FindFlows: Um detector de ataques Syn-Flooding em Redes Definidas por Software

Eduardo Fox, Vivek Nigam, Iguatemi Fonseca

***Resumo*— O Syn-flooding é um ataque de negação de serviço (DoS) simples de ser realizado, porém tem consequências desastrosas, podendo impossibilitar o acesso a um site ou outros recursos em uma rede. Em Redes Definidas por Software (SDN), ele pode parar uma rede por completo, a partir da negação do serviço do próprio controlador. Este trabalho propõe uma detecção de ataques Syn-flooding, em uma rede SDN, através da medição da variação da quantidade de fluxos em um determinado intervalo de tempo. Para a realização da proposta, foi desenvolvida uma ferramenta chamada FindFlows que exibe os hosts participantes do ataque.**

Palavras-Chave*— *Redes Definidas por Software, DoS, Syn-flooding, Detecção.

***Abstract*— Syn-flooding is a Denial of Service (DoS) attack, which is simple to be performed but has disastrous consequences and could make it impossible to access a site or other resources on a network. In Software Defined Networks (SDN), it can stop an entire network by the denial of the service of the controller. This work proposes the detection of Syn-flooding attacks in an SDN network by measuring the variation of the number of flows in a given time interval. By implementing the proposal, a tool called FindFlows has been developed to display the participants of the attack.**

Keywords*— *Software Defined Network, DoS, Syn-flooding, Detection.

I. INTRODUÇÃO

Com o crescimento da Internet, as informações ficam cada vez mais fáceis de serem acessadas. Assim também mais facilmente se consegue tutoriais ou *scripts* para realizar ataques pelas Redes de Computadores, sem necessidade de ter um grande conhecimento computacional para isso. Dessa forma, ataques de negação de serviço (*Denial of Service* - em inglês), conhecidos como DoS, passaram a ser algo que um internauta pode realizar com um mínimo de conhecimento, podendo atingir as redes ou servidores que encontram-se mais vulneráveis, ocasionando em consequências desastrosas.

Quando um servidor WEB recebe solicitações de conexões, reserva recursos para tratá-las. Porém, centenas ou até milhares de solicitações de estabelecimento de conexões, em um curto intervalo de tempo, fazem esses recursos se esgotarem. Este é objetivo de um tipo de ataque DoS muito conhecido, o *Syn-flooding*. Funciona enviando inúmeras solicitações de conexões TCP com a *flag* SYN para a vítima, fazendo com que o servidor não consiga mais responder as conexões legítimas [1]. Em Redes definidas por *Software* (SDN - *Software Deined Networks*) esse ataque gera um novo problema como é mostrado a seguir.

Eduardo Fox, Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, João Pessoa-PB, Brasil, E-mail: eduardobfox@gmail.com. Vivek Nigam, Iguatemi Fonseca, Universidade Federal da Paraíba, João Pessoa-PB, Brasil, E-mails: vivek.nigam@gmail.com, iguatemi@ci.ufpb.br.

SDN é um novo paradigma às Redes de Computadores. Veio para alterar a administração distribuída da rede para uma administração centralizada e de maior flexibilidade para o administrador da rede [2]. Em Redes Definidas por *Software*, o plano de controle é desacoplado dos ativos e colocado a parte, em um controlador. Assim, toda a rede local passa a se tornar programada através de um ponto central, facilitando a criação de novas abstrações na rede, proporcionando evolução e inovação às redes de computadores [3].

OpenFlow é o protocolo padrão em SDN [4]. Basicamente ele faz a comunicação entre o controlador da rede (plano de controle) com os *switches* (plano de dados). Cada *switch* SDN possui uma tabela de fluxos que é administrada pelo controlador e nessa tabela existem as regras que indicam que ação tomar com cada novo pacote. Para cada pacote que chega ao *switch*, este verifica se já existe uma entrada na tabela de fluxos informando o que fazer com aquele determinado pacote. Caso não exista nenhuma entrada na tabela informando a ação necessária para o novo pacote, o *switch* encapsula o novo pacote dentro de uma mensagem chamada *packet-in*, do protocolo OpenFlow, e encaminha para o controlador, esperando uma resposta sobre qual a ação a ser tomada [1].

Assim, um ataque *Syn-flooding*, pode ocasionar um duplo problema em uma rede SDN. O primeiro é o de negar o serviço de algum servidor da rede, já o segundo, sendo ainda mais grave, é negar o serviço do próprio controlador da rede. Isto se explica porque cada pacote deste ataque é uma nova solicitação de conexão, possuindo cabeçalhos diferentes e, consequentemente, gerando inúmeras mensagens *packet-in* com destino ao controlador da rede.

Inserido neste contexto, este presente trabalho contribui na detecção de um dos tipos de ataque DoS em uma rede SDN, o *Syn-flooding*. Através de uma ferramenta implementada, chamada FindFlows, é possível detectar os *hosts* da rede que possivelmente estejam participando de um ataque. Para a realização de tal contribuição, dentro de tantas características positivas de uma rede SDN, a que mais ajudou foi a flexibilidade da rede, que dá ao administrador uma extensa possibilidade de ações para serem utilizadas em benefício desta.

Na seção II deste artigo é apresentado o funcionamento do ataque Syn-flooding. Na seção III é mostrado a criação e o funcionamento da ferramenta FindFlows. Na seção IV é realizada a descrição dos cenários e testes criados para validar a detecção do ataque. Na seção V é apresentado os resultados e, por fim, na seção VI, as considerações finais.

II. Ataques Syn-flooding em SDN

Conexões TCP são realizadas através de uma metodologia conhecida como *Three-Way Handshake*. Que funciona com o cliente enviando um pacote com a *flag* SYN, o servidor respondendo com um SYN-ACK e ficando em espera do pacote final do requisitante para o estabelecimento da conexão, o ACK [5].

No caso do *Syn-flooding*, o atacante não envia a última *flag* ACK, fazendo o servidor se manter em uma espera interminável, não completando o processo de estabelecimento de conexão. Uma conexão incompleta aberta consome os recursos do servidor. Várias dessas conexões incompletas abertas fará se esgotar os recursos do servidor, que é a consequência do *Syn-flooding* [5].

É típico de um ataque DoS *Syn-Flooding* abrir centenas ou milhares de requisições TCP com a flag SYN. Considerando que para cada nova requisição, é utilizada uma nova porta de origem, um controlador SDN pode também identificar cada nova requisição como um novo fluxo. Assim, um crescimento exorbitante, do quantitativo de fluxos, em um curto intervalo de tempo, pode caracterizar em um ataque deste gênero.

III. Trabalhos relacionados

O AVANT-GUARD [6] faz com que cada *switch* atue como proxy para requisições SYN. Só envia mensagens *packet-in* para o controlador depois que o *switch* SDN estabelece conexão com o requisitante. Desta forma, a solução não permite que o *Syn-flooding* aconteça, devido a este ataque nunca estabelecer uma conexão TCP.

O LineSwitch [7] é semelhante a solução do AVANT-GUARD, devido a cada switch também atuar como proxy para requisições SYN. A diferença é que esta utiliza uma *blacklist* para bloquear tráfego advindo de *host* suspeito.

O OPERETTA [5] também é uma proposta de detecção e mitigação de *Syn-flooding*. Funciona como um módulo instalado no controlador que faz este atuar como um proxy para validar as conexões entre cliente e servidor. Em uma conexão TCP, o cliente envia um SYN para o servidor WEB, mas é o controlador que responde com um SYN-ACK. Se o cliente enviar um ACK, o controlador percebe que se trata de uma conexão legítima e envia um RST para o cliente, liberando a instalação de uma nova regra entre o cliente e o servidor. A solução define um limiar de conexões, não estabelecidas, que cada *host* pode fazer. Ultrapassando o limiar definido, o controlador bloqueia o host por tê-lo como atacante.

O SLICOTS [1] é uma proposta de detecção e mitigação de *Syn-flooding* em uma rede SDN. Funciona com o monitoramento de todo o processo do TCP *Three-way Handshake* entre dois hosts. Durante este processo, é instalada regras temporárias enquanto a conexão não é estabelecida. Só após o estabelecimento das conexões é que é instalada uma regra permanente entre cliente e o servidor. Caso a conexão não chegue a se estabelecer, a solução a denomina de *half-open*. Depois de certa quantidade de *half-open* é instalada uma regra para bloquear futuras tentativas de conexões do host. A tecnologia é implementada em um controlador OpenDayLight.

A *Selective Packet Inspection* [8] é uma proposta colaborativa de detecção de ataques SYN Flooding. Monitora continuamente os fluxos de SYN *packets* com endereços IP diferentes (*IP Spoofing*). Faz isso através de uma base que armazena o IP relacionado a cada porta do switch e, caso encontre um IP diferente do esperado para determinada porta, considera um ataque e bloqueia o *host* malicioso.

O SPHINX [9] é um *Framework* para detecção de ataques em redes SDN. Ele detecta um ataque *Syn-flooding* monitorando a quantidade de mensagens *packets-in*, correspondente as novas requisições de conexão com a *flag* SYN. Se essa quantidade passar de um limiar estabelecido pelo administrador da rede, é gerado um alerta de possível ataque.

Estas soluções demonstram o atual estado da arte para detecção ou mitigação do ataque *Syn-flooding* em SDN. O AVANT-GUARD [6], LineSwitch [7] e OPERETTA [5] são semelhantes por utilizarem *proxy* para requisições TCP. O grande problema da utilização *proxy* está em não garantir o princípio fundamental da Internet, que é permitir conexões fim-a-fim livre de intermediações. Com o uso do *proxy* quem recebe as conexões TCP é o próprio *proxy* e, apenas em seguida, repassa para os *hosts* de destinos, ocasionando em um certo *delay* no estabelecimento de conexões legítimas. O SLICOTS [1] é eficiente na mitigação do *Syn-flooding*, porém requer um alto custo de processamento por abrir o pacote para verificar as *flags* TCP. Por quebrar a flexibilidade do uso de endereço IP em qualquer porta dos *switches*, a solução de inspeção seletiva de pacotes [8] não seria adequada em alguns ambientes, como em um campus universitário que possui laboratórios de informática em que os professores e alunos realizam testes na rede.

A solução mais próxima da proposta deste artigo é o SPHINX [9] por também estabelecer o limiar da quantidade de fluxos, porém faz isso apenas no total de fluxos que chega ao controlador, abrindo margem para falsos positivos em período de alta vazão nas conexões do servidor WEB. Além do que esta solução não identifica quem o atacante e a vítima do *Syn-flooding*.

As diferenças das soluções apresentadas até aqui e a proposta atual deste artigo está em o FindFlows conseguir detectar o atacante e a vítima do *Syn-flooding* ao mesmo tempo que garante a conectividade fim-a-fim sem interceptação. Também não necessita abrir os cabeçalhos dos pacotes para analisar as *flags* TCP, reduzindo o custo computacional. Além disso, a proposta permite a contínua flexibilidade dos dispositivos da rede, sem necessidade de fazer com que cada *host* precise estar de forma permanente em uma porta do *switch*.

IV. Construção da ferramenta FindFlows

A proposta identifica cada *host* da rede e analisa a variação da quantidade de fluxos de cada um em dois instantes diferentes. Em um segundo momento, analisa as portas TCP de destino para identificar se trata de um atacante ou vítima. A seguir é mostrado detalhadamente sobre a implementação da proposta.

Utilizada com o controlador Ryu (versão 3.6), essa tecnologia foi desenvolvida em Python e é utilizada em conjunto com

uma tabela do banco de dados MySQL (5.7), preenchendo-a com os resultados de suas consultas.

A tabela do banco possui 5 colunas: SRC-IP (varchar), Q-FLUXOS-A (int), Q-FLUXOS-D (int), VARIACAO (int), SYN-FLOODING (varchar).

Segue a descrição de cada coluna:

- SRC-IP: possui o IP de Origem capturados através do fluxo;
- Q-FLUXOS-A: possui a quantidade de fluxos de um mesmo IP de origem, no instante da captura;
- Q-FLUXOS-D: possui a quantidade de fluxos de um mesmo IP de origem, 5 segundos depois da captura de Q-FLUXOS-A;
- VARIACAO: possui a variação da quantidade fluxos entre o valor de Q-FLUXOS-A e Q-FLUXOS-D;
- SYN-FLOODING: Irá possuir a palavra *NO* para não suspeito de algum ataque *Syn-flooding*, ATACANTE para o *host* identificado como suspeito de realizar ataque e VITIMA para o *host* que está sendo o alvo do ataque.

O FindFlows realiza os seguintes procedimentos:

A. Captura de todos os fluxos do switch selecionado:

O Open vSwitch permite a captura dos fluxos através do comando "ovs-ofctl -O OpenFlow13 dump-flows". Assim o primeiro procedimento que a ferramenta FindFlows realiza é capturar todos os fluxos ainda armazenados no buffer do switch e colocá-los em um arquivo denominado flows-SW-X, sendo X a identificação do Switch da rede. Vale salientar que esse tempo que os fluxos pode ficar na memória no switch é configurável pelo controlador da rede SDN. O controlador utilizado foi configurado para armazenar os fluxos durante 25 segundos.

B. Separação dos fluxos com endereços IP diferentes da rede local:

Entendendo-se que o administrador de uma rede local sabe qual faixa de endereçamento IP é utilizada em sua rede, o FindFlows separa os fluxos que possuem IPs de origem com prefixos diferentes da rede local, estes são armazenadados em um arquivo a parte. Assim, se houver fluxos suspeitos de realizarem IP *Spoofing* (mascaramento de IP), estarão todos separados para análise futura do administrador. Esse procedimento é realizado através da leitura do IP de origem de cada fluxo, comparando-o com o prefixo estabelecido pelo administrador.

C. Contabilização da quantidade de fluxos por IP de origem:

O FindFlows contabiliza a quantidade de fluxos por cada IP de origem, no instante A e D (Q-FLUXOS-A e Q-FLUXOS-D). Faz isso selecionando o IP de origem de cada fluxo, do arquivo separado no início, e o comparando se ele já existe na tabela do banco de dados. Caso na tabela não exista nenhuma entrada para o IP de origem do fluxo em questão, é criada uma nova linha na tabela com o novo IP de origem e a quantidade de fluxos igual a 1. Caso já exista, é apenas incrementado o valor da quantidade de fluxos daquele IP selecionado. Quando a contabilização é finalizada, a tabela estará preenchida de acordo com a quantidade endereços IP de origem na captura dos fluxos. Em um exemplo, se houver 5000 fluxos de apenas 2 endereços IP, a tabela estará preenchida com apenas 2 linhas, informando a quantidade de fluxos correspondente a cada IP de origem, ou seja, a cada *host* da rede.

D. Armazenamento da variação da quantidade de fluxos:

Nesse nível a tabela estará com quantidade de fluxos A (instante da captura) e D (5 segundos depois de A) preenchidas. Assim a coluna *VARIACAO* será preenchida com a variação da quantidade de fluxos nos dois instantes e esse valor será crucial na detecção do *Syn-flooding*.

E. Categorização do host de origem:

Com a tabela preenchida já com os endereços IP de origem, suas respectivas quantidades de fluxos em instantes diferentes e a variação da taxa de fluxos, será possível detectar o ataque. Para isso, é determinado um valor de *threshold* que limitará a variação da quantidade de fluxos em dois instantes distintos. O limite escolhido foi de 1000 fluxos, devido a esse valor apenas ser alcançado nos testes de ataques realizados, não chegando a ser um valor de variação alcançado quando se utilizou apenas tráfego legítimo, também pode ser adaptado de acordo com o perfil de cada rede. Desta forma, se a variação da quantidade de fluxos for maior que o valor do *threshold*, será realizada uma segunda verificação para saber se o host trata-se um atacante ou uma vítima. Faz isso analisando se os fluxos deste *host* possuem uma única porta TCP de destino, em caso positivo este é classificado como um atacante e a vítima também é descoberta pela análise do endereço IP de destino dos fluxos do atacante. Nos casos em que a segunda verificação constata que o *host* possui portas TCP de destinos diferentes, este é identificado como vítima, visto que uma vítima de um ataque *Syn-flooding* tenta responder as milhares de requisições de conexões advindas com portas de origem diferentes.

V. CENÁRIOS DE TESTES

Utilizou-se do emulador Mininet para a emulação de uma rede SDN real. A escolha do Mininet foi considerada como mais viável devido a possuir as características cabíveis e aplicáveis ao cenário proposto: flexibilidade, interatividade e escalabilidade [10]. Este software ainda traz uma realidade mais próxima do real, por utilizar o próprio *kernel* para emular switches e hosts da rede, não sendo uma simulação, mas uma emulação de um cenário [10].

Dentro do Mininet, foram utilizadas as seguintes tecnologias:

- **Apache2:** Servidor Web escolhido para ser utilizado no cenário. Foi realizada a escolhada do Apache devido a ser o servidor responsável pelo maior número de hospedagem de páginas WEB ativas no mundo, estando com uma porcentagem de 45,78%, seguida de 19,60% do NGINX, segundo uma pesquisa realizada em fevereiro de 2017 pela netcraft[1] .

[1]https://news.netcraft.com/archives/2017/02/27/february-2017-web-server-survey.html Acessado em: 12/01/2018

- **Hping3:** Essa ferramenta permite fazer um *flood* de conexões TCP com a flag SYN, abrindo milhares de conexões em poucos segundos. Foi utilizada a versão 3.0.0 para realizar o ataque *Syn-Flooding*. Através desta ferramenta, foi simulado o atacante.
- **Siege:** Ferramenta utilizada para simular o tráfego de clientes legítimos, possibilitando simular requisições HTTP partindo de uma quantidade de clientes pré-determinada pelo usuário. A versão utilizada foi a 3.0.5.

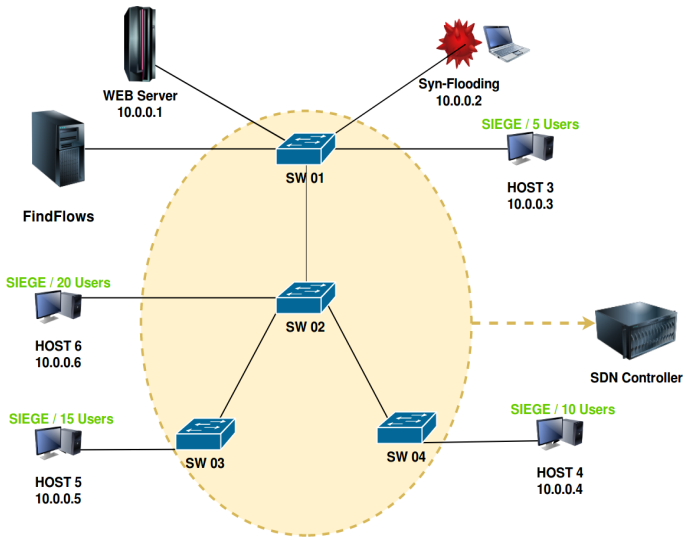

Fig. 1. Cenário principal

Para a validação do teste, foram implementadas 3 variações de testes, todas derivando do cenário principal, esboçado na Figura 1. Assim, a rede foi composta por um Servidor Web, um atacante e quatro *hosts*, simulando clientes legítimos, distribuídos em 4 *switches*. Todo o tráfego dos *hosts* e do atacante tiveram, como destino, o Servidor WEB, com IP 10.0.0.1.

Foram realizados 3 tipos de testes, alterando a duração de tráfego comum e ataque.

- **1º Teste:** 10 segundos de tráfego legítimo (Siege) + 10 segundos de Ataque *Syn-flooding*.
- **2º Teste:** 25 segundos de tráfego legítimo (Siege) + 25 segundos de Ataque *Syn-flooding*.
- **3º Teste:** 50 segundos de tráfego legítimo (Siege) + 50 segundos de Ataque *Syn-flooding*.

Como cada fluxo passa apenas 25 segundos na tabela de fluxos de cada *switch* (sendo esse tempo configurável no controlador), os testes acima englobam a grande maioria das situações possíveis: antes de zerar os fluxos (1º teste), depois de zerar os fluxos (2º teste) e um múltiplo do 2º teste (3º teste), tendo a intenção de representar a maioria das situações de tráfego legítimo e ataque.

Em todos os testes, a ferramenta FindFlows foi executada logo em seguida do tempo do ataque, com o objetivo de detecção do *Syn-flooding*. Cada teste foi realizado 5 vezes, totalizando em 15 amostras de resultados. A próxima seção descreve detalhadamente os resultados destes testes.

VI. Resultados

A ferramenta da proposta, FindFLows, preenche uma tabela *MySQL* para exibição dos resultados. Os dados das tabelas seguintes correspondem a uma das amostras, do *switch* 1, realizada no 1º teste.

TABELA I

Amostra do 1º teste - *switch* 1

SRC-IP	Q-FLUXOS-A	Q-FLUXOS-D	VARIACAO	SYN-FLOODING
10.0.0.3	127	0	127	NO
10.0.0.1	997	305	692	VITIMA
10.0.0.4	197	0	197	NO
10.0.0.5	287	0	287	NO
10.0.0.6	387	0	387	NO
10.0.0.2	0	3686	3686	ATACANTE

Na coluna ”Q-FLUXOS-A” é mostrado a quantidade de fluxos no instante 0. Já a coluna Q-FLUXOS-D contém a quantidade de fluxos 5 segundos depois. Em ”VARIACAO” encontra-se a variação dos valores em ”Q-FLUXOS-A” e ”Q-FLUXOS-D”, este valor é o que é utilizado para a detecção do *Syn-flooding*. A tabela mostra que foi detectado o *host* atacante e a vítima.

TABELA II

Amostra do 2º teste - *switch* 1

SRC-IP	Q-FLUXOS-A	Q-FLUXOS-D	VARIACAO	SYN-FLOODING
10.0.0.5	9	0	9	NO
10.0.0.1	121	428	297	VITIMA
10.0.0.3	1	0	1	NO
10.0.0.4	11	0	11	NO
10.0.0.6	5	0	5	NO
10.0.0.2	1838	3563	1725	ATACANTE

Aumentando o tempo de ataque antes da execução do *FindFlows*, percebe-se que a quantidade de fluxos processados diminuiu já em ”Q-FLUXOS-A”. Também foi detectado o *host* atacante.

TABELA III

Amostra do 3º teste - *switch* 1

SRC-IP	Q-FLUXOS-A	Q-FLUXOS-D	VARIACAO	SYN-FLOODING
10.0.0.2	1904	3600	1696	ATACANTE
10.0.0.1	91	391	300	VITIMA

Atacando por mais tempo (3º teste, 50 segundos), foi verificado que os fluxos processados foram apenas do atacante e da vítima, devido ao ataque atingir o seu objetivo de negar o serviço para as conexões legítimas dos outros *hosts*.

A ferramanta FindFlows também foi executada sem realizar o ataque, a fim de verificar se realmente não seria detectado nenhum *host* como atacante incorretamente. Segue o resultado:

TABELA IV

Teste sem atacante

SRC-IP	Q-FLUXOS-A	Q-FLUXOS-D	VARIACAO	SYN-FLOODING
10.0.0.3	100	205	105	NO
10.0.0.4	210	359	149	NO
10.0.0.1	997	1994	997	NO
10.0.0.6	372	844	472	NO
10.0.0.5	316	589	273	NO

Conforme esperado, os *hosts* que não estavam envolvidos no *Syn-flooding*, sendo atacante ou vítima, não foram detectados como tal.

Como cada teste foi realizado por 5 vezes, 5 amostras para cada teste foram geradas do valor da variação de quantidade de fluxos. Assim, foi calculado a média aritmética dessas amostras, para cada *host*, em cada um dos testes. Os dados foram armazenados em um gráfico histograma, para melhor visualização e comparação dos resultados obtidos (ver Figura 2).

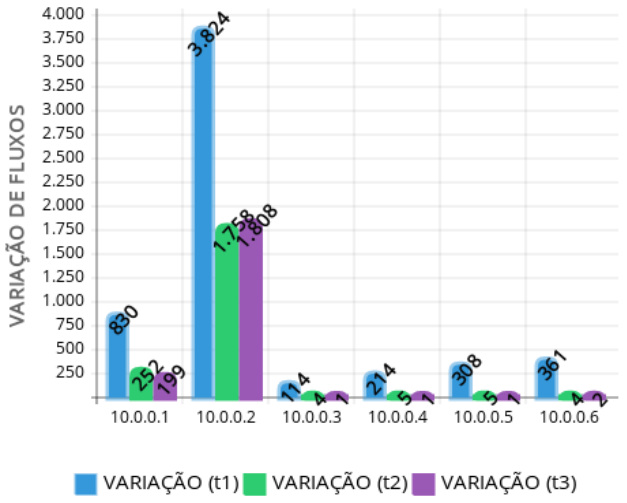

Fig. 2. Comparação do valor da variação da quantidade de fluxos para cada *host*

Em azul, encontra-se a média dos valores da variação da quantidade de fluxos para o 1º teste. Já na cor verde e roxa, para o 2º e 3º teste, respectivamente. Foi observado que os maiores valores dessa variação vem do IP do atacante (10.0.0.2), acompanhado logo em seguida do IP da vítima (10.0.0.1). Os valores da variação da quantidade de fluxos do IP da vítima também são altos devido a receber a grande quantidade de pacotes, tentando responder a cada um (não conseguindo responder a todos, por ter o serviço negado depois de um tempo), gerando também uma alta quantidade de fluxos. O tempo médio de detecção do ataque foram 3 minutos e 9 segundos.

VII. Considerações finais

Um ataque do tipo *Syn-flooding* tem o objetivo de ocasionar a negação do serviço de um Servidor de conexões TCP, mais comumente utilizado para atacar serviços WEB. Porém, conforme descrito na introdução deste trabalho, em uma rede SDN esse ataque pode ter consequências ainda maiores por ter a possibilidade de negar o serviço do próprio controlador, através das inúmeras mensagens *packet-in* geradas pela quantidade de requisições diferentes para o servidor, confirmando assim a necessidade da proposta deste trabalho.

Assim, a ferramenta FindFlows foi desenvolvida para a detecção do *host* autor do ataque e a vítima do *Syn-flooding* em uma rede SDN. Mostrando ao administrador da rede uma tabela com todos os *hosts* da rede, a quantidade de fluxos em dois instantes diferentes, a variação destes dois instantes e se houve ou não ataque *Syn-flooding*. Os *hosts* que possuem endereço IP não correspondente a subrede local, tem seus fluxos separados em um arquivo a parte por serem suspeitos de utilizarem IP *Spoofing*, possibilitando ao analista da rede uma análise mais cautelosa destes fluxos, em um tempo posterior.

Para trabalhos futuros sugere-se a criação de um armazenamento de fluxos suspeitos para que o próprio administador da rede possa analisar estes posteriormente e entender melhor as características dos fluxos do ataque. Também se recomenda o desenvolvimento de uma *blacklist* para armazenar IPs de *hosts* suspeitos de realizarem *IP Spoofing*, tendo o objetivo de fazer a mitigação de ataques que tenham os endereços IP da *blacklist* como origem.

Referências

[1] Reza Mohammadi, Reza Javidan, and Mauro Conti. Slicots: An sdn-based lightweight countermeasure for tcp syn flooding attacks. *IEEE Transactions on Network and Service Management*, 2017.

[2] Yang Xu and Yong Liu. Ddos attack detection under sdn context. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, pages 1–9. IEEE, 2016.

[3] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2015.

[4] Haoyu Song. Protocol-oblivious forwarding: Unleash the power of sdn through a future-proof forwarding plane. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 127–132. ACM, 2013.

[5] Silvia Fichera, Laura Galluccio, Salvatore C Grancagnolo, Giacomo Morabito, and Sergio Palazzo. Operetta: An openflow-based remedy to mitigate tcp synflood attacks against web servers. *Computer Networks*, 92:89–100, 2015.

[6] Seungwon Shin, Vinod Yegneswaran, Phillip Porras, and Guofei Gu. Avant-guard: Scalable and vigilant switch flow management in software-defined networks. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 413–424. ACM, 2013.

[7] Moreno Ambrosin, Mauro Conti, Fabio De Gaspari, and Radha Poovendran. Lineswitch: Efficiently managing switch flow in software-defined networking while effectively tackling dos attacks. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 639–644. ACM, 2015.

[8] Tommy Chin, Xenia Mountrouidou, Xiangyang Li, and Kaiqi Xiong. Selective packet inspection to detect dos flooding using software defined networking (sdn). In *Distributed Computing Systems Workshops (ICDCSW), 2015 IEEE 35th International Conference on*, pages 95–99. IEEE, 2015.

[9] Mohan Dhawan, Rishabh Poddar, Kshiteej Mahajan, and Vijay Mann. Sphinx: Detecting security attacks in software-defined networks. In *NDSS*, 2015.

[10] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM, 2010.