Detecção e autorreparo de anomalias em redes definidas por software*

Fernando Luiz Moro, Alexandre Amaral, Ana Paula Amaral, Rodrigo Nogueira

Instituto Federal de Educação, Ciência e Tecnologia Catarinense – Campus Camboriú
Caixa Postal 2016 – 88.340-055 – Camboriú – SC – Brasil

fernandoluizmoro@gmail.com,
{alexandre.amaral,ana.amaral,rodrigo.nogueira}@ifc.edu.br

Abstract. *The accelerated technological advances have resulted in the increased occurrence of network anomalies, such as DoS attacks. The complexity and heterogeneity of the current networks has been the main obstacle to detect and especially block the attacks without human intervention. Thus, this work presents a solution capable of detecting and blocking anomalies automatically in the context of software defined networks (SDN). The proposed solution consists of three main modules, responsible for collecting of data, detecting anomalies and applying a corrective action to solve the identified problem. To validate the solution a case study was performed using a real network attack.*

Resumo. *Os avanços tecnológicos acelerados resultaram no aumento da ocorrência de anomalias de redes, como os ataques DoS. A complexidade e heterogeneidade das atuais redes têm sido os principais óbices para a realização da tarefa de detectar e principalmente bloquear os ataques sem a intervenção humana. Assim, neste trabalho é apresentada uma solução capaz de detectar e bloquear anomalias de forma automática no contexto das redes definidas por software (SDN). A solução proposta é constituída de três módulos principais, responsáveis pela coleta dos dados, detecção de anomalias e aplicação de uma ação corretiva para solucionar o problema identificado. Para validar a solução, um estudo de caso foi realizado utilizando um ataque de rede real.*

1. Introdução

Bilhões de dispositivos como *smartphones*, *notebooks*, *smart tvs*, geladeiras, micro-ondas e celulares estão conectados à Internet, denominada agora de Internet de Todas as Coisas (*IoE – Internet of Everthing*) [Conti *et al.* 2017]. Porém, o número de ataques também tem crescido em larga escala nos últimos anos. Segundo o balanço divulgado pelo Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.br) em 2016, houve um aumento de 138% de ataques DoS (*Denial of Service*) originados por equipamentos de Internet das Coisas infectados que faziam parte de botnets [Abranet 2017]. Uma previsão realizada pela Forrester estimou que 500.000 dispositivos de IoT seriam comprometidos em 2017 [Francis 2017]. Uma das principais

*Artigo apoiado pelo IFC - Campus Camboriú, edital nº 007/GDG/IFC-CAM/2017

causas para o aumento dos ataques está nas vulnerabilidades presentes nas atuais tecnologias. Assim, com o objetivo de explorar estes pontos de falhas, cibercriminosos disparam uma série de ataques, causando alterações no comportamento da rede, também conhecidos como anomalias de redes [Amaral *et al.* 2017].

Mecanismos para detectar e bloquear as anomalias de redes se fazem necessários. Todavia, as atuais redes de computadores se tornaram complexas e heterogêneas, contendo dispositivos e *softwares* de inúmeros fabricantes denominados "caixas pretas" por [Costa 2013], ou seja, implementações integradas baseadas em *software* e *hardware* proprietário. Isto traz uma complexidade para detectar e principalmente para aplicar ações automatizadas capazes de bloquear a ocorrência da anomalia na rede.

Neste contexto, surgiram as redes definidas por software (SDN – *Software Definied Network*) que podem ser implementadas através da tecnologia OpenFlow[2]. Dentre as vantagens advindas deste paradigma está a simplicidade para a realização do gerenciamento e a execução das tarefas de detecção e bloqueio dos ataques de rede [Ahmad *et al.* 2015].

Na literatura um dos principais pontos abordados sobre as redes definidas por *software,* no que diz respeito aos aspectos de segurança, tem sido a centralidade do controlador. Embora o controle central seja a principal vantagem do SDN, também é um único ponto de falha e alvo de ataques [Dacier *et. al* 2017]. Em uma rede SDN convencional (controlador único), se um controlador falhar, a disponibilidade da rede pode ser perdida completamente [Yang 2015].

Mediante a problemática encontrada, [Mousavi 2014] apresenta um mecanismo de detecção antecipada de um ataque DDoS (*Distribute Denial of Service*) utilizando entropia. O mecanismo é implementado no controlador de uma rede SDN com o objetivo de ser uma solução leve. A proposta apresentada pelo autor é efetiva, detectando os ataques nos primeiros 250-500 novos pacotes, evitando a perda do controlador. Contudo, a tarefa de bloqueio do ataque não é apresentada pelo autor, recomendo-a como trabalho futuro.

Embora a centralidade do controlador seja enxergada como um ponto de falha, a visão holística da rede oferecida por ele, simplifica a representação dos problemas e decisões, bem como a execução das ações de reparo [Macedo *et al.* 2015]. Com isso, a tarefa de detecção e bloqueio de anomalias em uma SDN é facilitada, sem a necessidade de *hardware* especializado [Porras *et al.* 2015].

Este artigo tem como principal objetivo propor uma solução para detectar e bloquear as anomalias de rede, em uma rede definida por *software.* Diferente dos trabalhos atuais, o enfoque da proposta deste trabalho não está na detecção de ataques realizados especificamente no controlador, e sim, como explorar as características de uma SDN para detectar e principalmente aplicar ações de autorreparo em ataques lançados a qualquer dispositivo na rede. A solução proposta é composta por três módulos principais, *Flow Collector* (FC), *Network Attack Detector* (NAD) e *Action Executor* (AE). O primeiro módulo é responsável pela coleta e exportação dos fluxos IP

[2] http://archive.openflow.org/wp/learnmore/

originados dos dispositivos de redes. O segundo módulo faz a detecção de anomalias de redes através da análise dos fluxos coletados e por último, o terceiro módulo possui a função de reparar a ocorrência da anomalia com a mínima intervenção humana.

2. Fundamentação teórica

2.1. Anomalias de rede e fluxos IP

Anomalias de redes são consideradas eventos anormais, como ataques (*e.g.*, DDoS) que podem trazer inúmeros prejuízos à rede [Ahmed *et al.* 2016]. Assim, é imprescindível o desenvolvimento de mecanismos capazes de detectar e bloquear as anomalias de redes de forma automatizada, para que a integridade das informações e dispositivos envolvidos sejam preservados.

Uma das fontes de dados para a detecção de anomalias é o fluxo IP. Em uma rede de computadores, um fluxo IP é gerado pela troca de dados entre dois dispositivos de redes, sendo uma sequência de pacotes unidirecional com propriedades comuns, como endereços de origem e de destino, porta de origem e destino e protocolo observados em um dado intervalo de tempo [Amaral *et al.* 2017]. Através da análise estatística obtidas através dos fluxos é possível detectar os eventos anômalos na rede.

2.2. Redes definidas por software

A proposta de novas soluções de redes para atender a demanda das novas tecnologias é dificultada pela complexidade das redes atuais. Solucionar a ocorrência de anomalia de rede se tornou uma tarefa árdua e improfícua de ser realizado manualmente pelo administrador de rede, devido ao aumento do número de dispositivos na rede e por eles possuírem suas próprias configurações. A necessidade de criar novas soluções para as atuais redes e a falta de homogeneidade levou a criação das redes definidas por *software*.

Este novo paradigma de rede promove um padrão mais simplificado e livre das limitações encontradas nas arquiteturas de redes existentes, possibilitando um melhor controle sobre os recursos da rede [Mousavi 2014]. Atualmente os dispositivos de redes possuem implementado em seu interior o plano de dados e de controle. O plano de dados é responsável pelo encaminhamento de pacotes por meio de protocolos e regras definidas pelo plano de controle. Ambos os planos são exclusivamente definidos pelos fabricantes, dificultando a criação de novos serviços e a gerência da rede. Diferentemente, nas redes SDN há uma separação do plano de dados do plano de controle, resultando em elementos de comutação (*switches* e roteadores) com uma interface simplificada e que podem ser controlados por um elemento externo via *software*. Para a implementação das redes definidas por software foi desenvolvido pela Universidade de Stanford o OpenFlow [Costa 2013 apud McKeown *et al.* 2008].

O OpenFlow é um padrão aberto para o paradigma de redes SDN. Ele consiste de uma interface de programação, que permite ao desenvolvedor controlar diretamente os elementos de encaminhamento de pacotes presentes no dispositivo de rede [Costa 2013]. A estrutura desta nova tecnologia é apresentada na Figura 1.

O controlador é o elemento externo responsável por definir o comportamento da rede através do gerenciamento das tabelas de fluxos. Este componente realiza a

abstração dos elementos de comutação, permitindo através de uma linguagem de programação de alto nível a criação de aplicações para agirem sobre a rede. Todos os switches OpenFlow em uma rede se comunicam através do protocolo OpenFlow com o controlador que age de forma centralizado sobre eles, embora, podendo ser implementado de forma distribuída.

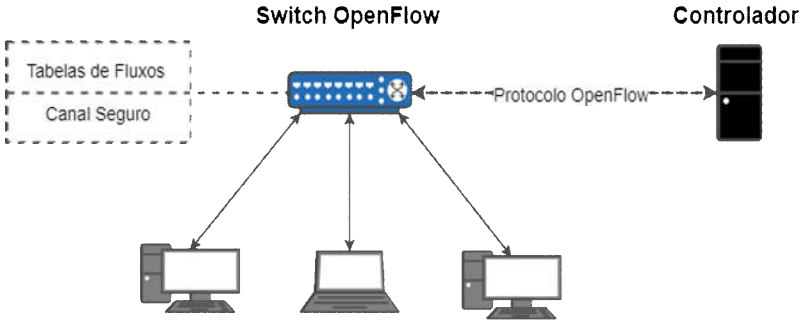

Figura 1. Estrutura OpenFlow [Costa 2013 apud Stanford University].

O *switch* OpenFlow é composto por tabelas de fluxos e um canal seguro. O protocolo OpenFlow não garante a segurança das mensagens, por conta disto, é estabelecido um canal seguro através do protocolo SSL **(***Secure Socket Layer*) e TCP (*Transmission Control Protocol*). As tabelas de fluxos são responsáveis pela verificação de cada pacote trafegado na rede em relação as regras inseridas pelo controlador. Cada regra define como um fluxo de pacotes deve se comportar. Segundo [Costa 2013] uma regra é formada por uma tupla de doze elementos que reúne características dos protocolos da camada de enlace, de rede, e de transporte segundo o modelo TCP/IP. Além disso, toda entrada de fluxo está associada a um conjunto de instruções, contadores, prioridade, intervalo de tempo e *cookies*. Em um *switch* que não possui a tecnologia OpenFlow as regras são definidas apenas pelo endereçamento MAC (*Media Access Control*) e IP (*Internet Protocol*).

Todo pacote é comparado com as entradas de fluxos presentes nas tabelas. Caso haja uma regra compatível com o cabeçalho do pacote, uma ação será aplicada pelo conjunto de instruções. Caso não haja, o pacote é enviado para o controlador determinar o que será realizado. O pacote pode ser enviado como um todo para o controlador ou o *switch* pode armazenar a carga útil e enviar apenas o cabeçalho [Mousavi 2014]. O controlador pode inserir uma nova entrada de fluxo para que o pacote possa prosseguir para o seu destino ou optar por outra ação. Cada entrada em uma tabela pode possuir um intervalo de tempo predeterminado e para cada análise de pacote o campo de contadores é atualizado para a geração de estatísticas dos fluxos. Nota-se, que o controlador é o responsável por determinar o comportamento de cada fluxo trafegado em uma rede definida por *software*. Com isso, ao detectar uma anomalia de rede, pode-se enviar imediatamente uma regra para rejeitar todos os pacotes responsáveis pela anormalidade, evitando o esgotamento dos recursos.

3. Implementação

A solução proposta neste trabalho é constituída de três módulos principais representados na Figura 2. O primeiro módulo proposto é o *Flow Collector* que coleta os fluxos IP dos *switches* OpenFlow. A coleta dos fluxos IP é obtida via protocolo NetFlow[3] v.9 e a exportação é realizada para um computador através da ferramenta Softflowd[4]. A captura dos fluxos exportados foi feita através da ferramenta Nfcapd[5]. A ferramenta utilizada para o processamento dos fluxos foi a Nfdump, gerando fluxos em um formato acessível para o módulo *Network Attack Detector*.

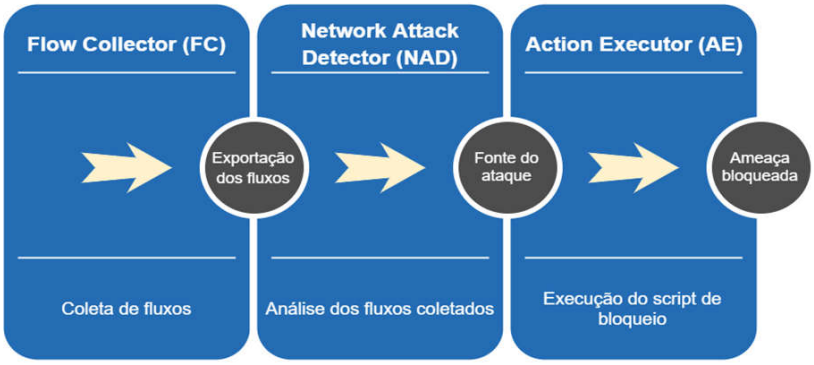

Figura 2. Módulos da solução proposta.

Na etapa seguinte, todos os fluxos IPs coletados são analisados pelo detector desenvolvido por [Amaral *et al.* 2017] com o objetivo de detectar um ataque de rede. Ao detectar um ataque é gerado como entrada para o módulo *Action Executor,* o IP de origem, IP de destino e o tipo de ataque detectado.

O *Action Executor* realiza o bloqueio automático do ataque, utilizando um *shell script* desenvolvido com base no módulo *Static Entry Pusher* presente no controlador (*e.g.*, controlador floodlight). Este módulo é acessado via REST API (*Representational State Transfer Application Programming Interface*) e possui como função principal a consulta, a inserção e a exclusão das entradas de fluxos nas tabelas presentes nos *switches* OpenFlow [Izard 2017].

4. Resultados e discussão

A simulação de uma rede definida por *software* foi realizada utilizando a ferramenta Mininet[6] através de sua interface gráfica Miniedit. Esta aplicação permite a criação e a execução de uma rede SDN de forma simples e rápida. A Figura 3 apresenta a topologia utilizada para a realização dos experimentos. O Mininet utiliza o Open

[3] Tecnologia que permite obter informações sobre o fluxo IP como: data, hora, duração, protocolo, IP de origem, IP de destino, número de pacotes, número de bytes e número de fluxos.
[4] http://www.mindrot.org/projects/softflowd/.
[5] http://nfdump.sourceforge.net/.
[6] http://mininet.org/.

vSwitch[7] para a criação dos *switches* virtuais com a tecnologia OpenFlow. Para o controle dos dispositivos de rede foi escolhido o controlador OpenFlow Floodlight[8].

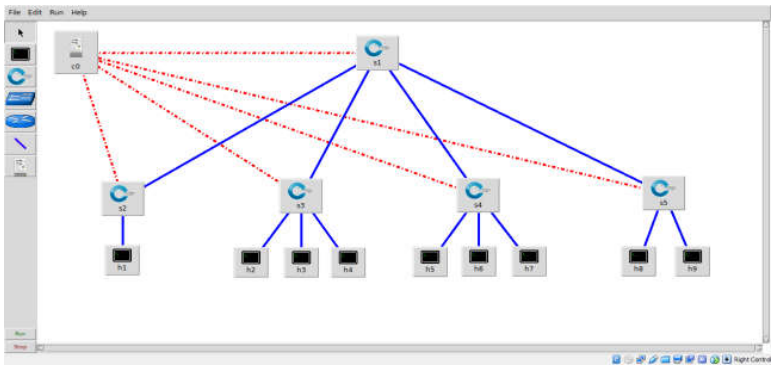

Figura 3. Topologia de rede criada.

A fim de avaliar o funcionamento da solução foi realizado um ataque DoS (*Denial of Service*) do tipo SYN *flooding*, utilizando a ferramenta Hping3[9]. Esta ferramenta foi configurada para enviar `100` pacotes de `64` bytes por segundo para a porta `80` do dispositivo alvo. Esse ataque envolveu dois dispositivos de rede. O primeiro foi o atacante com IP `192.168.0.5` e o segundo o dispositivo alvo com IP `192.168.0.1`.

Todos os fluxos encaminhados pelas máquinas foram exportados corretamente pelo módulo *FC*. Ao ser acionado, o módulo *NAD* detectou o ataque, apresentando uma visão holística da rede e destacando em vermelho os dispositivos envolvidos, como pode ser observado na Figura 4.

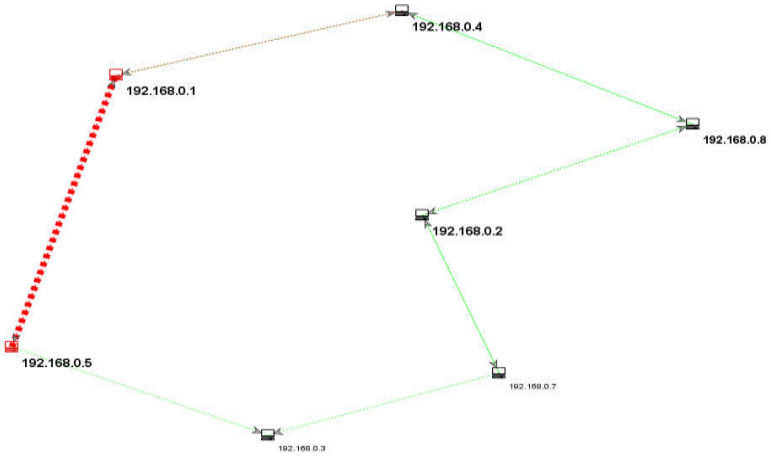

Figura 4. Visão global da rede gerada pelo módulo *NAD*.

[7] http://docs.openvswitch.org/en/latest/.
[8] https://floodlight.atlassian.net/wiki/display/floodlightcontroller/User+Documentation.
[9] http://www.hping.org/.

Após a detecção do ataque DoS, o módulo *AE* executou um *shell script* de bloqueio automatizado, conforme apresentado na Figura 5 (a). O bloqueio foi realizado através do envio de uma entrada de fluxo na tabela do *switch* vinculado ao IP atacante. Esta entrada possui a ação de rejeitar todos os pacotes enviados do IP `192.168.0.5` para o IP de destino `192.168.0.1`, impedindo que a continuidade do ataque culminasse no esgotamento dos recursos do dispositivo alvo.

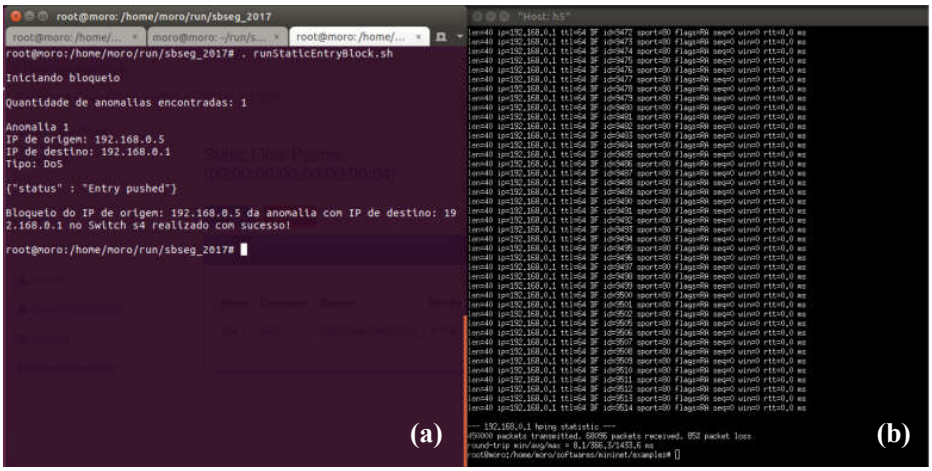

Figura 5. (a) Shell script de bloqueio (b) Ataque DoS bloqueado.

É possível visualizar na Figura 5 (b) que, neste ataque, `450.000` pacotes foram enviados para o alvo. Porém, deste montante, apenas `68.096` pacotes chegaram ao destino, ou seja, `15%`. Um total de `381.904 (85%)` dos pacotes não chegaram ao alvo, mostrando que a solução proposta neste trabalho consegue bloquear o ataque automaticamente, sem a intervenção humana. Os `15%` dos pacotes que obtiveram sucesso ocorreram devido ao tempo demandado na coleta e na exportação dos dados da rede (fluxos IP) para serem analisados pelo detector.

5. Considerações Finais

Este trabalho apresentou uma solução capaz de detectar e aplicar ações de autorreparo de anomalias em uma rede definida por *software*. Através de um estudo de caso inicial utilizando um ataque DoS foi possível validar a solução, destacando a sua capacidade para detectar e interromper a ocorrência da anomalia sem a intervenção humana. A ação de bloqueio executada pelo módulo *AE* mostrou como as ações de reparo automáticas podem ser aplicadas em uma rede SDN utilizando a tecnologia OpenFlow. Com isto, é possível salientar que a segurança e o gerenciamento de rede, neste novo paradigma, são simplificados, reduzindo a complexidade imposta pelas redes atuais em função da heterogeneidade dos equipamentos. Como trabalho futuro outros testes com diferentes ataques serão realizados e a autonomicidade da solução, bem como a sua capacidade de inferir ataques, através do aprendizado de máquina, serão explorados.

Referências

Abranet. (2017) “Internet das Coisas faz ataques DDoS crescerem 138% no Brasil”, http://www.abranet.org.br/Noticias/Internet-das-Coisas-faz-ataques-DDoS-crescerem-138%25-no-Brasil-1531.html?UserActiveTemplate=site#.WabyFciGPIV, Agosto.

Ahmad, I., Namal, S., Ylianttila, M. and Gurtov, A. (2015) “Security in Software Defined Networks: A Survey”, In: IEEE Communications Surveys & Tutorials, vol. 17, no. 4, p. 2317-2346.

Ahmed, M., Mahmood, A. N. and Hu, J. (2016) “A survey of network anomaly detection techniques”, In: Journal of Network and Computer Applications 60, p. 19-31, Austrália.

Amaral A. A. et al. (2017), “Deep IP flow inspection to detect beyond network anomalies”, Computer Communications, vol. 98, p. 80-96.

Conti, M., Dehghantanha, A., Franke, K. and Watson, S. (2017) “Internet of Things Security and Forensics: Challenges and Opportunities”, In: Future Generation Computer Systems, ISSN 0167-739X.

Costa, L. R. (2013) “OpenFlow e o Paradigma de Redes Definidas por Software”, In: Universidade de Brasília, Monografia (Licenciatura em Ciência da Computação), ix, 143 f., Brasília.

Dacier M. C. *et al.* (2017), "Security Challenges and Opportunities of Software-Defined Networking," in IEEE Security & Privacy, vol. 15, no. 2, p. 96-100.

Francis, R. (2017) “Data breaches through wearables put target squarely on IoT in 2017”, http://www.csoonline.com/article/3150881/internet-of-things/data-breaches-through-wearables-put-target-squarely-on-iot-in-2017.html, Agosto.

Izard, R. (2017) “How to add a REST API to a Module”, https://floodlight.atlassian.net/wiki/display/floodlightcontroller/How+to+add+a+REST+API+to+a+Module, Julho.

Macedo, D. F., Guedes, D., Vieira, L. F. M., Vieira, M. A. M. and Nogueira Michele (2015) “Programmable Networks – From Software-Defined Radio to Software-Defined Networking”, In: IEEE Communications Surveys & Tutorials, vol. 17, no. 2, p. 1102-1125.

Mousavi, S. M. (2014) “Early Detection of DDoS Attacks in Software Defined Networks Controller”, In: Thesis (master), Carleton University, Ottawa, Ontario.

Porras, P., Cheung, S., Fong, M., Skinner, K. and Yegneswaran, V. (2015) “Securing the Software-Defined Network Control Layer”, In: Network and Distributed System Security Symposium (NDSS), Sand Diego, California.

Yan, Q., Yu, F. R., Senior member, IEEE, Gong, Q. and Li, J. (2015) “Software-defined networking (SDN) and distributed denial of servisse (DDoS) attacks ind cloud computing environments: a survey, some research issues, and challenges”, In: IEEE Communications Surveys & Tutorials, vol. 18, no. 1, p. 602-622.

