BrainChain - A Machine learning Approach for protecting Blockchain applications using SDN

Zakaria Abou El Houda [1,2], Abdelhakim Hafid [1] and Lyes Khoukhi [2]

[1] Department of Computer Science and Operational Research, University of Montreal, Canada
[2] ICD/ERA, University of Technology of Troyes, France
zakaria.abou.el.houda@umontreal.ca, ahafid@iro.umontreal.ca
{zakaria.abou_el_houda, lyes.khoukhi}@utt.fr

Abstract— **Nowadays, blockchain technology is seen as one of the main technological innovations to emerge since the advent of the internet. Many applications can benefit from blockchain to protect their exchanges. Nonetheless, applications with more restricted interests cannot use public blockchains. Permissioned blockchains promise to combine effectiveness of blockchains with stricter permissions to join blockchain's network. In permissioned blockchain, the number of participating entities is limited compared to public blockchain. However, by targeting the peers of the blockchain, the attackers can easily take control of consensus process and halt the blockchain operations. In this paper, we propose BrainChain, a scalable and efficient scheme to protect permissioned blockchain nodes from the largest ever Distributed Denial of Service (DDoS) attack (i.e., Domain Name System (DNS) amplification attack) in the context of software defined networks (SDN). BrainChain consists of 4 schemes: (1) Flow statistics collection scheme (FS) to gather the features of flows in an efficient way using sFlow; (2) Entropy based scheme (ES) to measure disorder of network features; (3) Bayes Network based Filtering scheme (BF) to classify, based on entropy values, illegitimate DNS requests; and (4) DNS Mitigation (DM) scheme to mitigate in an effective way the illegitimate flows (i.e., illegitimate DNS requests). Experimental results show that BrainChain can quickly and effectively detect and mitigate the attacks (i.e., DNS amplification attacks) with a high accuracy and a small false positive rate making it a promising scheme to protect blockchain applications from DNS Amplification attacks.**

Keywords— DDoS ; DNS Amplification ; SDN ; Bayes Classifier ; Entropy ; Blockchain ;

I. Introduction

DDoS attacks are considered serious network security threats. They have evolved to be destructive and powerful causing severe damage to network operators as well as Internet service providers (ISPs). DNS amplification attack is one of the most devastating types of DDoS attacks; on October 2, 2016, a huge attack was conducted against the servers of Dyn, a company that controls many Internets DNS servers. As a consequence, many popular Internet services, e.g., Amazon, Twitter, GitHub [1], PayPal and others became unavailable for several hours [2]. This DNS amplification attack [2] is considered as the largest ever DDoS attack, exceeding a rate of 1 Tbit/s. Such incidents harm ISPs and cost millions of dollars of lost revenues for enterprises.

The continuous growth in complexity and size of current networks, such as enterprise networks and data centers, is giving rise to SDN as a novel technology that facilitates network management and provides new approaches to deploy and manage networks dynamically [3] – [4]. SDN separates data and control planes; this separation allows for more control over the network and brings a new way to deal with DDoS attack (e.g., DNS amplification attack).

In this paper, to protect blockchain's nodes from DDoS attack, we design a scalable and efficient scheme based on SDN, named BrainChain. SDN brings various benefits by decoupling the control plane from the data plane. While SDN brings new capabilities to protect from various forms of DDoS attacks [5-6], it can be targeted by these attacks [7]. To address this issue, BrainChain makes use of an efficient information collection scheme, called FS, to gather the features of flows in an efficient way using sFlow [8] protocol. Then, it uses an entropy-based scheme (ES) to measure network feature's changes; and it uses machine learning algorithm, called BF, to automatically detect network anomalies. By combining these schemes (i.e., FS, ES and BF), DNS amplification attack can be effectively/accurately detected then mitigated using DNS Mitigation (DM) scheme. BrainChain protects blockchain's nodes without modifying software neither of the nodes nor of the blockchain.

The contributions of this paper are as follow:

- We propose a scalable flow statistics collection scheme (FS) to gather the features of flows in an efficient way using sFlow. FS highly reduces exchanges between data plane and control plane.
- We introduce an Entropy calculation scheme (ES) to measure data's disorder/randomness.
- We design a real-time detection scheme, called Bayes Network based Filtering scheme (BF), to automatically detect network anomalies.
- We propose DNS Mitigation (DM) module to quickly /effectively mitigate illegitimate flows (i.e., DNS requests).
- We evaluate the performances of BrainChain in terms of efficiency, effectiveness and scalability. BrainChain detection accuracy is evaluated through Receiver Operating Characteristic (ROC) curves and compared to the most prominent state-of-art schemes. Experiments results show that BrainChain can quickly and effectively detect and mitigate the attacks (i.e., DNS amplification attacks) with a high accuracy and a small false positive rate.

The rest of this paper is organized as follows. In Section II, we present related work. In Section III, we introduce the system design. Section IV presents our *machine learning DDoS detection module*. Section V describes *DM scheme*. Section VI evaluates BrainChain. Finally, we conclude the paper in Section VII.

II.RELATED WORK

Blockchain technology is considered as new technology to secure and store information in fully decentralized and automated way without any trusted third party. Due to limited number of peers (i.e., blockchains nodes) of permissioned blockchains, the security and availability of peers need to be more investigated. Our previous works [5-6] show their effectiveness in protecting from DRDoS attacks (Distributed reflection denial of service attacks), in this work we consider a typical DNS Amplification attacks. In [9], we used an entropy scheme in order to protect permissioned blockchain from DNS Amplification attacks. However, it has considerable false positive Rate. In this paper, we use machine learning algorithm (i.e., BF) in order to decrease false positive Rate while maintaining a high detection Rate. In [10], *Mathis et al*. proposed an OpenFlow (OF)-based firewall to ensure security to blockchain nodes. They implemented their solution as module, in SDN controller, that uses SDN functionalities to filter networks traffic. However, a high flows rate from data plane to control plane may overload the SDN controller. In [11], Rodrigo et al. proposed a flow-based detection scheme using OF protocol in order to gather flow statistics. This scheme [11] focuses only on DDoS attacks in data plane without any analysis of the overhead to the control plane caused by flow table entries collecting process. Moreover, the performance analysis does not include the overall system performance. In [12] Mehdi et al. proposed a scheme that consists of four prominent traffic intrusion detection algorithms in SDN's context. However, the scheme was designed for small-scale setup as it focused on the home environment; in large-scale setup, high traffic's rate from OF based switches to control plane may overload the SDN controller. In [13], Yu et al. proposed OpenSketch, a software defined traffic measurement scheme. In order to count traffic in an effective manner, OpenSketch implements a hash-based data structure in OF based switches. OpenSketch provides an efficient scheme to collect measurement data through a three-stage pipeline (hashing, filtering, and counting). However, OpenSketch relies on sending all the counters to the controller for analysis which may overload the SDN controller. In [14], Wang et al. proposed an entropy-based flow statistics collection scheme in OF switches; it focuses only on anomaly detection, without nor find the target (i.e., victim of attack) nor the illegitimate nodes to, eventually, block them. In [15], K. Giotis et al. proposed the use of the sFlow protocol with OF protocol to detect DDoS attacks reducing the communication overhead between OF switches and the SDN controller. This scheme [15] works well; nonetheless, it has high false positive Rate. In [16], Lim et al. proposed a DDoS blocking scheme that runs on SDN controller; this scheme requires a large amount of communications between the data and control plane to protect the target. This scheme [16] not only causes DDoS attacks against the control plane but also requires high latency to cooperate with SDN controller. *Zaalouk et al.* [17] proposed a scheme that uses SDN to mitigate DNS amplification attacks; it uses sFlow to monitor DNS traffic. This solution [17] may degrade the performance of SDN controller and allow for DDoS attack against SDN controller itself.

To address the weaknesses of these existing solutions [9-17], we propose BrainChain, a scalable and an efficient mitigation scheme to detect and mitigate DNS amplification attack. BrainChain employs sFlow protocol to separate flow monitoring from the forwarding logic; this makes it much more scalable compared to existing native OF schemes [9-14] and [16]. And using machine learning algorithms, our scheme is much accurate in comparison with the ones using sampling technology [15-17]. The REST [18] API is used in the process of detection/mitigation to manage any SDN controller to mitigate illegitimate flows (i.e., illegitimate DNS requests) based on DM scheme.

III.SYSTEM DESIGN

A. Design Overview

When designing BrainChain, we did consider the following objectives. First, BrainChain should ensure full protection to permissioned blockchain nodes and should be more accurate and sensitive to DNS amplification attack in order to ensure real-time detection. In addition, to protect Ternary Content Addressable Memory (TCAM) of data plane devices (i.e., OF based switches) that is limited in size, BrainChain uses an accurate detection scheme (i.e., BF). Finally, the whole system should be as scalable as possible, and the attack has to be effectively mitigated.

B. System Architecture

The architecture of BrainChain consists of four main schemes (see Fig.1): (1) FS, a novel Flow statistics collection scheme to gather the features of flows in an efficient way using sFlow; (2) Entropy calculation scheme (ES); (3) a real-time detection scheme (BF), to automatically detect network anomalies; and (4) DNS mitigation scheme (DM).

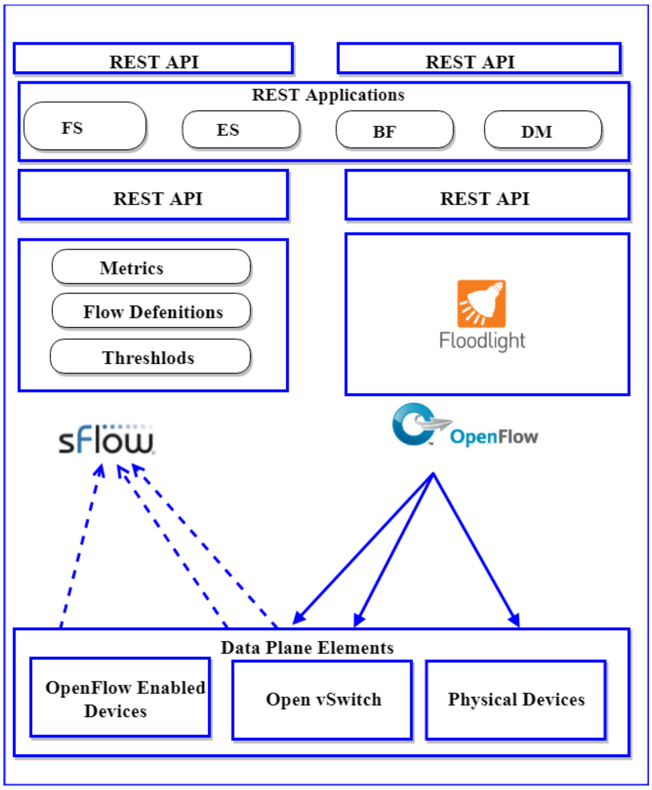

Fig.1. System Architecture

BrainChain has two phases: (1) *a machine learning DDoS detection module*. This module aims to detect, in real-time, illegitimate flows (i.e., illegitimate DNS requests). This module consists of FS scheme, ES scheme and BF; it is running as application on the top of the SDN controller (i.e., application layer) and using sFlow, FS collects networks traffic statistics, ES and BF detect automatically illegitimate flows; and (2) *DNS mitigation (DM) scheme*. This scheme aims to mitigate illegitimate flows effectively to enable the network to recover quickly.

IV. *MACHINE LEARNING DDOS DETECTION MODULE.*

The procedure of our machine learning DDoS detection module consists of the following (1) FS to gather the features of flows in an efficient way using sFlow protocol; (2) ES to extract network features; and (3) BF to detect attacks.

1. FS

In SDN environment, there are two used methods to collect networks traffic statistics. The first one is based on OF protocol. In this method, control plane sends a feature request (*ofp_flow_stats_request*) to the data plane devices (i.e., OF switches) that respond with one or more messages (*ofp_flow_stats_reply*) (i.e., flow table content). This method can gather the overall flow information of data plane's traffic. However, a high rate DDoS attack traffic sent by OF switches to the SDN controller may on one hand, exhaust the bandwidth between data plane and control plane (i.e., SDN controller), on the other hand, may exhaust TCAM of data plane devices. Therefore, this method is not suitable to detect high rate attack. To address this issue, FS uses flow sampling technologies using sFlow. The sFlow collection process separates the flow monitoring from the forwarding logic. FS is more efficient and scalable since the access to packet counters per flow is eliminated. sFlow performs flows aggregation required during a high-rate DDoS attack. FS collects packet sampling and updates the counter of each flow entry at each monitoring interval. sFlow collector (sFlow-RT [19]) receives networks traffic samples from sFlow agents and updates flow's counters. Hence, there is no need to maintain flow entries in OF switches and compare flow statistics of every monitoring interval. Then, periodically, ES extracts the network features from collected information and calculates the entropy values. In what follows, we describe the feature extraction method using ES scheme.

2. ES

The purpose of ES is to extract the network features, from the collected information, and calculates entropy values. The concept of entropy calculation was introduced in information theory [20] to measure the randomness change of incoming flows in a given time window. When the blockchain's network is under DDoS attacks, number of flows for a given flow will increase sharply causing a more concentrated distribution of IP source address (e.g., target blockchain node). High entropy values lead to a more dispersed distribution of the probability of the IP source address, while low entropy values mean the concentration of distribution of IP source address. A flow can be defined as sequence of packet that have same properties. If a source sends similar requests (e.g., requests from the same IP_{src} , with the same type of DNS request (e.g., "ANY" requests) and from the same UDP source port Src_{port}), then its Entropy will be low; this means that we are very likely in the presence of illegitimate flows. Therefore, we use ES to measure changes in traffic information (e.g., IP_{src} , "ANY" requests, UDP Src_{port}) during each monitoring interval ΔT.

In this work, we define each flow as a five-tuple: $\{ IP_{src} , IP_{dst} \ , Src_{port} , Dst_{port} \ = 53, proto \ = UDP\}$. In the reminder of this paper, we use I as a set of positive integers and R as a set of real numbers.

Let $F_{i,j}$ denote the flow f_i at local OF switch S_j; it is defined as follows:

$$F_{i,j}\left(IP_{src_i}, S_\mathrm{j}\right) = \{<IP_{src_i}, S_\mathrm{j}, t>|S_\mathrm{j} \in S \ , i,j \in I, t \in R\} \quad (1)$$

$IP_{src_i}, S_\mathrm{j}, t , i,j \in I, t \in R,$ where IP_{src} is the source IP address of f_i, t is the current timestamp, and $S = \{S_\mathrm{j}, j \in I \}$ is the set of OF switches.

Let $|F_{i,j}(IP_{src_i}, S_\mathrm{j}, t)|$ be the count number of packets of flow $F_{i,j}$ at time t . We define the variation of the number of packets for flow f_i at a local OF switch S_j given flow during ΔT as follows:

$$N_{F_{\mathrm{i,j}}}\left(IP_{src_i}, S_\mathrm{j}, t \ + \Delta T\right) = \left|F_{i,j}\left(IP_{src_i}, S_\mathrm{j}, t + \Delta T\right)\right| - |F_{i,j}(IP_{src_i}, S_\mathrm{j}, t)| \quad (2)$$

The probability $p_{i,j}$ of flow f_i over all the flows at local OF switch S_j can be expressed as follows:

$$p_{i,j}(IP_{src_i}, S_\mathrm{j}) = \frac{N_{F_{\mathrm{i,j}}}\left(IP_{src_i}, S_\mathrm{j}, t \ + \Delta T\right)}{\sum_{i=1}^{N} N_{F_{\mathrm{i,j}}}} \quad (3)$$

where $\sum_{i=1}^{N} p_{i,j}(IP_{src_i}, S_\mathrm{j})$=1 and N is the total number of flows at local OF switch S_j

Let IP_{src} be random variable that denotes the number of flows during time interval ΔT. We define the entropy of flow f_i at local OF switch S_j, duringΔT, as follows:

$$H(IP_{src}) = -\sum_{i=1}^{N} p_{i,j}\left(IP_{src_i}, S_\mathrm{j}\right) log_2\, p_{i,j}(IP_{src_i}, S_\mathrm{j}) \quad (4)$$

Lemma: The lower bound and upper bound of $H(IP_{src})$ *are 0 and* log_2 *N respectively (see inequality (5)).*

$$0 \leq H(IP_{src}) \leq log_2 N \quad (5)$$

Proof. Let $f(x) = \log x,\ x \geq 0$. We know that $f(x)$ is a monotonically increasing concave function. Let X be random variable for the flow distribution. Applying Jensen's inequality [21] to $f(x)$, we have $E(f(x)) \leq f(E(x))$. Let $P(X) = \{p_1, p_2, p_3, p_4 \ldots \ldots p_N\}$ be the distribution of flows at an OF switch.

Then, $\forall\, p_k, 0 \leq p_k \leq 1, \sum_{k=1}^{N} p_k = 1$, $\sum_{k=1}^{n} p_k \ f(x_k\ ) \leq f(\sum_{k=1}^{n} p_k \ x_k)$ More specifically,

$$H(IP_{src}) \ = \sum_{k=1}^{N} p_k log_2 \frac{1}{p_k} \leq log_2\left(\sum_{k=1}^{N} p_k \ \frac{1}{p_k}\right) = log_2(N) \quad (6)$$

Since $log_2 \frac{1}{P_k} \geq 0, \forall\, p_k, 0 \leq p_k \leq 1$, then, $H(IP_{src}) \ \geq 0$ (7)

□

Then, from inequalities (6) and (7), we obtain the inequality:

$$0 \leq H(IP_{src}) \leq log_2 N$$

The normalized Entropy values are in [0, 1] and are defined as follows:

$$H'(IP_{src}) = \frac{H(IP_{src})}{log_2\, N} \qquad (8)$$

The attribute (i.e., header fields of the packet) to aggregate flows depends on the attack scenario under consideration. Many attackers use a script to launch the attack; each attacker sends multiple DNS requests from the same UDP port source number with the same DNS request type (i.e., "ANY"), as a legitimate client sends requests from random UDP port source number with different type (e.g., A, MX, NS, etc.). Consequently, we use $\{IP_{src}, Src_{port}, \text{"ANY"}\}$ as the attribute to aggregate flows. Similarly, we define the UDP port source Src_{port} entropy $H'(Src_{port})$ and DNS request type entropy $H'(\text{ANY})$. Finally, we represent the network features at the k^{th} time period as:

$$\mathrm{X_k} = \left\{H'(IP_{src})_{\mathrm{k}}, H'\left(Src_{port}\right)_{\mathrm{k}}, H'(\mathrm{ANY})_{\mathrm{k}}\right\} \qquad (9)$$

3. BF

BF can be seen as a binary classifier in the field of machine learning; it uses stateful features (i.e., the network feature vector $\mathrm{X_k}$) to classify the current flow as legitimate or illegitimate. At the k^{th} time period ΔT, this module receives the vector $\mathrm{X_k}$, then it classifies the flow according to the probability of illegitimacy which is calculated via the Bayesian filter (see Section 3.b). If the probability of illegitimacy in the time interval ΔT is above a fixed threshold, then BF notifies the SDN controller in order to deploy the mitigation scheme against this illegitimate flow. In the following section, we detail BF which aims at detecting abrupt behaviors of attackers that try to overload the network with illegitimate DNS traffic.

a) Flow representation

In BF, each sample is characterized by the vector $x = (x_1, x_2, x_3)$ where x_1, x_2, x_3 are the values respectively taken by random variables $H'(IP_{src}), H'\left(Src_{port}\right)$ and $H'(\mathrm{ANY})$. Each of these random variables indicates, respectively, entropy values of IP source address, UDP port source and ANY type of DNS request.

b) Criterion of classification

We consider two classes of requests: illegitimate and legitimate requests denoted by $illeg$ and leg, respectively. The class of flow x, denoted by c, is the one which the probability of being in this class noted $p(c/x\,)$ is maximal.

$$c = argmax_{c\epsilon\{illeg,leg\}}\, p(c/x\,)$$

We have $p(illeg/x\,) + p(leg/x\,) = 1$; thus, the selection criterion can be defined as follows:

$$x\ is\ illegitimate\ iff\ p(illeg/x\,) > .5$$

Classification: According to the Bayes theorem [22] [23] and the total probabilities theorem, the probability of flow $x = (x_1,x_2,x_3)$ to belong to class c is defined as:

$$p(C\ = c\,/\,\mathrm{X} = \mathrm{x}\,) = \frac{p(C = c).p(\,\mathrm{X} = \mathrm{x}\,/C = c)}{p(\,\mathrm{X} = \mathrm{x})} \qquad (10)$$

Using the theorem of the total probability, we conclude:

$$p(C\ = c\,/\,\mathrm{X} = \mathrm{x})\ = \frac{p(C=c).p(\,\mathrm{X=x}\,/C=c)}{\sum_{c\epsilon\{illeg,leg\}} p(C=c).p(\,\mathrm{X=x}\,/C=c)} \qquad (11)$$

Thus, the selection criterion is equivalent to:

x is illegitimate if and only if

$$p(C\ = c/\mathrm{X} = \mathrm{x}) = \frac{p(C=c).p(\,\mathrm{X=x}\,/C=c)}{\sum_{c\epsilon\{illeg,leg\}} p(C=c).p(\,\mathrm{X=x}\,/C=c)} > .5\ (12)$$

c) Bernoulli distribution

Bernoulli distribution is a discrete probability distribution which takes the value 1 with the probability p and 0 with the probability $q = 1 - p$. In other words:

$$P(X = x) = \begin{cases} p & if\ \ x\ =\ 1, \\ 1\ -\ p & if\ \ x\ =\ 0, \\ 0 & otherwise. \end{cases}$$

It can be presented as follows:

$$p\ (X\ = x)\ =\ p^x\ (1-p)^{1-x} 1_{\{0,1\}}(x).$$

The average and variance are given by the formula:

$$E(X) = p\ \ , Var\ (X)\ =\ p.(1\ -\ p).$$

d) Classification

$H'(IP_{src}), H'\left(Src_{port}\right)$ and $H'(\mathrm{ANY})$ are conditionally independent given category c (see Section 3.a). Let $p_i(ill)$ and $p_i(leg)$ denote the conditional probabilities that the incoming flow f_i, during ΔT, is illegitimate and legitimate, respectively. By using Eq. (12), flow x is illegitimate if and only if (in case of Bernoulli classifier):

$$\frac{\prod_{i=1}^{n} p_i^{x_i}(ill)\left(1-p_i(ill)\,\right)^{(1-x_i)}\ p(ill)}{\prod_{i=1}^{n} p_i^{x_i}(ill)\left(1-p_i(ill)\,\right)^{(1-x_i)}\ p(ill)+\prod_{i=1}^{n} p_i^{x_i}(leg)\left(1-p_i(leg)\right)^{(1-x_i)}\ p(leg)} > .5\ \ (13)$$

When $p(ill) = p(leg)$, the selection criteria become as follows: The flow x is illegitimate if and only if:

$$\frac{\prod_{i=1}^{n} p_i^{x_i}(ill)(1-p_i(ill)\,)^{(1-x_\mathrm{i})}}{\prod_{i=1}^{n} p_i^{x_i}(ill)(1-p_i(ill))^{(1-x_\mathrm{i})} + \prod_{i=1}^{n} p_i^{x_i}(leg)(1-p_i(leg))^{(1-x_\mathrm{i})}} > .5(14)$$

BF is trained and then used to classify the network feature vector $\mathrm{X_k}$ of the k^{th} time period ΔT as legitimate or illegitimate. By combining ES and BF, we can accurately detect the attack in real time with low False positive rate and high Detection Rate (see Section VI).

V.DNS MITIGATION (DM) SCHEME

When BF indicates that $\mathrm{X_k}$ is illegitimate, the mitigation action is performed to protect blockchain's nodes (i.e., target). For this aim, new OF actions are installed, using the API of SDN controller, into the OF based switch under attack; these rules have high priority to monitor speed of the suspicious flows. DM aims to mitigate illegitimate flows effectively. To this aim, each flow entry specifies *meter* with different *Meter_id* in order to monitor speed of the illegitimate flows; if the flows rates surpass the *band* (rate limiter), then DM drops suspected packets.

VI.EVALUATION

This section presents the evaluation of BrainChain. First, experiment environment is introduced. Then, performance of BF in terms of Detection Rate and False positive rate is evaluated through ROC curves.

A. Experimental Environment

In In order to evaluate the performance of BrainChain, we built a realistic experiment.

We implemented our DDoS detection module as application on top of the SDN controller. Using sFlow protocol, this module first, collects networks traffic statistics then, detects automatically illegitimate traffic. To emulate real network environment, we use Mininet [24], a popular SDN emulation tool. Mininet uses the Linux containers and virtual OF switches (e.g., OpenVswitch [25]) to allow a realistic test environment of hosts and OF switches.

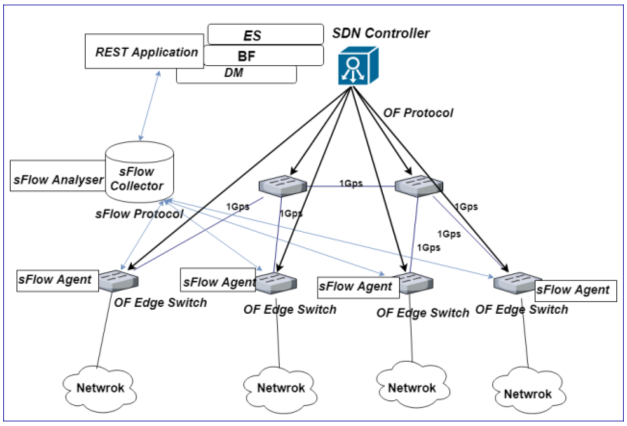

Fig.2. Experimental Environment

Mininet is installed on a VirtualBox [26] VM; VM is connected to the Internet through Network Address Translation (NAT) (for software installation and updates), and a host-only adapter is configured on VM to enable it to communicate with the host-system. Additionally, Secure Shell (SSH) is used to allow access to VM for running different software at the same time. In our testing environment, the network monitor (i.e., sFlow-RT) and SDN controller (i.e., Floodlight [27]) are installed on the host-system. We run our experiments on a PC with CPU Intel Core i7-8750H-2.2 GHz and 16GB RAM.

Fig. 2 shows the 5 components of the testbed: (a) SDN controller (i.e., Floodlight; (b) sFlow network monitor (i.e., sFlow-RT): it performs monitoring of 7500 port's switches in data center networks; (c) REST application: it executes the feature extraction scheme (ES) and BF; (d) 6 OF switches where the links's bandwidth is 1 Gbps. Each OF network is composed of multiple hosts; ones are used to launch DDoS attack and others are used as legitimate hosts executing blockchain application; (e) multiple hosts are used as Open Resolvers to send amplified flows (i.e., DNS responses). Attack rate varies between 100 and 500 Mbps; the objective is to test scalability of the proposed scheme. Table 1 shows the parameter values for our experimentation.

TABLE 1: Parameter values

	Average traffic rate (Mbps)	Sampling rate	Attack rate(pkts/s)
Exp.1	100	$\frac{1}{64}$	$200-500$
Exp.2	500	$\frac{1}{256}$	$1000-2500$

BF is a supervised machine learning algorithm that needs to be trained. To this aim, we use Scapy's Python library [28] to forge DNS requests and dnsd package in NodeJs [29] to create DNS test server. DNS server is used to send amplified DNS responses to the blockchain target nodes (i.e., victim nodes).

Fig. 3 shows that, when the control is disabled, the attack traffic is over 2000 packets per second (i.e., illegitimate DNS). However, when BrainChain is enabled, the attacks traffic is stopped when BF classifies the k^{th} vector $\mathrm{X_k}$ as illegitimate and instructs SDN controller to mitigate the traffic attack.

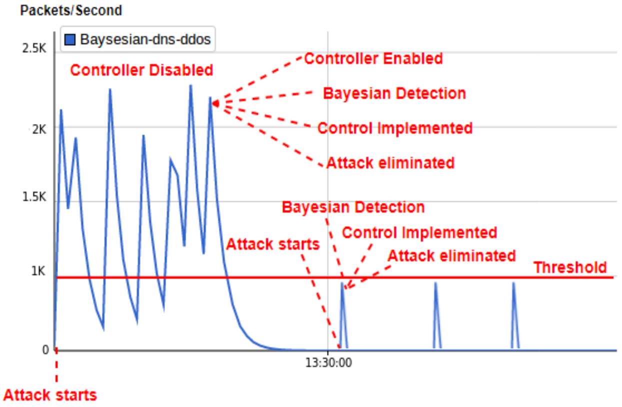

Fig.3. Blockchain network traffic before and after BrainChain

Fig. 4 shows that time taken by the BrainChain pipeline (i.e., detection/mitigation) is less than 13 seconds; this enables the network to recover quickly.

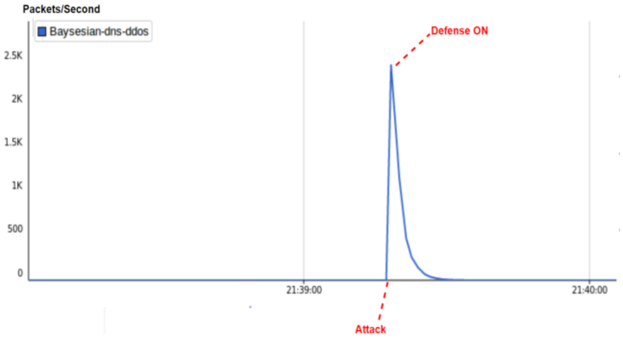

Fig.4. attack mitigation

B. Performance Evaluaion

The performance of BF is evaluated using ROC curve. In our simulation, we conducted two experiments (i.e., 100Mbps and 500Mbps) and we compared BF in terms of accuracy and false positive Rates (FPR) with our previous work [9] and one of the most prominent related schemes [15].

The ROC curve shows the TPR (True Positive Rate) called sensitivity or Detection Rate (DR) according to the FPR called anti-specificity (1-specificity). To measure BF's performance, we define DR and FPR as follows:

$$DR = \frac{TP}{TP + FN}, FPR = \frac{FP}{TN + FP}$$

where, TP (True Positives) represents illegitimate flows (i.e., illegitimate DNS requests) that are correctly classified as illegitimate ones, while FN (False Negatives) represents illegitimate flows (i.e., illegitimate DNS requests) that are classified as legitimate ones. Therefore, DR represents attacks detection rate, FP (False Positives) represents legitimate flows (i.e., legitimate DNS requests) that are identified as illegitimate ones, while TN (True Negatives) represents legitimate flows (i.e., legitimate DNS requests) that are correctly classified as legitimate ones. Fig.5 shows that BrainChain achieves around 100% of detection rate for 100 Mbps case while it has just 23% of false positive ratio, while ChainSecure [9] and [15] achieve the same rates (i.e., around 100% detection rate) with respectively 31% and 40% of false positive ratio. Fig.6 shows that BrainChain achieves around 100% detection rate for 500 Mbps case while it has just 21% of false positive ratio, while ChainSecure [9] and [15] achieve around 100% detection rate with respectively 30% and 34% of false positive ratio.

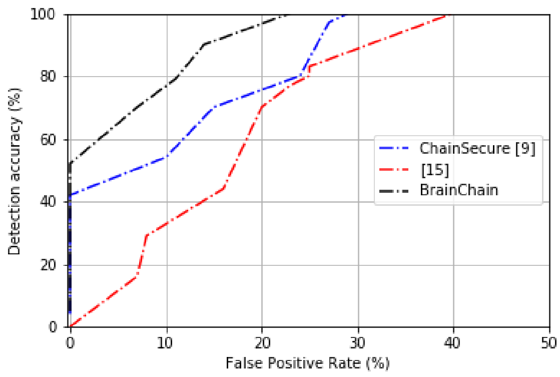

Fig.5. ROC curves for the 100 Mbps case

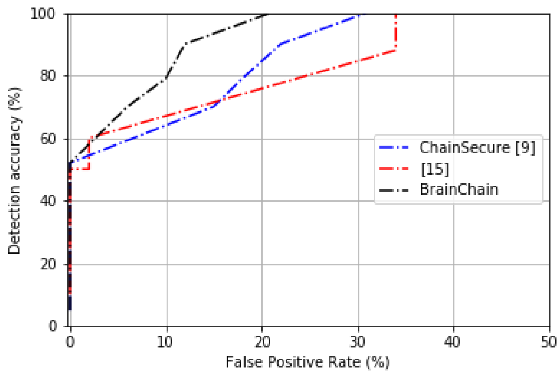

Fig.6. ROC curves for the 500 Mbps case

VII.CONCLUSION

This paper proposed, BrainChain, a scalable and efficient scheme to protect permissioned blockchain from the largest ever DDoS attack (i.e., DNS amplification attack) in the context of SDN. Because of limited number of the peers (i.e., blockchains nodes) on this blockchain's type, specific peers (i.e., blockchain nodes) can be targeted by these attacks, which may halt blockchain operations. To this aim, first, we described FS, a novel flow statistics collection scheme to gather the features of flows in an efficient way. Then, we designed a real-time detection scheme (i.e., ES and BF). Finally, DM is elaborated to mitigate illegitimate flows (i.e., legitimate DNS requests). For future work, we intend to extend our scheme to multi-domain setup by using an efficient and easy to deploy inter-domain DDoS collaboration scheme.

REFERENCES

[1] S. Sharwood. GitHub Wobbles Under DDOS Attack. Accessed: Mai. 1, 2019. [Online]. Available: https://www.theregister.co.uk/2015/08/26/github_wobbles_under_ddos_attack.

[2] B. Schneier. Lessons From the Dyn DDoS Attack. Accessed: Mai. 1, 2019. [Online]. Available: https://www.schneier.com/blog/archives/2016/11/lessons_from_th_5.htm.

[3] S. Scott-Hayward, S. Natarajan, and S. Sezer, “A survey of security in software defined networks,” IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 623–654, 2015.

[4] K. Kalkan, L. Altay, G. Gür and F. Alagöz, "JESS: Joint Entropy-Based DDoS Defense Scheme in SDN," in *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2358-2372, Oct. 2018.

[5] Z. A. El Houda, A. Hafid, and L. Khoukhi, “Co-iot - a collaborative DDoS mitigation scheme in IoT environment based on blockchain using SDN.” Accepted for publication in 2019 IEEE Global Communications Conference (GLOBECOM).

[6] Z. A. El Houda, A. Hafid and L. Khoukhi, "Cochain-SC: An Intra- and Inter-Domain DDoS mitigation scheme based on blockchain using SDN and smart contract," in *IEEE Access*. doi: 10.1109/ACCESS.2019.2930715.

[7] Yan, Q., Yu, F.R., Gong, Q., and Li, J.,‘Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: a survey, some research issues, and challenges’, IEEE Commun. Surv. Tutor., 18, pp. 602–622, 2016.

[8] ‘sFlow‘,. [Online] Available: https://www.ietf.org/rfc/rfc3176.txt

[9] Z. A. El Houda, L. Khoukhi and A. Hafid, "ChainSecure - A Scalable and Proactive Solution for Protecting Blockchain Applications Using SDN," 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 2018, pp. 1-6.

[10] Mathis Steichen, Stefan Hommes and Radu State,” ChainGuard - A Firewall for Blockchain Applications using SDN with OpenFlow”, IPTComm, 2017.

[11] Rodrigo Braga, Edjard Mota, Alexandre Passito, Lightweight DDoS flooding attack detection using NOX/OpenFlow, in: LCN ‘10 Proceedings of the 2010 IEEE 35th Conference on Local, Computer, 2010, pp. 408–415.

[12] Mehdi S A, Khalid J, Khayam S A. Revisiting traffic anomaly detection using software defined networking. In Recent Advances in Intrusion Detection, Springer Berlin Heidelberg, 2011: 161-180.

[13] Yu M, Jose L, Miao R. Software Defined Traffic Measurement with OpenSketch. In NSDI. 2013, 13: 29-42.

[14] Wang R, Jia Z, Ju L. An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking. IEEE Trustcom/BigDataSE/ISPA. 2015:310-317.

[15] Giotis K, Argyropoulos C, Androulidakis G, et al. Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. In Computer Networks, 2014, 62: 122-136.

[16] S. Lim, J. Ha, H. Kim, Y. Kim, and S. A. Yang, “A SDN-oriented DDoS blocking scheme for botnet-based attacks,” in Proc. 6th Int. Conf. Ubiquitous Future Netw. (ICUFN), Jul. 2014, pp. 63–68.

[17] Zaalouk, A., R. Khondoker, R. Marx and K, Bayarou. “OrchSec: An Orchestrator-Based Architecture For Enhancing Network-Security Using Network Monitoring And SDN Control Functions”. Network Operations and Management Symposium, May 5-9, 2014.

[18] “REST API”, . [Online] Available:” http://www.sflow-rt.com/reference.php”.

[19] “sFlow-RT”, http://www.sflow-rt.com”.

[20] H.J. Landau,J.E. Mazo,S. Shamai,” Shannon theory: perspective, trends, and applications special issue dedicated to aaron d. wyner”, IEEE Transactions on Information Theory,2002.

[21] Corentin Briat, "Convergence and Equivalence Results for the Jensen's Inequality—Application to Time-Delay and Sampled-Data Systems",IEEE Transactions on Automatic Control,2011.

[22] C. P. Robert. Le choix Baysien. Principes et pratiques. Springer, 2006.

[23] Peter D. Hoff. A First Course in Bayesian Statistical Methods. Springer, 2009.

[24] Mininet. [Online] Available: http://mininet.org.

[25] OpenVswicth. Available: https://www.openvswitch.org/.

[26] "Virtual Box", [Online]. Available: https://www.virtualbox.org/.

[27] Floodlight. [Online] Available:http://www.projectfloodlight.org/ .

[28] Scapy,”http://www.secdev.org/projects/scapy “.

[29] NodeJs,” “Nodejs”, “https://nodejs.org/en//”.