. Rede Cliente/Servidor

As vantagens observadas numa rede Peer-to-Peer é o baixo custo por não necessitar de um servidor dedicado e pela facilidade de configuração. Já em uma rede Cliente/Servidor, que é o foco do artigo, a centralização aumenta o controle e a segurança dos dados e recursos, e, como veremos nas sessões seguintes, há possibilidade de escalabilidade, flexibilidade e interoperabilidade.

II. NOX

O NOX é uma plataforma de controle da rede Cliente/Servidor que vem sendo desenvolvida atualmente pela Nicira Networks [2]. O propósito do NOX é prover uma interface de programação de alto-nível em que aplicações de gerenciamento da rede possam ser construídas.

Abordaremos o NOX em três categorias principais: Componentes, onde serão explicados cada componente de uma rede baseada em NOX; Escalabilidade x Flexibilidade, que será a abordagem da questão de como é possível estabelecer um sistema para rede altamente escalável sem perder flexibilidade; abstração do switch, mostrando os aspectos principais da tecnologia OpenFlow, bem como seu papel no funcionamento do NOX e operação, onde é explicado como um pacote na rede é tratado pelo NOX.

A. *Componentes*

A figura 3 ilustra os componentes básicos de uma rede baseada em NOX. Essa rede é constituída por switches com suporte à abstração OpenFlow (será detalhado mais à frente), para que os servidores com o software NOX e com aplicação de gerenciamento que rodam em cima do NOX possam controlar o tráfego da rede. O software do NOX é, na verdade, um conjunto de processos controladores rodando em servidores anexados à rede. Como se pode observar, app1, app2 e app3 representam três aplicações NOX rodando em um (ou vários) servidor. Outro componente é a visão da rede (Network View), que constitui em um banco de dados rodando em um dos servidores da rede.

A visão da rede é utilizada para armazenar o resultado de observações da rede pelo NOX, tais informações podem ser a localização de elementos da rede (usuários, clientes e *middleboxes*), topologia dos comutadores e os serviços oferecidos (por exemplo, HTTP ou NFS).

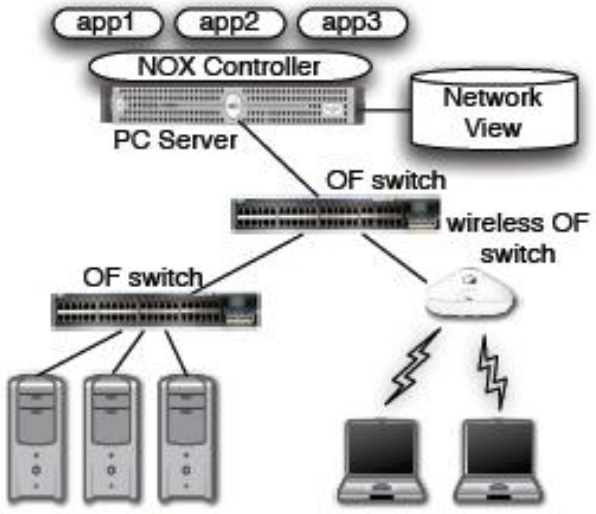

Fig. 3. Componentes de uma rede baseada em NOX [1]

B. *Escalabilidade x Flexibilidade*

Uma questão importante a ser levantada é quanto a

escalabilidade do sistema, fator crucial para qualquer sistema eletrônico com potencial de expansão. Como foi visto na sessão anterior, o NOX utiliza-se de uma visão de rede (network view), sendo ela um banco de dados com informações de elementos da rede, rodando em um dos servidores da rede e compartilhando esses dados entre todos controladores, porém, sem conter o estado atual da rede, possibilitando que, cada vez mais, um maior número de controladores (Scaling out [3]) com aplicações de gerenciamento da rede sejam utilizados, sendo qualquer controlador apto a manipular um fluxo e possibilitando a escalabilidade e consistência do sistema.

Essa divisão de controle (feita pelos controladores) e observação (através da visão da rede) seria inviável, caso houvesse um controle de rotas por pacote, e ocorreria lentidão excessiva, visto que a chegada de pacotes é em torno de milhões por segundoe e a inicialização de fluxo é na ordem de uma ou mais magnitudes menores que a taxa de chegada de pacotes, enquanto que uma mudança na visão da rede é de apenas dezenas de eventos por segundo, em uma rede com milhares de hosts [4]. Se esse controle fosse feito baseado em tabelas de prefixo, também não seria plausível, já que, apesar do número de entradas na visão da rede cairem dastricamente, o gerenciamento pecaria em controle, pois pacotes entre dois hosts teriam que seguir o mesmo caminho. Nesse contexto, o NOX optou por utilizar um meio intermediário de direcionamento de pacotes: por fluxo. O controle por fluxo é feito da seguinte forma: ao chegar um pacote ao controlador, esse é processado e tratado e os pacotes seguintes com o mesmo cabeçalho serão tratados da mesma meneira.

Dessa forma, eventos podem ser manipulados por instâncias do controlador, sendo eles um número limitado, e ainda há possibilidade de escalar o sistema sem deixar de ser flexível quanto ao controle exercido, sendo ideal para grandes redes.

C. *Abstração do Switch - OpenFlow*

Para que as aplicações NOX possam contralar o tráfego da rede, é necessário enviar instruções aos switches, pois eles que criam o canal de comunicação entre a origem e o destino do dado. Porém, para que isso ocorra, o switch deve permitir o controle a nível de fluxo (como descrito na sessão b), por isso, utiliza-se da abstração OpenFlow.

Em um switch clássico, o encaminhamento rápido de pacotes (data path) e as decisões de roteamento de alto-nível (control path) ocorrem no mesmo dispositivo. O switch OpenFlow separa essas duas funções; função de data path continua no switch, enquanto a função de decisões de roteamento de alto nível são direcionadas a um outro controlador separado (normalmente, um servidor comum). O switch OpenFlow e o controlador comunicam entre si através do protocolo OpenFlow,o qual define mensagens, como recebimento de pacotes, envio de pacotes e modificação da tabela de roteamento.

O data path de um switch OpenFlow representa a abstração de uma tabela de fluxo; cada entrada na tabela de fluxo contém um conjunto de campos dos pacotes para equiparar (header), um conjunto de elementos que serão atualizados (counters), utilizados para estatísticas ou remoção de fluxos inativos e uma ação, como enviar para porta de saída, modificar campo,enviar para controlador, entre outras. Essa tabela é representada na Figura 4.

Header Fields	Counters	Actions
Ether src	Received Packets	ALL
Ether dst	Transmitted Packets	CONTROLLER
IP src	Received Bytes	LOCAL
IP dst	Transmitted Packets	IN PORT
...	...	...

Fig. 4. Tabela de Fluxos

D. *Operação*

Ao chegar um pacote, ele é comparado na tabela de fluxos do switch, caso ocorra uma correspondência (match), o switch atualiza os counters apropriados e realiza as ações descritas. Caso contrário (o pacote não corresponde com nenhuma entrada da tabela de fluxos), o pacote é redirecionado para um processo controlador. Normalmente, esses pacotes são os primeiros pacotes de um fluxo (inicializações de fluxo). O controlador pode também escolher receber todos pacotes de um certo protocolo (ex.: DNS) e, portanto, nunca inserir uma entrada de fluxo para eles.

Com essas inicializações de fluxo e redirecionamento de tráfego, as aplicações NOX constroem uma visão da rede global a todos os controladores, sendo possível construir aplicações que a utilizem como uma base de dados para os encaminhamentos, sem que haja inconsistência dos dados. É possível, também, determinar qual rota deve ser utilizada para o encaminhamento do pacote, bem como se um fluxo será ou não permitido, sendo, assim, possível realizar o controle através de aplicações NOX (control path).

II Interface de Programação

A interface de programação do NOX envolve dois conceitos principais: eventos e a visão da rede.

A. *Evento*

Um evento ocorre sobre um fluxo, e pode ser um fluxo novo chegando, fluxo saindo, usuários entrando e deixando a rede, links novos, etc. Quando um evento ocorre, aplicações NOX utilizam um conjunto de manipuladores (*handlers*) que são registrados para executar quando esse determinado evento acontece. Os manipuladores são registrados com uma prioridade, que determinará a ordem de execução dos manipuladores. Um valor de retorno do manipulador indica se o evento continuará executando, passando para o próximo manipulador registrado (na ordem de prioridade) ou, então, parar a execução.

Alguns eventos são gerados diretamente por mensagens OpenFlow, eventos de baixo-nível, tais como: chegada e saída

de switches na rede, recebimento de pacotes e atualização dos counters (contadores) para estatísticas. Outros eventos, de alto-nível, são gerados pelo NOX como resultado do processamento desses eventos gerados pelo OpenFlow ou gerados por outras aplicações. Como exemplo, há aplicações para autenticar usuários, redirecionando um pacote HTTP para um captive portal (evento de recebimento de pacote) e, então, gerando um evento de usuário autenticado para outros aplicações utilizarem.

B. Visão da Rede

O NOX possui várias aplicações que constroem a visão da rede e um *namespace* de alto-nível para que outras aplicações possam fazer a conversão de um nome em alto-nível em um endereço de baixo-nível(ou vice versa), permitindo que as aplicações sejam desenvolvidas em qualquer topologia. Essas aplicações manipulam a autenticação do host e usuário e inferem os nomes dos hosts monitorando o DNS.

As aplicações fazem a conversão compilando declarações de alto-nível contra a visão da rede, produzindo funções de lookup de baixo-nível. Essas funções são recompiladas a cada mudança na visão da rede, mantendo, assim, consistente para todas instâncias de controlador NOX.

C. Serviços de Alto-Nível

Um conjunto de bibliotecas do sistema auxilia na implementação de funções comuns a muitas aplicações de rede no NOX. Entre elas estão o módulo de roteamento, classificação rápida de pacotes, serviços padrões de rede (DHCP e DNS, por exemplo) e um módulo de filtragem baseado em políticas.

III. EXEMPLOS DE APLICAÇÕES

A. Aplicando NOX em Datacenter

Alguns dos requerimentos de um datacenter que o NOX é capaz de suprir, dentre outros inúmeros, é quanto a questão da flexibilidade, escalabilidade e estabilidade.

Em um datacenter, o requisito de escalabilidade mais vexatório de se conseguir, é quanto o tamanho da tabela de encaminhamento do switch. O NOX não dita um algoritmo de roteamento em particular nem uma topologia, como visto anteriormente, ao contrário disso, é possível implementar algoritmos gerais de roteamento. Nesse contexto, devido a tal flexibilidade do NOX, pode-se implementar esquemas de roteamento de arquiteturas de rede como VL2 e PortLand [5], porém com maior eficiência (menor quantidade de entradas de fluxo)[6].

Além disso, a capacidade de escalabilidade dos controladores é outro fator importante. Estudos indicam que o NOX é capaz de suportar um número similar de novos fluxos que VL2 e PortLand (0,03% do número de end hosts)[6], sendo, assim, necessário um baixo número de controladores para manipular o mapeamento desses fluxos.

Mensagem em broadcast é outra questão a ser tratada em qualquer rede de grande proporção. O NOX tem a capacidade de lidar com todos broadcasts ARP e DHCP (os pacotes em broadcast são enviados ao controlador, o qual responde diretamente à fonte de maneira apropriada). Esses protocolos representam a grande maioria dos broadcasts, e o NOX, por segurança, suprime todas as outras mensagens em broadcast evitando congestionamentos na rede. Mas, em alguma situação, pode-se desejar que haja um domínio privado para broadcast, então o NOX cria entradas de fluxo na tabela de fluxo para uma spanning tree com regras de broadcast, possibilitando, então, que apenas a fonte desejada envie mensagens em broadcast para esse domínio sem ser bloqueada

.

B. Rotulagem VLAN baseada em usuário

Uma aplicação simples é a rotulagem VLAN baseada em usuário. Ela configura regras de rotulagem na autenticação de usuários, baseada em um mapeamento usuário-VLAN predefinido. O NOX é responsável por detectar todas inicializações de fluxo, atribuir o fluxo ao usuário correto e host e por enviar o evento à aplicação. Afigura 5 mostra a implementação dessa aplicação.

```
# On user authentication, statically setup VLAN tagging
# rules at the user's first hop switch
def setup_user_vlan(dp, user, port, host):
    vlanid = user_to_vlan_function(user)
    # For packets from the user, add a VLAN tag
    attr_out[IN_PORT] = port
    attr_out[DL_SRC] = nox.reverse_resolve(host).mac
    action_out = [(nox.OUTPUT, (0, nox.FLOOD)),
                  (nox.ADD_VLAN, (vlanid))]
    install_datapath_flow(dp, attr_out, action_out)
    # For packets to the user with the VLAN tag, remove it
    attr_in[DL_DST] = nox.reverse_resolve(host).mac
    attr_in[DL_VLAN] = vlanid
    action_in = [(nox.OUTPUT, (0, nox.FLOOD)),
                 (nox.DEL_VLAN)]
    install_datapath_flow(dp, attr_in, action_in)
nox.register_for_user_authentication(setup_user_vlan)
```

Figura 5. Aplicação NOX escrita em Phyton que configure estaticamente as regras de roteamento VLAN em autenticação de usuários[1]

C. Gerenciamento de Energia

Estudos sobre o gerenciamento de energia estão em alta[7], e duas tecnicas vem sendo destacadas . Uma é a redução de links inutilizados ou, então, o desligamento deles de uma vez. A visão global da rede provida pelo NOX claramente auxilia nesse processo, uma vez que há o armazenamento de estastísticas sobre essa visão da rede, como visto anteriormente.

Outra técnica é prover proxies para interceptar “ruídos” na rede, e apenas pacotes necessários cheguem aos hosts. O fato

do NOX interferir em todas inicializações de fluxo, possibilita que esses pacotes indesejados sejam interceptados e bloqueados.

D. Ethane

O sistema Ethane provê controle de acesso usando políticas centralizadas declaradas baseadas em elementos de alto nível, ou seja, entidades no namespace da visão da rede. Essa aplicação foi implementada sem o NOX (em C++) e em com NOX (Phyton)[1].

É visível que o NOX facilita o desenvolvimento de aplicações de gerenciamento, uma vez que a aplicação sem NOX foi feita em mais de 45 mil linhas e, com o NOX, em pouco mais que mil linhas. Isso ocorre porque é necessário conhecimento de alguns fundamentos da rede, com: usuários, nós, user, host etc. Isso tudo é de conhecimento das aplicações NOX, pois estão associados a eventos. A implementação basica do Ethane é checar cada fluxo com as políticas internas e o resultado ser passado ao módulo de roteamento do NOX, que irá tomar a decisão necessária.

IV. CONCLUSÕES

As redes podem conter uma variedade de complexidades que dificultam não só o seu gerenciamento, mas também o seu crescimento. Com o uso do NOX, é possível obter um nível de abstração que, além de facilitar o gerenciamento da rede, provê uma interface de programação, facilitando o desenvolvimento de novas aplicações .

A visão da rede fornecida pelo NOX, originária do projeto 4D[8] e incorporada ao NOX, trouxe flexibilidade na criação de aplicações de gerenciamento da rede, centralizando e fornecendo dados para que decisões sejam tomadas por diversas aplicações controladoras, tirando a dependência de algoritmos distribuidos.

Apesar dessa abstração auxiliar no desenvolvimento de aplicações de gerenciamento, falta ainda um suporte maior a interações entre esss aplicações. A tendência é que as técnicas atuais nas redes não sejam substituidas pelo NOX, mas que aplicações do NOX de gerenciamento sejam aderidas, principalmente no direcionamento de fluxo. Ainda falta emadurecimento

REFERÊNCIAS

[1] Natasha Gude, Teemu Koponen, Justin Pettit, Ben Pfaff, Martín Casado, Nick McKeown Scott Shenker, NOX: *Towards an Operating System for Networks).* Available: http://www.cs.yale.edu/homes/jf/nox.pdf

[2] W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

[3] "Database Scalability - Dispelling myths about the limits of database-centric architecture. Available: http://www.boic.com/scalability.htm

[4] OpenFlow e NOX : Propostas para Experimentação de Novas Tecnologias de Rede. Carlos Alberto Braz Macapuna – Universidade de Campinas (UNICAMP)

[5] A. Greenberg, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta. Vl2: A scalable and flexible data center network. In Sigcomm, 2009.

[6] Arsalan Tavakoli (UC Berkeley), Martin Casado and Teemu Koponen (Nicira Networks) and Scott Shenker (UC Berkeley, ICSI) - Applying NOX to the Datacenter

[7] B. Heller and N. McKeown. A comprehensive power management architecture. Work in progress, 2008.

[8] J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. A Clean Slate 4D Approach to Network Control and Management. In ACM SIGCOMM Computer Communication Review, 2005.