National College of Ireland

Implementation of Blockchain Technology in IoT environment to mitigate DDoS Attacks

MSc Research Project
Masters in Cybersecurity

Rishabh Vyas
Student ID: 21108714

School of Computing
National College of Ireland

Supervisor: Ross Spelman

National College of Ireland

National College of Ireland

Project Submission Sheet – 2021/2022

Student Name:	Rishabh Vyas		
Student ID:	21108714		
Programme:	MSc. In CyberSecurity	**Year:**	2021-2022
Module:	MSc. Research Project/Internship (MSCCYB1)		
Supervisor:	Ross Spelman		
Submission Due Date:	01/02/2023		
Project Title:	Implementation of Blockchain technology in IoT environment to mitigate DDoS Attacks		
Word Count:	12250	**Page Count:**	29

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:

Date: 29/02/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Implementation of Blockchain Technology in IoT environment to mitigate DDoS Attacks

Rishabh Vyas
21108714

Abstract

There are a number of applications in the field of the Internet of Things (IoT) such as smart homes, smart agriculture, pharmaceutical companies, etc. The security of IoT devices is at risk as a result of distributed denial of service (DDoS) attacks. Security issues associated with IoT devices can be exploited by cybercriminals to operate botnets in the event of a DDoS attack. The limited memory and processor capabilities of IoT devices result in resource constraints. As a result of the limited amount of memory and the computational complexity of today's IoT devices, many cyberattacks could be launched against such devices. The use of blockchain technology may provide a solution to some of the security challenges faced by IoT. In order to mitigate DDoS attacks on IoT devices, blockchain-based solutions should therefore be used to monitor and mitigate these attacks. The prevalence of DDoS attacks has been rising in recent years on IoT devices. To perform a DDoS attack, rogue devices or exploited devices are introduced to the system and are used in conjunction with other exploited devices to create unending congestion throughout a computer network. This project aims to integrate IoT devices with blockchain technology to tackle the DDoS security challenges associated with using IoT within the context of this research. Furthermore, we will determine the cost of deployment and the flexibility for such devices to be used in small and large operations.

1 Introduction

As a result of the phenomenal growth in the adoption of IoT environments, blockchain technology has gained a variety of applications that facilitate the delivery of security. The use of blockchain technology for delivering security in IoT environments has risen significantly in recent years. It is becoming increasingly common for platforms like IoT devices to be targeted for DDoS attacks due to the sheer prevalence of these devices. As a result, both service providers and network operators are at risk of substantial losses due to these attacks. An extensive evaluation of the number of digital devices connected to the Internet of Things (IoT) in the world suggests that the number is likely to grow from 9 billion devices in 2019 to nearly 30 billion by 2030, according to a report by (Vailshery, 2022). IoT devices are inherently vulnerable, and therefore a variety of these vulnerable devices might be exploited by a cybercriminal to launch a large-scale botnet attack. Software-Defined Networking (SDN) is a unique paradigm to simplify network administration and enable cutting-edge techniques to dynamically build and manage networks with a continuous focus on simplicity and functionality. As the complexity and scale of today's networks continues to increase, Software-Defined Networking (SDN) is becoming increasingly popular. By segmenting the network into two planes - data planes and control planes - SDN gives the network additional authority over the data plane in addition to introducing new methods for dealing with various types of DDoS

attacks. The current DDoS mitigation solutions do have several problems, including their high cost, their lack of versatility, and the fact that they require a lot of effort and are cumbersome to implement; most importantly, they are centralized. As a consequence of this centralized approach, several disadvantages can occur, such as failure points in the network, data availability and dependability, which are vulnerable to DDoS attacks and thus require extra caution. As a result of the introduction of new technologies, including software-defined networking (SDN) and blockchain, DDoS attack collaboration is becoming more affordable, efficient, and adaptable (El Houda, Hafid, & Khoukhi, 2019). As part of this research, decentralized blockchain technology will be integrated into an IoT environment using open-source software, Ethereum. A public test network, Görli (Etherscan, 2022), will be used, which is the first cross-client testnet that implements proof-of-authority.A major objective of this project is to propose a means of sharing DDoS information that is decentralized, secure, and can be applied across many SDN-based domains that can leverage blockchain technology and smart contracts to reduce DDoS attacks based on a decentralized and robust approach. Furthermore, a testing phase is intended to assess the feasibility of deploying the final system at a reasonable cost.

The remainder of the research paper can be summarized as follows: Using examples laid out in **Section 2**, this paper is aims to lay out and justify why implementing a mitigation infrastructure of this type is vital. **Section 3** of the paper contains background research on several similar topics, which is followed by a critique of these papers, highlighting their shortcomings or flaws. A DDoS mitigation solution is presented in **Section 4** that combines the methodology of intra-domain DDoS mitigation with the method of inter-domain DDoS mitigation in order to mitigate DDoS attacks. An integrated system of blockchain technology using smart contact that can be used across a number of SDN-based domains will be created by integrating the technologies and software presented in **Section 5** of the paper in order to mitigate DDoS attacks that are occurring across SDN domains on the internet. A significant amount of illegal traffic generated by compromised IoT botnets will be assessed in **Section 6** of the study in order to determine whether the deployed system can effectively handle the impact. In order to assess the system, the system will be subjected to a series of tests, using a protocol called sFlow (inmon: sFlow-RT, 2022), which will monitor the flow of packets through the system. It will enable us to compare the amount of time it takes the system to mitigate a DDoS attack as a result of this. A cost-benefit analysis for evaluating the deployment of the system is performed by aggregating the amount of gas Ethereum consumes as well as estimating its value in euro terms. **Section 7** of this paper concludes the research that has been conducted in this paper.

2 Related Work

Presented in this section are the existing DDoS mitigation techniques that have been published in the field of cyber security that can help to mitigate DDoS attacks.

It is a key aspect of this study to develop a decentralized, secure, and cost-effective means for integrating devices in an IoT environment using blockchain technology in a way that keeps the devices in the environment connected and decentralized. With the help of open-source blockchain software, Ethereum, and its test network Görli, which is the first proof-of-authority cross-client testnet for blockchain technology, this research aims at analysing the incorporation of the decentralised blockchain technology into IoT infrastructure using an open-source architecture.

In the literature, there have been several approaches to mitigating DDoS attacks. However, many of the approaches depend on the wider/large-scale deployment of a mitigation strategy, so the majority of the approaches might be unable to address this issue owing to the difficulty associated with their implementation. Additionally, collaborative DDoS mitigation solutions are notoriously costly, rigid, and difficult to set up, as well as being centralized throughout the entire process. The centralised solution is inefficient, as it only has one point of failure and has issues with data consistency and availability, which makes it difficult for autonomous systems to exchange information and make effective decisions regarding the mitigation of DDoS attacks when using it. Using some of the new technologies available today, like software-defined networking, blockchain technology, and smart contracts, we are able to reduce the occurrence of DDoS attacks among multiple autonomous systems in a more efficient, effective, and adaptable manner.

2.1 DDoS Attack Mitigation Techniques

The blockchain-based Co-IoT approach (El Houda, Hafid, & Khoukhi, 2019) has been shown to be effective not only in terms of reducing current attack pathways, but also in terms of reducing the attack source itself. The Co-IoT architecture is built upon an open blockchain platform by leveraging the Ethereum's official test network Ropsten, which is the official test network of Ethereum. Throughout the project, blockchain technology has been utilized in order to build decentralized, efficient, and robust mechanisms for tackling DDoS attacks that are decentralized, efficient, and robust. Several SDN domains were vulnerable to DDoS attacks in the past, which is why a low-cost, safe, customizable, and decentralized mechanism for DDoS cooperation was developed using Ethereum smart contracts to mitigate the attacks. This paper intends to build a research and system that is very similar to the project created and the research system done by (El Houda, Hafid, & Khoukhi, 2019). There has been a decline in the open public network Ropsten, which was used by the author in this paper. It has since been deprecated and it is unlikely to be maintained well past 2022. This study applies the findings of the Co-IoT study to the Görli testnet of Ethereum and compares the results of the tests with those of the Ethereum main net merger. Since the Görli testnet is considered to be the fastest among all Ethereum testnets, the study considers that it will continue to receive long-term support after the Ethereum main net merger.

DDoS attacks are aimed at overwhelming the computers of a target with illegal traffic, such as a large amount of network bandwidth and CPU usage in the target system. When launching

a DDoS attack, a large number of compromised computers, usually computers or smart devices, are used by hackers to conduct the assault. Botnets which have been compromised send large volumes of illegitimate traffic to a target network service in order to deny legitimate users access to it. A DDoS attack may overwhelm HTTPS and SSL servers, showing that encryption alone does not offer a viable solution for mitigating the effects of DDoS attacks. There are several security flaws in the IoT, which can lead to DDoS attackers being able to execute their attacks. It is common for IoT devices to be insecure, have unsecured networks, lack encryption, authentication and authorization problems due to insecure software, unsecured networks, and no encryption available. NFV and SDN are technologies that allow automatic traffic analysis to be used in order to detect a DDoS attack, which can reduce the threat of a DDoS attack (Hanan Mustapha, 2018).

A new AntibIoTic scheme for the protection of IoT devices from DDoS attacks is proposed by the authors in (De Donno, Dragoni, Giaretta, & Mazzara, 2018). Due to the fact that some IoT devices are susceptible to being patched at first, their centralized approach does have some limitations in relation to the computational requirements and the complexity of the process.

Using the IoT environment as the context for a study of the mitigation of DDoS attacks, (Zawar, Ullah, Li, Levula, & Khurshid, 2022), created a survey on DDoS mitigation in their study in 2022. To begin with, the author discusses the impact that DDoS cyberattacks have on IoT networks and the services that are associated with them. The use of blockchain in the Internet of Things is then discussed. This is followed by a discussion of the potential for its implementation to mitigate DDoS attacks, along with its limitations in its implementation. A final consideration was made about several Blockchain-based solutions which can be used for mitigating DDoS attacks in the context of IoT environments. It is then that (Zawar, Ullah, Li, Levula, & Khurshid, 2022) summarizes the four main types of Blockchain-based solutions: Those that are enabled via Distributed Architecture, those that seek access control, those that utilize traffic control, and those that are based on the Ethereum Platform. It is true that the author proposes a variety of Ethereum-based solutions in the paper, however, the author fails to incorporate Software Defined Networking to be a controller for the Internet of Things devices in many of the pre-existing surveys in the paper.

2.2 Inter Domain DDoS Mitigation Schemes

By modifying the Border Gateway Protocol (BGP) for the purpose of allowing incidents to be reposted in the BGP signal in order to mitigate DDoS attacks, the authors (Giotis, Apostolaki, & Maglaris, 2016) attempted to modify BGP for the purpose of preventing DDoS attacks from being mitigated. However, the process of changing the BGP protocol proved to be exceptionally difficult due to its complexity. It was also reported that the SND-based domains that were used also experienced a significant amount of event latency because they do not update in real-time like many others. As a consequence of this approach, there is a possibility of fake incidents being reported from illegal domains. This is because there is no mechanism to validate the legitimacy of the incident.

(Steinberger, Kuhnert, Sperotto, Baier, & Pras, 2016) has proposed a method called DDoS Open Threat Signalling (DOTS) that allows DDoS attacks to be advertised in a more effective way. One of the components of the DOTS protocol is the DOTS client, and another is the DOTS controller. The DOTS controller is responsible for the supervision and transmission of inter-domain interactions and intrusions that occur between domains across the DOTS network.

It then contacts the DOTS client to seek a mitigation service as soon as an attack is discovered. The use of a flow-based event sharing protocol facilitates the deployment and collaboration across domains by enabling the sharing of events among participants. There is a high probability that the global deployment is not taken into consideration due to the complexity of implementing it and thus may result in ineffectiveness. Furthermore, collaborative processes are susceptible to being compromised in a variety of ways. As much as it is possible to resolve this challenge through the use of a secure public key infrastructure (PKI), it is also expensive to maintain and implement one of these systems.

The author (Stankovic, 2014) emphasizes a number of challenges that arise when implementing IoT-based systems, emphasizing the need for further research into a range of issues related to IoT-based systems. These include scaling, architecture, dependencies, knowledge creation, big data, robustness, openness, security, privacy, and human-in-the-loop. As IoT networks grow in scale, it becomes more difficult to ensure that the two main security concerns of IoT are respected: trust and control. Despite the benefits PKI solutions offer for large-scale systems, it may not be possible to implement key management in an IoT environment due to resource constraints.

(Rashidi, Fung, & Bertino, 2017) proposal, CoFence, was conceived as a means of securing Network Function Virtualization-enabled domains in a distributed network by using an NFV platform to facilitate collaboration among these domains. In order to protect an NFV-based domain from traffic attacks, traffic attack packets are redirected to another NFV-based domain. CoFence presents a number of privacy concerns because you need to redirect traffic to another NFV-based domain in order to accomplish this. Additionally, the redirection process results in a delay in the submission of incident reports.

An Ethereum smart contract for reporting malicious IP addresses should be designed using reputation scores, according to (Spathoulas, et al., 2018) research paper. Nevertheless, the collaboration process is at risk of being compromised.

(Simpson, et al., 2018) scheme remains challenging because of its deployment complexity and overhead.

2.3 Intra DDoS Mitigation Schemes

In his presentation, (Rodrigues, et al., 2017) presented how blockchain technology and smart contracts can be used to develop a decentralized method of sharing banned IP addresses. A central organization, however, is required to provide certificates of IP address ownership as part of this system in order for it to work.

(Alharbi, Aljuhani, Liu, & Hu, 2017) proposed an alternative DDoS detection strategy aimed at detecting DDoS attacks with the intention of preventing them, though it was mostly intended for detection and there was no recommendation for preventing them.

According to an article published by (Zhang & Green, 2015), a method for preventing DDoS attacks has been proposed by the authors. There is no doubt that its system is very similar to that of a firewall, as it differentiates between suspicious requests and authentic ones and deals with each one accordingly. Although the technique provides useful differentiations, it is

not suitable for large-scale systems due to its high computational demands, and as a result, it is less effective for such systems due to its less efficient processing.

(Ferguson & Senie, 2013) proposes a filtering method in the BCP 38 standard, which requires ISPs to adhere to the following:

a) Ascertain that the IP addresses that are used in packets coming from its network are valid; and
b) If a packet contains forged source addresses, it should be further filtered in order to be sure that it is not in its range of legitimate addresses.

A solution such as this has proved to be effective against IP spoofing attacks in the past. Despite this, it does absolutely nothing to prevent attacks originating from a real IP address that is based on the Internet.

A software-defined traffic measurement scheme known as OpenSketch was proposed by (Yu, Jose, & Miao, 2013). Through a powerful pipeline that consists of three stages (hashing, filtering, and counting), OpenSketch provides an efficient method to collect measurement data from multiple sources. There is, however, a risk of overloading the control plane in OpenSketch, as all the counters are sent to it for analysis.

The OF protocol was suggested by (Mehdi, Khalid, & Khayam, 2011) as a means of detecting anomalies in the context of SDN. However, the design of the scheme was based on a small-scale setup (i.e., a home environment); in larger environments, high-rate data traffic to the SDN controller may have consequences for the control plane, such as overflow.

The author of (Lim, Ha, Kim, Kim, & Yang, 2014) in their paper proposed a DDoS mitigation scheme for botnet-based attacks that runs on a controller using SDN technology to mitigate their impact. In order for the victim to be protected, it is necessary to establish a large number of communications between the control plane and the data plane. Aside from making the SDN controller vulnerable to DDoS attacks, this scheme proposed by (Lim, Ha, Kim, Kim, & Yang, 2014) also required the SDN controller to have a high level of latency when cooperating with the SDN controller, which makes the controller susceptible to DDoS attacks.

The entropy-based scheme (Wang, Jia, & Ju, 2015) proposed is a method for detecting OF switches, however, it does not find the victim, nor the illegitimate hosts possible for blocking them in the long run.

The protocol combination of the sFlow and OF method is presented in (Maglaris, 2014), a paper that shows how to detect DDoS attacks by reducing the overheads of communication between the data plane and the control plane by combining sFlow with OF. The scheme that was proposed by (Maglaris, 2014) works well, but there is a high risk of false positives associated with it.

In order to address the inadequacies of existing schemes, this study proposes a system for enabling several SDN-based domains to collaborate safely and efficiently in a decentralized manner, using blockchain technology, to solve the inadequacies of existing schemes through an inter-domain mitigation system. By using Ethereum's blockchain technology, the intricacies of developing and updating new protocols can be avoided, and as a result, a central authority will not be required and the collaboration will be enforced with authorization for each party participating.

3 Research Methodology

In this section, a brief summary of the Ethereum platform is presented, as well as an outline of the methodological approach that will be adopted during the research as well.

There are a number of digital money platforms, online payment platforms, and other applications that are powered by the Ethereum open-source platform. In addition to its open-source nature, Ethereum is also known for its level of transparency. It is possible to fork and reuse the features provided by other developers. There has been a significant increase in the use of decentralized applications in recent years, which have transformed business models and disrupted industrial sectors as a result. Dapps, on the other hand, use a decentralized network to run their backend code (smart contracts), in contrast to servers that are centralized. As the name suggests, Ethereum is a distributed ledger platform that stores data on the blockchain, while smart contracts handle the program logic. Like rules on a blockchain, smart contracts follow their rules exactly as they are written and are observable by anyone. Code can now be used to mediate agreements and transactions. On the Ethereum network, Dapps cannot be changed once they have been deployed. Dapps are controlled by logic written into their contracts, not by individuals or companies (Welcome to Ethereum, 2022).

The system created as part of this project is going to take advantage of multiple SDN domains to collaborate with one another and mitigate DDoS attacks on them using the inter- and intra-domain DDoS mitigation scheme. We will also be utilising the use of smart contract built on top of the blockchain platform which when integrated with the SDN controllers share the attack information as well as the IP address of the attacker with each block mined to help with the mitigation procedure. The SDN-based domains within this system can share attack information decentralized in a manner based on blockchain technology and smart contracts in order to better protect themselves against cyber-attacks. As a first step, the creator of the smart contract must create and deploy a collaboration contract as part of the production process. Following the creation of the repository, collaborators and participants who have been approved will be added. In order to detect and mitigate DDoS attacks against a victim, it is imperative to remember that the domain under attack is utilizing a mitigation scheme in order to detect and mitigate DDoS attacks when attackers are controlling many compromised IoT devices and creating DDoS attacks through multiple SDN-based domains. Using Ethereum's smart contract system, it is also possible to store the IP address of the attacker who used this suspicious IP address to attempt to attack the system. Then, once the block has been mined, each participant or collaborator in the collaboration scheme for preventing DDoS attacks becomes able to access malicious IP addresses that should be blocked as a result of the block being mined. In order to transmit the resulting report on attack information from each domain running an Ethereum client, a platform that implements the Ethereum protocol is used, such as GoEthereum (Go Ethereum, 2022), which is a platform that supports the implementation of the Ethereum protocol.

The creation of smart contracts is integral to the creation of the system because the contract owner (CO), who intends to work with multiple autonomous systems, makes use of the Ethereum platform to create smart contracts which will enable them to collaborate with one another. In order to ensure pseudonymity while maintaining transparency on the Ethereum platform, the CO initially creates a pair of keys, which include a private key, and an external

account owned by the CO. The address of the external account owned by the CO is the hash value of the public key; the addresses of these two keys are the addresses of the public keys. Using Ethereum's wallets to generate keys is an easy way to establish the key generation process. The key(s) created in the process will then be able to be used in the creation of smart contracts as well as in the execution of embedded functions within them. Using smart contracts that have been created by the CO, the collaborators will be able to be incorporated into the system with the help of smart contracts. Among the information that will be included will be the addresses of collaborators. Moreover, the CO has access to a number of tools related to the management of collaborators in the context of a project, including the ability to add and remove them transparently. Moreover, these collaborations allow the CO to report malicious IP addresses as well as the option of removing those addresses from their database in the event that they have been reported.

A system that incorporates autonomous systems will be developed for the purpose of classifying them into three different categories, which are the "source domain", the "intermediate domain", and the "destination domain". As a result of attacking a network within the domain of the source network, an attacker may initiate DDoS attacks against it. Illegitimate DDoS traffic is routed through intermediate network domains. Domains that host victims are considered destination networks, as they are the places where victims are located. Upon identifying the DDoS attempt, the SDN controller for the victim's domain immediately ceases the attack and terminates it as soon as possible. In the event that the protocol is approved, the SDN controller for the domain that corresponds to the malicious IP address will send a transaction to Ethereum's smart contract informing the network of the IP address known to be malicious. Once a transaction has been confirmed on the blockchain, smart contracts automatically emit an event upon the confirmation of that transaction. It is anticipated that either the SDN controllers of the source and intermediate networks will receive this event, as well as the authorized collaborators under the collaboration agreement. The illegitimate traffic is eventually intercepted near the point of origin by authorized collaborators.

The goal of this study is to develop a system that is easy to deploy, secure, efficient, and affordable to mitigate the effects of DDoS attacks on a blockchain-based system utilizing smart contracts to mitigate DDoS attacks. As part of the test, Ganache (Ganache, 2022), a popular private test simulator, will be used, along with the official Ethereum test server Görli. In order to deploy an autonomous system, one must first create a smart contract, and then a collaboration contract must be drawn in order to have the autonomous system run itself automatically without human intervention from any of the users. The smart contracts that will be created using the truffle framework (Truffle Suite: Sweet Tools for Smart Contracts, 2022), using the Solidity (Solidity, 2022) programming language as the coding language, will then be compiled into the Ethereum Virtual Machine (EVM) [Figure 1. (Ethereum Virtual Machine, 2022)] which will then be used for the execution of the smart contracts. In order to generate EVM byte code for the collaboration contract, the EVM will be compiled together with its 'Application Binary Interface' (ABI) in order to generate EVM byte code. As a next step, it is necessary to deploy the collaboration contract on the blockchain in order to complete the process. When smart contracts have been deployed, they can be invoked once a set of addresses and ABI definitions have been defined. Should the need arise, it is also possible to delete the contract if it is necessary to do so. Ganache, which is a private blockchain simulator, will be used to test

Ethereum's smart contracts during the initial testing process. After that, the smart contract will be deployed to Ethereum's official test network Görli where it will be tested for functionality. After this evaluation has been completed, the results will be compared with those produced by (El Houda, Hafid, & Khoukhi, 2019) and will be evaluated accordingly.

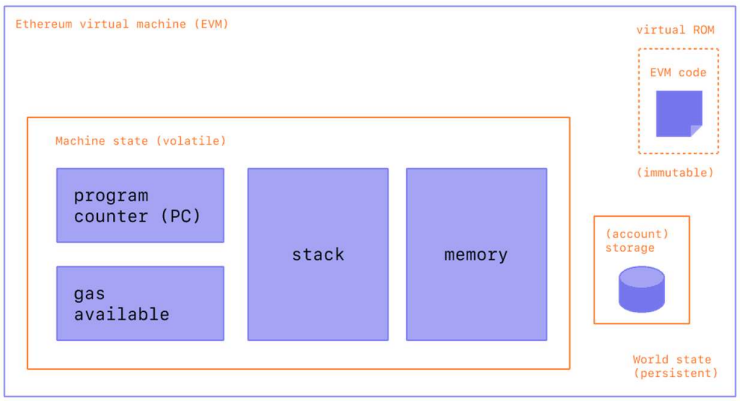

Figure 1 Ethereum Virtual Machine.

The functions of the Collaboration contract that will be created with the aim of establishing cooperation will be as follows:

- One of the primary functions that the CO is tasked with executing is one which involves adding collaborators to the process, which is exclusively used by the CO. As a prerequisite to this function, both the collaborator's information and the external account which was previously generated are required. Furthermore, it also enables the addition of timestamps when adding a collaborator to a contract, so that these timestamps can be clearly identified. Adding the collaborator to the process does not begin until the credentials of the owner are validated, at which point the process of adding the collaborator to the process can begin.
- Either the CO or the collaborator that has been authenticated will be able to access the function to add a record to the database of the reported malicious IP addresses, which will allow them to monitor any new IP addresses that are reported.
- The other feature that can only be accessed by the creator of the collaboration contract is that of deleting a collaborator, which will also require the information of both the collaborator and the external account that was previously created for the collaborator to be provided.
- In addition to being able to set the status of a collaboration contract (active/inactive) when creating it, the creator of the collaboration contract also gains the ability to change its status.
- The CO or the authenticated collaborator can also access the function to remove the potentially malicious IP addresses from the database through the function to remove the potentially malicious IP addresses from the database (El Houda, Hafid, & Khoukhi, 2019).

It is important to note that Ethereum has two types of accounts: (1) Externally Owned Accounts (EOAs): this type has public and private keys; it can be used for sending transactions

between ETHs or to smart contracts; and (2) Contract Accounts: these accounts run code and do not have public/private keys. An organization deploys a smart contract, called the Collaboration contract, in order to implement the collaboration process. As a first step, we will describe the initialization of a Collaboration Contract in the following points. Following that, we will discuss the functions of the Collaboration Contract in more detail. Initiation of the Collaboration Contract: This process is responsible for defining the state variables in the collaboration contract.

An active or inactive Collaboration Contract indicates the status of the contract. In order for the contract to be activated or deactivated, it is the responsibility of the contract owner. An indication of who is the owner of the contract. The NumberOfCollaborators property defines how many collaborators will work together. NumberOfRecords: It defines how many records (i.e., IP addresses) there are in the database. In this case, the address of a collaborator is stored in the CollaboratorsAdr variable. There is a purpose for this array, which is to reduce the cost involved in finding and removing a specific collaborator from the collaborators mapping, by making use of the following format. A RecordsAddress is used to store the records of suspicious IP addresses. RecordsAdr aims to reduce the amount of time and energy that it takes to find and remove a specific record from a records mapping by utilizing the records address. The Collaborator class defines a mapping collection from the address of the collaborator to a corresponding Collaborator struct that holds the details of the collaborator. There is a collection of mappings between an IP address and its corresponding record struct within the IP_address collection. There is only one owner of a modifier type. This modifier was applied to both functions that (add or remove) collaborators from/to the smart contract and functions that either activate or deactivate the smart contract. In other words, only the owner of the contract is able to invoke the functions (adding/removing) the collaborators or (activating/deactivating) the contract. Collaborators with modifier types are the only ones that can be selected. Basically, the caller's address is the input and the caller's authorization is checked to see if he is eligible to execute the function for which the modifier is applied on behalf of the caller. As a result of applying this modifier to the function that can be invoked by collaborators (owner included) of the contract to add or remove records (to or from) the smart contract (as described above), only collaborators (owner included) will be able to invoke (add or remove) the records.

4 Design Specification

The research is going to be performed on a laptop equipped with a 12-core Intel Core i7-8750H processor running at a speed of 2.2 GHz as well as 16GB of RAM.

4.1 System Architecture

It is important to note that the architecture of the intra-domain DDoS mitigation method consists of four main modules: the Intra Entropy-based scheme (I-ES) which measures the randomness of data inside the domain by using sFlow. This paper proposes an intra-Bayes-based scheme (I-BS) for identifying illegitimate flows by using entropy values and classifying them. A system called intra-domain mitigation (IDM) is used to effectively mitigate the flow of illegitimate data within a domain; as well as the blockchain layer, which is used to mitigate the flow of illegitimate data between domains. In order to mitigate intra-domain DDoS attacks, there are two main phases: detecting the DDoS within the intra-domain machine learning scheme and mitigating the attack. In this module, I-ES, I-BS, and I-DM are used to detect, in real-time, illegitimate flows that are trying to flow through the system. In order to mitigate DDoS attacks from inter-domains, a blockchain module is used.

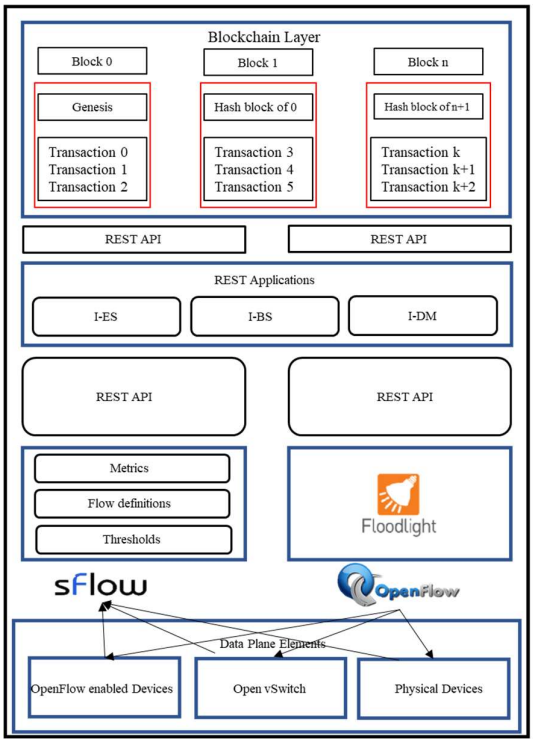

Figure 2 System Architecture.

Through the use of network traffic flow features that are unique to the victim's domain, I-ES aims to measure the randomness of data inside the victim's domain. In addition to detecting stateful network traffic features in real-time, the purpose of I-BS is to detect illegitimate flows that are taking place in the network. In the application that runs on top of the SDN controller, the entropy values are being used in order to collect traffic information in order to detect illegitimate flows automatically. In addition to using the REST API in our process of detect and mitigate illegitimate traffic, we also use the API to manage any SDN controller and block any traffic that is illegitimate. I-DM is aimed at mitigating illegitimate traffic inside an organization to the extent that it is possible. It was not designed to support QoS features, but because OF 1.3 has introduced meters into the protocol, OF is now able to support such features. There is a meter associated with each flow entry. The Meter entries are configured with different Meter_ids for monitoring the speed of the classified illegitimate flows that are detected by I-BS; if the speed of the classified illegitimate flows exceeds the band, the suspected flows are dropped by I-DM.

4.2 DDoS detection and mitigation module

Throughout this section, we will provide an overview of our information collection method, which is based on flow packet sampling using sFlow. In this section, we will describe I-ES, which measures the randomness of data inside a domain and extracts network features; in the next section, we will describe I-BS, which detects illegal flow.

4.2.1 Statistical collection for Flow Analysis

There are two methods that are commonly used to gather information during the collection process: the first is based on the OF protocol, and the second is focused on the flow monitoring process. Our approach to flow monitoring in this paper combines the use of the sFlow protocol with flow monitoring methods. It is necessary to discuss each of these methods in the following paragraphs as well as to justify the choice chosen. As seen in the earlier solutions, proposed in the paper by authors (Ferguson & Senie, 2013), (Mehdi, Khalid, & Khayam, 2011), (Yu, Jose, & Miao, 2013), (Wang, Jia, & Ju, 2015) and (Lim, Ha, Kim, Kim, & Yang, 2014) in the intra-domain section, propose a way to detect DDoS attacks in the context of SDN by collecting and sending, periodically, information regarding the flow characteristics (e.g., the number of received packets and the duration of matched flows) to the SDN controller via the OF protocol. In order to collect features, the OF protocol can be used. OF switches respond by sending the flow table content (ofp_flow_stats_reply) to the SDN controller, who responds by requesting the features (ofp_flow_stats_request). As a consequence of this method, it is possible to identify the overall traffic of flow information as it passes through the data plane. The downside of this approach is that it may overload the control plane and exhaust the bandwidth between OF controllers and OF switches, or it may also exhaust the Ternary Content Addressable Memory (TCAM) in OF switches if there is an overload. Due to these limitations, OF-based methods cannot detect high-rate DDoS attacks.

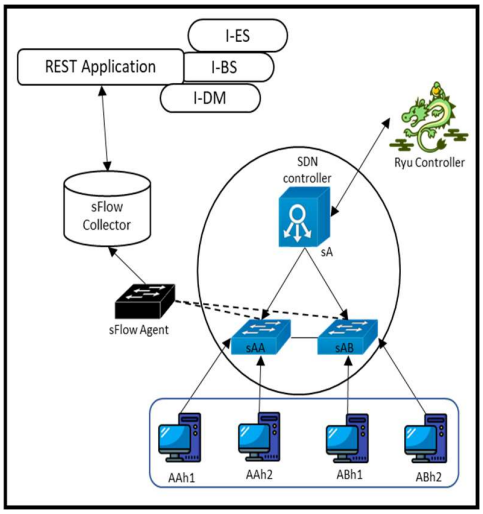

Figure 3 Experimental Environment.

Due to the limitations of the flow monitoring method described above, it was considered efficient to use the flow monitoring method based on the SFlow protocol in order to address some of these shortcomings. By doing this, we are able to achieve increased efficiency and scalability and avoid consuming bandwidth between the SDN controller and OF switches. When DDoS attacks occur at a high volume, sFlow is used for flow aggregation which is needed to manage the influx of traffic that is generated during those attacks. Every sFlow agent embedded in the data plane (data plane devices) sends out periodic packet samples to the sFlow collector that updates the counters every time a packet sample passes through the collector during the monitoring interval. Then, I-ES calculates entropy values periodically and I-BS detects illegitimate flows automatically as a result of the calculated entropy values.

4.2.2 Intra Entropy-based Scheme

The Table 1 shows the list of naming conventions that have been used to describe I-ES. Shannon's information theory (Shannon, 1951) forms the basis of I-ES. Entropy is a metric for measuring disorder or randomness in incoming data, i.e., the total quantity of data coming in over a given period of time. The I-ES application runs on top of the controller and utilizes the sFlow protocol as the method for communicating with the controller. Using traffic information, it calculates the entropy of each flow and examines the correlation between them. The number of packets sent to a victim's domain as a result of a DDoS attack, referred to as IPDST (i.e., the victim's IP address), increases significantly, resulting in a concentrated distribution of IPDST (i.e., the victim's IP address); however, if the victim's network is in its normal state, the probability distribution of IPDST becomes more dispersed. A high entropy value indicates that IPDST probability distributions are widely dispersed, whereas a low entropy value indicates that IPDST probabilities are concentrated. Consequently, we use the I-ES tool to measure the changes in information about traffic inside the victim's domain during the monitoring interval which is denoted by △T.

Table 1 Notations.

Notations	Definition
MAC_{src}	The source packet's MAC address
MAC_{dst}	The destination packet's MAC address
IP_{src}	The source IP address of a packet
IP_{dst}	The destination IP of a packet
$Port_{src}$	The source $Port$ of a packet
$Port_{dst}$	The destination $Port$ of a packet
IP_{proto}	The protocol used for transportation of a packet (TCP/UDP)
S_j	The OF switch S_j
$\triangle T$	The monitoring interval
$F_{i,j}$	Flow F_i at a local switch S_j
$p_{i,j}$	The probability distribution of overall flows at local OF switch S_j for the flow f_i
N	The total number of flows at the local OF switch S_j
R	Set of real numbers
I	Set of positive integers

It is our aim in this research to define a flow as a seven-tuple:

$$\{MAC_{src}, MAC_{dst}\ , IP_{src}, IP_{dst}\ , Port_{src}, Port_{dst}\ , Port_0\}$$

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

$$F_{i,j}(IP_{dst}, S_j) = \{< IP_{dst}\ , S_{j,t} > | S_j \in S_{i,j} \in I, T \in R\} \rightarrow \text{I}$$

Here, IP_{dst} is refered to as the Ip address of the destination which is $F_{i,t}$ with the current timestamp, and $S = \{S_{i,j} \in I\}$ which is the set of the OF switches. Here we also assume that $F_{i,j}(IP_{dst}, S_j)$ to be the count of the number of packets of the flow that is $F_{i,j}$ as the time t During the monitoring interval △T at local OF switch S_j the variation of the number of packets for the flow $F_{i,j}$ is defined as:

$$| N_{F_{i,j}}(IP_{dst_i}, S_j, t + \triangle \mathrm{T}) | = |F_{i,j}(IP_{dst}, S_j, t + \triangle \mathrm{T})| - |(F_{i,j}(IP_{dst}, S_j, t)| \rightarrow \mathrm{II}$$

Flow f_i is assumed to have a probability $p_{i,j}$ over all the flows at the local OF switch S_j such that the following equation is given:

$$p_{i,j}(IP_{dst}, S_j) \frac{N_{F_{i,j}}(IP_{dst}, S_j, t+\triangle \mathrm{T})}{\sum_{i=1}^{N} N_{F_{i,j}}} \rightarrow \mathrm{III}; where \ \sum_{i=1}^{N} p_{i,j}(IP_{dst}, S_j) = 1$$

In the following example, we will take a random variable IP_{dst} which represents the number of flows that occurred during the time interval $\triangle$T. As we can see from the following equation, we can compute the entropy of flow f_i at local OF switch S_j as follows:

$$H(IP_{dst}) = -\sum_{i=1}^{N} p_{i,j}(IP_{dst}, S_j) \log_2 p_{i,j}(IP_{dst}, S_j) \rightarrow \mathrm{IV}$$

As a measured metric, we divide the values of entropy by a metric that is independent of the number of distinct values, which is log2 N, so as to have a metric that is independent of the number of distinct values. Thus, we can define that normalized entropy values fall between [0, 1] and are thus defined as follows:

$$H(IP_{dst})` = \frac{H(IP_{dst})}{\log_2 N} \rightarrow \mathrm{V}$$

It depends on the attack (e.g., DRDoS, DDoS) under investigation as to which attribute (e.g., IP_{src}, IP_{dst}) should be utilized to aggregate flows. DDoS attacks cause significant decreases in the entropy values of flows that have the same destination IP address (i.e., the path to the attack target) due to a sudden increase in the number of flows having the same destination IP address (i.e. the attack target); whereas the entropy values of the sources are relatively constant. It is not uncommon for hackers to originate illegitimate traffic by using the same UDP/TCP port source number as legitimate users use when generating legitimate traffic to UDP/TCP ports. As a consequence, this attribute is a more accurate reflection of the characteristics of a DDoS attack than any other. Therefore, we use the attribute [IP_{dst} , $Port_{src}$, and $Port_{dst}$] as the basis for aggregating flows based on IP addresses. Lastly, we represent the features of the network traffic at the kth time period in the following manner:

$$X_k = \{H(IP_{dst})`_k , H(Port_{src})`_k , H(Port_{dst})`_k\} \rightarrow \mathrm{VI}$$

4.2.3 Intra-Bayes-based Scheme

An I-BS classifier is a machine learning method that uses binary classifications to classify data. In order to categorize vector X_k as legitimate or illegitimate, it uses stateful traffic features (i.e., traffic features vector X_k) to determine whether vector X_k is legitimate or illegitimate. In the I-BS algorithm, whenever a vector X_k is received, it is categorised based on its probability of being illegitimate. The I-BS informs I-DM once Xk has been classified as an illegal vector, informing I-DM to deploy mitigation measures against this illegal vector Xk. The following sections explain I-BS in detail; first, we take a brief look at the flow representation, and then we go into more depth about the criterion classification of I-BS.

In I-BS, each sample is represented by a vector

$x = (x_1, x_2, x_3)$
$where\ x_1 = H(IP_{dst})` \rightarrow This\ indicates\ the\ entropy\ values\ of\ random\ vriables$;
$x_2 = H(Port_{src})` \rightarrow The\ UDP\ or\ the\ TCP\ port\ of the\ sorce$;
$x_3 = H(Port_{dst})` \rightarrow The\ UDP\ or\ the\ TCP\ port\ of\ the\ destination$;

4.2.3.1 Criterion of classification

Specifically, $illeg$ represents an illegitimate vector and leg represents a legitimate vector. The two classes of vectors are identified by leg and $illeg$ respectively. The class of the k^{th} vector X_k, denoted by c, can be either leg or $illeg$ and is represented as follows:

$$C = \arg\max_{c \in \{leg, illeg\}} p(c|X_k) \text{ where, } p(leg|X_k) + p(illeg|X_k) = 1$$

Thus, the selection criterion is defined as follows:

$$X_k \text{ is illegitimate if } p(illeg|X_k) \geq 0.5 \rightarrow \text{VII}$$

Based on the Bayes theorem (Hoff, 2009), it can be computed that vector X_k has a probability of belonging to class c as follows:

$$P(C = c|X = X_k) = \frac{p(C=c) * p(X=X_k|C=c)}{P(X=X_k)} \rightarrow \text{VII}$$

According to the total probability theorem we have:

$$P(C = c|X = X_k) = \frac{p(C=c) * p(X=X_k|C=c)}{\sum_{c \in \{leg, illeg\}} P(C=c) P(X=X_k \mid C=c)} \rightarrow \text{IX}$$

Therefore, it is possible for the selection criterion equal to Xk to be illegitimate if the following criteria apply:

$$P(C = c|X = X_k) = \frac{p(C=c) * p(X=X_k|C=c)}{\sum_{c \in \{leg, illeg\}} P(C=c) P(X=X_k \mid C=c)} \geq 0.5 \rightarrow \text{X}$$

$H(IP_{dst})`$ , $H(Port_{src})`$ and $H(Port_{dst})`$ are conditionally independent variables given class c. The conditional probabilities $P_k\ (leg)$ and $P_k\ (illeg)$ reflect the conditional probability of the k^{th} vector X_k being receptively legitimate or illegitimate, respectively. As a result of equation (10), the selection criterion can be expressed as follows: If equation (11) is true, then X_k must be illegitimate.

$$\frac{\prod_{k=1}^{n} {p_k}^{X_k}(illeg)\left(1-p_k(illeg)\right)^{1-X_k} p(illeg)}{\prod_{k=1}^{n} {p_k}^{X_k}(illeg)\left(1-p_k(illeg)\right)^{1-X_k} p(illeg) + \prod_{k=1}^{n} {p_k}^{X_k}(leg)\left(1-p_k(leg)\right)^{1-X_k} p(leg)} \geq 0.5 \rightarrow \text{XI}$$

And the selection criterion becomes a package when $p(leg) = p(illeg)$ and X_k is declared to be illegitmate if,

$$\frac{\prod_{k=1}^{n} {p_k}^{X_k}(illeg)\left(1-p_k(illeg)\right)^{1-X_k}}{\prod_{k=1}^{n} {p_k}^{X_k}(illeg)\left(1-p_k(illeg)\right)^{1-X_k} + \prod_{k=1}^{n} {p_k}^{X_k}(leg)\left(1-p_k(leg)\right)^{1-X_k}} \geq 0.5 \rightarrow \text{XII}$$

4.3 Intra-Domain Mitigation scheme

This OF rule installation has been done using the API provided by the SDN controller and is installed into the OF switch that is under attack; these rules have a high priority to monitor suspicious packets and match them against OF rules. The purpose of I-DM is to effectively mitigate the traffic that is illegitimate in order to maintain network security. As part of a flow entry, a meter can be specified; meter entries with different Meter_id numbers are deployed to monitor the speed of defined illegitimate flows by I-BS; if the packet rate exceeds the band (rate limiter), the suspected packets will be dropped.

4.4 Collaboration smart contract

The use of smart contracts is a fundamental part of the system's creation, due to the fact that the system owner (CO), who is planning on collaborating with multiple autonomous systems, will be using the Ethereum platform to create and communicate with the smart contracts. With Ethereum, pseudonymity is maintained while transparency is enabled; therefore, the CO initially generates a pair of keys, primarily a private key, as well as a key-owner external account; the hash value of the key owned by the CO represents its address. Keys are generated using Ethereum's wallets. Once these keys are obtained, it will be possible to use them to create a smart contract as well as to use the embedded functions inside of it. The collaborators will be incorporated into the system through the use of smart contracts that have been designed by the CO. The addresses of collaborators will also be included. Moreover, the CO has the capability of adding collaborators to a contract, managing them transparently, and deleting them if necessary. These collaborations also provide the CO with the capability of reporting malicious IP addresses to the CO. In addition, these collaborations remove those addresses from the CO's database as a result.

Below are the functions associated with the Collaboration contract that will be created will include the following:

- As we already mentioned, the primary function, which is exclusive to the CO, entails adding collaborators to the process. This function relies both on the information provided by the collaborator as well as the previously created external account. A few of the capabilities that are included in the application include the capability of inserting timestamps while adding collaborators. Once the credentials of the owner are validated, then the entire process of adding collaborators to the process can begin.
- The other function that is exclusive to the creator of the collaboration contract is the one that pertains to deleting collaborators from the collaboration contract. To implement this, both the collaborator's details, as well as an external account created previously, are required.
- In addition to creating the collaboration contract, the creator of the collaboration contract has the ability to change the status of the collaboration contract (active/inactive) as well.
- Adding a record to the database of the reported malicious IP addresses can be done by either the collaborator who has been authenticated as well as the CO who has access to the database.
- In the same way, the CO or an authenticated collaborator can also access the function to remove potentially harmful IP addresses from the database by using the function to remove_IP addresses.
- Among other things, it checks whether the collaborator has an Externally Owned Account (c.EOA) in the collaboration contract. If it exists in that contract, it returns a

true value, otherwise, it returns false if the collaborator does not exist in that contract. A call to this function will be executed when the owner is trying to add or remove a collaborator from the collaboration contract.

- In this function, you can pass the IP address of the record as an input and it will return true if the IP address of the record has been included in the collaboration contract; otherwise, it will return false. When one of the collaborators tries to add or remove a record from or from the collaboration contract, this function will be invoked at the moment when one of the collaborators attempts to do so.

Additionally, the algorithm for the multiple functionalities of the collaboration contract is as given below:

```
Input: r.IP
Output: bool

if recordsAdr.length == () then
    return false;
else
    if recordsAdr[records[r.IP].index]==r.IP;
    then
        return true;
    else
        return flase;
    end
end
```

Figure 5 Algorith 2: isRecord().

```
Input: c.EOA
Output: bool

if collaboratorsAdr.length == () then
    return false;
else
    if collaboratorsAdr[collaborators[c.EOA].index]==c.EOA
    then
        return true;
    end
end
```

Figure 4 Algorithm 1: isCollaborator().

```
Input : c.EOA
Output: null
if msg.sender is not owner then
    throw;
end
if status is not true then
    throw;
end
if isCollaborator(c.EOA) == false then
    throw;
else
    rowToDelete ← collaborators[ c.EOA ].index ;
    keyToMove ← collaboratorsAdr[length-1] ;
    collaboratorsAdr[rowToDelete]=keyToMove ;
    collaborators[keyToMove].index=rowToDelete ;
    collaboratorsAdr.length - - ;
    emit CollaboratorRemoved(c.EOA) ;
    numberOfCollaborators - - ;
end
```

Figure 7 Algorithm 4: removeCollaborator().

```
Input : c.EOA, c.Infos
Output: null
if msg.sender is not owner
    then
    throw;
end
if status is not true
    then
    throw;
end
if isCollaborator(c.EOA) == true
    then
    throw;
else
    length ← collaboratorsAdr.push(c.EOA) ;
    collaborators[c.EOA]←
    Collaborator(c.EOA,c.Infos, now, length-1) ;
    emit CollaboratorAdded(c.EOA, c.Infos) ;
    numberOfCollaborators++ ;
end
```

Figure 6 Algorithm 3: addCollaborator().

```
Input : r.IP
Output: null
if msg.sender is not Collaborator then
throw;
end
if status is not true then
throw;
end
if isRecord(r.IP) == false then
throw;
end
if records[r.IP].submitter is not equal msg.sender then
throw;
else
rowToDelete ← records[ r.IP ].index ;
keyToMove← recordsAdr[length-1] ;
recordsAdr[rowToDelete]=keyToMove ;
records[keyToMove].index=rowToDelete;
recordsAdr.length- - ;
emit RecordRemoved(r.IP, msg.sender);
numberOfRecords- - ;
end
```

Figure 9 Algorithm 6: removeRecord().

```
Input : r.IP
Output: null
if msg.sender is not Collaborator then
throw;
end
if status is not true then
throw;
end
if isRecord(r.IP) == true then
throw;
else
length ← recordsAdr.push(r.IP) ;
records[r.IP]← Record(r.IP, msg.sender, now,
length-1) ;
emit RecordAdded(r.IP, msg.sender);
numberOfRecords++ ;
end
```

Figure 8 Algorithm 5: addRecord().

```
Input : status
Output: null
if msg.sender is not owner then
throw;
end
if status is true then
status ←false ;
emit StatusChanged(‘‘Smart Contract Deactivated’’)
;
end
if status is false then
status ← true ;
emit StatusChanged(‘‘Smart Contract activated’’) ;
end
```

Figure 10 Algorithm 7: chnageStatus().

5 Implementation

It is extremely difficult for a victim to stop a DDoS attack by blocking some or all of the sources that are generating large amounts of traffic at the same time when a DDoS attack is conducted due to a massive influx of traffic that comes from several sources simultaneously. It is the combination of threat actors, controllers, bots, and a victim server that leads to a successful DDoS attack. The Zombie Botnet is established when a master element interacts with agents through handlers, which leads to the establishment of an ecosystem of zombies. The core of

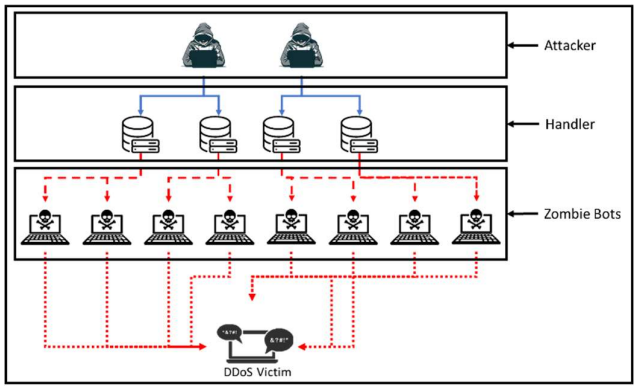

Figure 11 DDoS Components.

this framework can be attributed to the use of these zombie botnets, which are constantly trying to infiltrate target systems according to a series of commands and instructions that they receive from their masters. As a result, complicit botnets are designed to work in compliance with the instructions they are given by their controllers. In the IoT environment, there are many scanning methods that can be used to identify a vulnerable machine, but there are also many methods by which a DDoS attack can be conducted. It is important to understand that these botnets generally maintain a steady but aggressive offensive strategy until the victim has been infected.

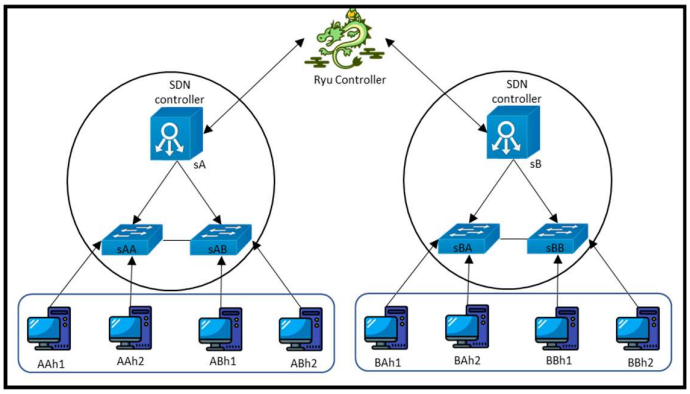

Figure 12 Framework.

Our goal is to model a realistic network environment by using one of the most widely used SDN emulation tools called Mininet (Mininet: An Instant Virtual Network on your Laptop, 2022), which is a tool for modelling a SDN environment. Using the Open vSwitch (Open vSwitch, 2022) virtual switch, the Mininet virtual machine constructs realistic virtual networks using Linux containers and virtual OF-switches in order to create realistic virtual networks. In this instance, Mininet will be installed on a VirtualBox virtual machine (Virtual Box, 2022), in which access to the Internet will be provided through Network Address Translation (NAT), and on the virtual machine itself a host-only adapter will be established to facilitate communications with the host machine. It will be used Secure Shell (SSH) to run different software applications on the virtual machine at the same time. We will also be utilising a ryu controller, Ryu is a software-defined networking framework that uses components to build software-defined networks. As a result of Ryu's well-articulated APIs, developers are able to create well-structured network management and control applications easily by combining software components with well-designed APIs. There are a variety of protocols (such as OpenFlow, Netconf, OF-config, etc.) that Ryu supports for managing network devices (component-based software defined networking framework Build SDN Agilely, 2017), connected to the SDN controller, the ryu controller will be monitoring the flow of packets and will work in-hand with the REST application. The host computer in the testing environment will be connected to the Ethereum network via the Geth client (v1.10.21-stable) (Go Ethereum, 2022), which will operate in conjunction with the network monitor and FloodLight (Floodlight Controller, 2018)a SDN controller, in order to connect the machine to the Ethereum network. Additionally, we will be utilizing the Ganache, which is a popular private test simulator that has been developed for the purpose of testing and tracking the gas used by the smart contracts when they are called by their functions during testing. As a result, we are able to determine how much ether will be consumed during the simulation process.

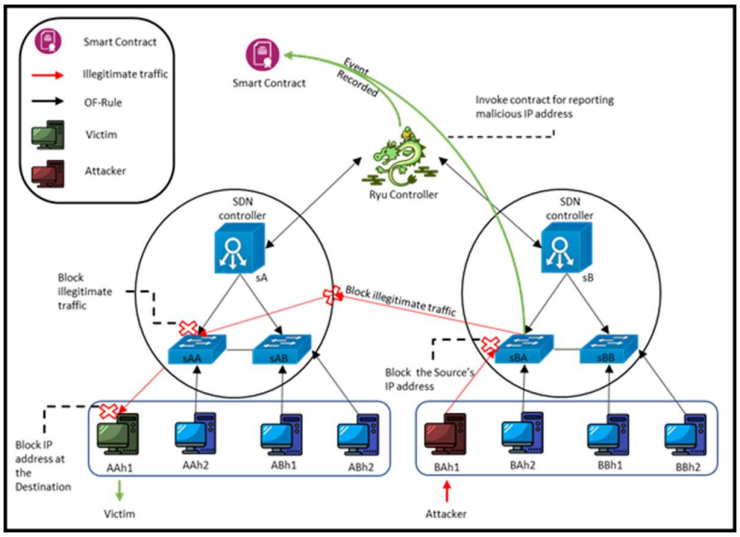

Figure 13 Architecture of inter-domain DDoS mitigation scheme.

In order to carry out this experiment, a victim registered under this domain will be subjected to a series of DDoS attacks as part of the experiment. As a safety precaution, I-DM should make sure the victim is kept inside the domain if the I-BS discovers the attack, and it should utilize the OF protocol to prevent any unauthorised traffic from entering the domain. A smart contract that triggers the SDN controller to alert suspicious IP addresses should also be triggered by the smart contract.

The present OF network consists of four hosts; each of these hosts is capable of launching an attack against a victim within the domain. Our aim is to test the scalability of the proposed solution, which means that we will be able to add more hosts to each domain if we choose to. sFlow uses a sampling rate of 1/64 in order to sample data. Scapy's Python library is utilized in the attack script in order to generate both DDoS flooding attacks and simulated legitimate traffic in order to flood the victim with DDoS flooding attacks. In order to simulate DDoS attacks (i.e., UDP/ICMP DDoS attacks), we use the Hping3 command line. Our next section will discuss the results of our experiments with a DDoS mitigation scheme.

There are four segments that make up the testbed, and they are as follows:

a) As part of the Static Flow Pusher API, FloodLight provides a simple way to connect a static flow pusher with an OF controller, and this connection may then be terminated by using Static Flow Pusher.
b) Data centre networks can be monitored using sFlow network monitors, which provide switch port monitoring for switch ports. As part of the implementation of the I-ES and I-BS, REST apps will be used (Anon., 2022), and
c) A bandwidth setting of 1 Gbps will appear to be set for the OF switch.
d) As the SDN controller is connected to the Ryu controller, it also helps in monitoring packets transmitted via the network once the SDN controller has been configured.

A network consists of more than 20 machines, and several hosts are used to emulate an intrusion on the target machine, which is hosted within the domain, in order to emulate the intrusion on the target machine. The attack software will use the Scapy Python library (Scapy, 2022) in order to create simulated genuine traffic as well as the DDoS flooding attack traffic from zombie servers going to the target server. The DDoS attempts will be emulated by using the Hping3 (Tool Documentation, 2022) command line.

6 Evaluation

A compilation of multiple intra-domain DDoS mitigation schemes has been proposed by the author (El Houda, Hafid, & Khoukhi, 2019), in the context of the Software Defined Network, including: "Intra Entropy-based Scheme (I-ES); Intra Bayes-based Scheme (I-BS); Intra-domain Mitigation (I-DM).

With the help of sFlow, the I-ES scheme measures the randomness of the data within the domain, whereas the I-BS scheme utilizes the entropy value of the packets to classify them as fraudulent, and the I-DM scheme effectively mitigates packets that flow inside the domain by using sFlow.

After the smart contract is deployed, it will be able to execute itself without the need for any manual intervention from the user. We use the truffle framework as well as the remix IDE (Vermouth, et al., 2022) to execute the deployment process. The remix IDE is a powerful toolkit which provides users with a wide range of tools for contract development and experimentation with Ethereum. The table below demonstrates the contraction creation transaction on the official Görli Testnet Network.

Table 2 Details of the Collaboration smart contract transaction.

Details of Collaboration Contract creation transaction in the Görli testnet network	
TxHash	0x55c835c4fecd72537057b131b04e2fe67ac50227357da7fe07f012ebeed28e24
Block Height	8043295
Timestamp	14 days 1 hr ago (Nov-29-2022 05:09:24 PM +UTC)
From	0x3a312d59e8a1ab214b317d87e9b7aef95f6b87d3
To	0xfa29bc4bd5e629f978f8d2c335525e874c64bfa5
Gas Used by Tx	1,119,303

As an application running atop the controller, ES is run as part of sFlow as an API. In order to calculate the flow's entropy, the traffic information collected from the flow is used. The effect of an attack on the domain of a victim is that the number of packets sent from the victim's IP address (dubbed ip_dst), results in the clustering of packets with the same IP address (dubbed ip_dst), namely the victim's IP address. Ip_dst will have a much more dispersed probability distribution if the victim's network is in a normal state, in contrast to the normal state of the victim's network. An increase in the entropy value will result in a more dispersed probability distribution of ip_dst while a decrease in the entropy value will result in a more concentrated probability distribution of ip_dst.

In Figure xxx, the illegitimate traffic generated by the attacker sustained over xxx packet attacks per second within the domain of the victim when the control is disabled. However, when I-BS determines that the incoming traffic is illegitimate, the illegitimate traffic is blocked, and the controller is notified to mitigate the attack using I-DM.

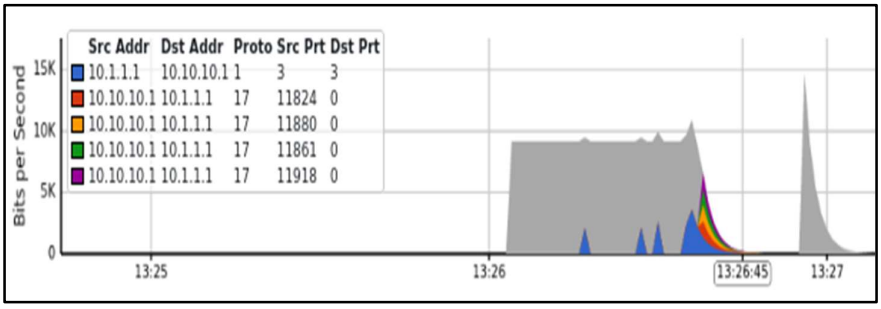

Figure 14 DDoS attack simulation and mitigation.

According to Figure 14, mitigation operations occur within a time period of less than 50 seconds.

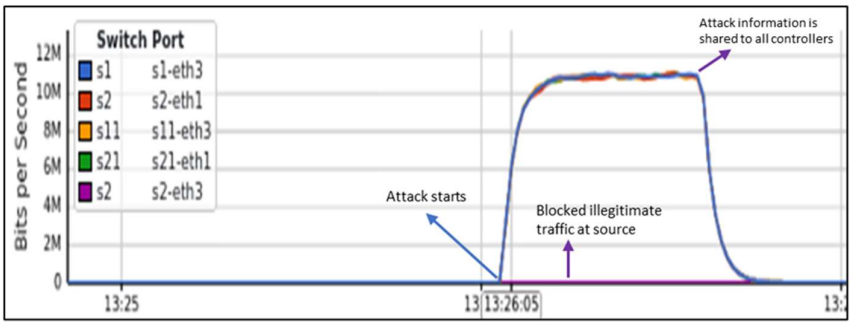

Figure 15 Mitigation.

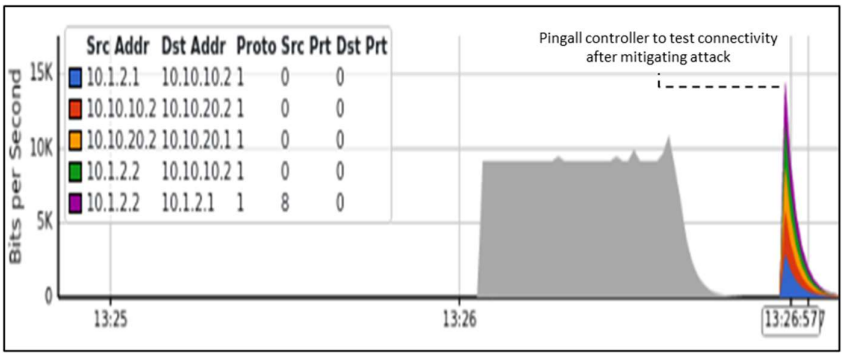

Figure 16 Connection between Domains

I-ES is designed to monitor the traffic information of the victim's domain over a period of time during which the monitoring interval will take place, during the course of which the victim's domain traffic will be monitored using I-ES. Furthermore, the scheme is used to investigate how long it takes for the intra-domain mitigation scheme to detect and mitigate the simulated DDoS attack both after and before the creation of the system for detecting and mitigating the DDoS attack. In order to test the strength of the I-ES, a simulation of an attack will be carried out within 250 seconds of interval time. As a way of measuring the performance of the I-BS, we will use the following formula:

$$DR = \frac{TP}{TP + FN} \quad \begin{matrix} DR = Detection\ rate \\ TP = True\ Positive \\ FN = False\ Negative \end{matrix} \quad ; \quad FPR = \frac{FP}{TN + FP} \quad \begin{matrix} FPR = False\ Positive\ Rate \\ TN = True\ Negative \\ FP = False\ Positive \end{matrix}$$

- The term TP refers to a flow identified as illegitimate that has been correctly identified as being unauthorised.
- It is represented by FN when an illegitimate flow has been classified as legitimized.
- TN represents those legitimate flows, which are defined as legitimate flows, and which fall into the legitimate flow category.
- An FP represents legitimate flows identified as illegitimate flows.

The experiment results showed a detection rate of 100% and successfully blocked the attackers IP address with a FPR of around 29%.

6.1 Characteristics

This project aims to provide a secured, easy-to-deploy, low-cost, efficient and flexible solution to mitigate DDoS attacks at the inter-domain level as well as blockchain, which uses a smart contract to address this issue, in order to be easily and efficiently deployed.

6.1.1 Flexibility

A two-tiered approach to flexibility is offered by the project:

i. Collaboration enables an organization (contract owner) to add and remove collaborators easily to and from the system by using the functions addCollaborator() and removeCollaborator() used by the system. As with the addRecord() and removeRecord() functions, collaborators can easily add/remove records from/to the system using the addRecord() and removeRecord() functions.

ii. In addition, the collaborative system gives organizations (contract owners) the flexibility of joining or leaving the system easily according to their needs. An organization is required to deploy the Collaboration contract before it can join the system and become part of it. This function provides the organization with the capability of deactivating a contract and leaving the system as a result using the ChangeStatus() function. Everyone in the Ethereum network has the ability to verify all these updates.

6.1.2 Low cost

It will be necessary to calculate the costs associated with establishing a collaboration contract as well as the costs associated with executing each function that will be used to verify whether the system that was created is actually an efficient and low-cost system. In order to

calculate the cost of the system, each Gwei unit is set to 10^9 *wei*, and 1 ether unit is equal to 10^{18} *wei*. Hence the value of 1 *Gwei* = 10^{-9} *Ether*, where the value of 1 *Ether* is equivalent to approximately 1,240€ as of today (*15/12/2022).* According to our comparison of the creation and deployment costs of the created system with those of the system developed by the author (El Houda, Hafid, & Khoukhi, 2019) on the Ropsten test network, our project costs more due to the fact that ether's value has increased over the last few years in comparison to EUR.

Table 3 Costs associated with creating and operating smart contracts.

Function	Gas used	Actual cost(ether)	Euro (EUR €)
Creation	1883238	0.001883238	2.28
addCollaborator()	155033	0.000155033	0.19
removeCollaborator()	34179	0.000034179	0.041
addRecord()	151733	0.000151733	0.18
removeRecord()	33273	0.000033273	0.040
changeStatus()	31733	0.000031733	0.038
renounceOwnership	30507	0.000030507	0.037

However, the system created does not show a significant difference in the amount of gas used to perform the functions and, in fact in all the cases, saves the amount of gas that is used to perform the functions when tested on ganache as well as the faster testnet (Görli).
We have deployed the collaboration contract on the Ethereum Görli Testnet with the following address:

- ✓ Owner's account address: 0x3a312d59e8a1ab214b317d87e9b7aef95f6b87d3
- ✓ Collaboration contract address: 0xf3d269eE51BbAb9b9F0fa3c6C6C8e1115d1b961E
- ✓ Using the URL, the transaction for the creation and deployment can be seen: https://goerli.etherscan.io/address/0xf3d269ee51bbab9b9f0fa3c6c6c8e1115d1b961e

6.1.3 Security

Reporting suspicious IP addresses can only be done by authorized collaborators who are granted permission to do so. The smart contract is able to achieve this through the use of modifiers. For instance, the modifier "OnlyOwner" enables the owner of the contract to execute the addCollaborator() and removeCollaborator() functions as well as the changeStatus() function only on behalf of the owner of the contract. The operation of these functions will be halted and no action will be recorded on the blockchain if an uninvited or malicious user tries to execute them by adding illegitimate collaborators in order to report fake IPs or by removing legitimate collaborators in order to report legitimate IPs.

6.1.4 Analysis

The first advantage of the project is that pseudonymity is preserved, and the identities of collaborators cannot be traced back. A key requirement of the collaboration project is that pseudonymity is preserved in order to prevent DDoS attacks from being launched against collaborators. Furthermore, due to the fact that the project can be successfully deployed and run on Ethereum, it does not suffer from a single point of failure problem as well. Further, the collaboration system is a decentralized scheme, meaning that a centralized authority (or a third party) is not required to maintain it; thus, it eliminates the need for a centralized authority. It is

guaranteed that the records, which are recorded on the blockchain, are reliable and available. Compared to other mitigation methods discussed other than (El Houda, Hafid, & Khoukhi, 2019), this project combines both intra-domain and inter-domain mitigation techniques, which is a two-tier method for mitigating DDoS attacks.

7 Conclusion and Future Work

In this paper, we propose a blockchain-based framework that could be used for addressing both intradomain and interdomain DDoS mitigation at the same time Hence since we have now successfully simulated and mitigated the attack on a private network, we could test the projects capabilities on an official public Ethereum network and test its limits. As a result, the I-ES and I-BS schemes will be combined within the intra-domain to detect and mitigate fraudulent activity in real-time, while the I-DM scheme will be used to detect and mitigate fraudulent activity within the intra-domain. A smart contract-based architecture based on Ethereum's smart contract technology is presented in order to facilitate collaboration among peers collaborating on SDN-based domains, hence further we could test the deployment of the smart contract on a public Ethereum server and connect it with the project to receive accurate data on the blockchain network. Last but not least, the collaboration contract will be tested and evaluated on Ethereum's official test network Görli.

7.1 Smart Contract Protection

Ethereum, Fabric, and a wide range of other Blockchain platforms use smart contracts as an integral part of their technology. Using Ethereum smart contracts, specific rules can be enforced on IoT devices in order to manage and control them. These rules can be enforced in accordance with access policies that have been implemented by the Ethereum platform. A DDoS attack can have devastating effects on a system and it is imperative to use the security of Blockchains and smart contracts in order to minimize the damage. On the other hand, there is scant attention paid to the fact that smart contracts themselves are vulnerable to security issues and in need of protection. This is necessary as smart contracts play an imperative role in the creation as well as the functioning of the project. In the event of an attack on a smart contract, there will be economic losses and the smart contract that has been attacked will be unable to function. In light of this, it is pertinent to research how smart contracts can be protected in the future.

7.2 The introduction of high-speed cellular networks such as 5G and 6G

It is increasingly evident that 5G/6G networks have enabled seamless communication between different platforms, e.g., SDIoT, SDIoT-Edge, blockchain, etc., and that these networks are increasingly being used as a communications resource. The development of standards and protocols, however, will provide researchers with the opportunity to join forces in order to be able to coexist wireless SDN networks with IoT networks in the future. By separating the control layer from the application layer, it can be possible to implement the security features of blockchain in the IoT edge devices so that the security features of blockchain can be utilized. As a result of this, various components within the cellular network will be implemented within the network, such as information authentication and protection as well (Rafique, et al., 2020). Additionally, the use of blockchain technology in IoT networks will also bring increased transparency to data exchanges while providing layers of security. This will help to ensure that transactions remain safe and secure. This in turn will provide users

with a significantly higher level of trust and assurance in the data they exchange, leading to more efficient and reliable networks.

7.3 The Internet of Things and Edge Computing

As IoT devices possess limited computational and networking capabilities, they often rely on traditional methods to connect to the internet. Consequently, when integrating them with blockchain technology, they present quite a few challenges that need to be overcome. Due to their limited computational power and limited battery life, IoT devices are unable to process Proof-of-Work consensus algorithms. As a result, they are not suitable to be used for securing a blockchain network, since blockchains require a large amount of computation power to reach consensus, which is why they cannot be used for this purpose. Additionally, IoT devices have limited networking capabilities, so they can have difficulty in creating and maintaining a distributed network. This further complicates their integration with blockchain technology. To this end, understanding the potential risks and vulnerabilities associated with the integration of IoT devices and their limited computational resources of IoT devices is essential for developing robust security protocols for their long-term protection.

8 References

Alharbi, T., Aljuhani, A., Liu, H., & Hu, C. (2017). Smart and Lightweight DDoS Detection Using NFV. *Proceedings of the International Conference on Compute and Data Analysis*, 220-227. doi:10.1145/3093241.3093253

Classification: ROC Curve and AUC. (2022). Retrieved 08 05, 2022, from https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

component-based software defined networking framework Build SDN Agilely. (2017). Retrieved from Ryu-SDN: https://ryu-sdn.org/

De Donno, M., Dragoni, N., Giaretta, A., & Mazzara, M. (2018). AntibIoTic: Protecting IoT Devices Against DDoS Attacks. (P. Ciancarini, S. Litvinov, A. Messina, A. Sillitti, & G. Succi, Eds.) *Proceedings of 5th International Conference in Software Engineering for Defence Applications*, 59-72. doi:https://doi.org/10.1007/978-3-319-70578-1_7

El Houda, Z., Hafid, A., & Khoukhi, L. (2019). Co-IoT: A Collaborative DDoS Mitigation Scheme in IoT Environment Based on Blockchain Using SDN. *2019 IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). Waikoloa, HI, USA: IEEE. doi:10.1109/GLOBECOM38437.2019.9013542

Ethereum Virtual Machine. (2022). Retrieved 08 03, 2022, from https://ethereum.org/en/developers/docs/evm/

Etherscan. (2022). Retrieved 08 07, 2022, from https://goerli.etherscan.io/

Ferguson, P., & Senie, D. (2013, 03 02). *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing.* Retrieved from IETF Datatracker: https://datatracker.ietf.org/doc/rfc2827/

Floodlight Controller. (2018). Retrieved 08 07, 2022, from https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/overview?homepageId=1343545

Ganache. (2022). Retrieved 08 07, 2022, from https://trufflesuite.com/docs/ganache/

Giotis, K., Apostolaki, M., & Maglaris, V. (2016). A reputation-based collaborative schema for the mitigation of distributed attacks in SDN domains. *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (pp. 495-501). Istanbul, Turkey: IEEE. doi:10.1109/NOMS.2016.7502849

Go Ethereum. (2022). Retrieved 08 07, 2022, from https://geth.ethereum.org/

Hanan Mustapha, A. M. (2018). DDoS attacks on the internet of things and their prevention methods. *ICFNDS '18: Proceedings of the 2nd International Conference on Future Networks and Distributed Systems, 4*, 1-5.

Hoff, P. D. (2009). *A first course in Bayesian statistical methods* (Vol. 580). Springer.

inmon: sFlow-RT. (2022). Retrieved 08 07, 2022, from https://sflow-rt.com/reference.php

Lim, S., Ha, J., Kim, H., Kim, Y. J., & Yang, S. H. (2014). A SDN-oriented DDoS blocking scheme for botnet-based attacks. *2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN)*, (pp. 63-68).

Maglaris, K. G. (2014). Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. *Computer Networks*, 122-136.

Mehdi, S. A., Khalid, J., & Khayam, S. A. (2011). Revisiting Traffic Anomaly Detection Using Software Defined Networking. In: Sommer, R., Balzarotti, D., Maier, G. (eds) Recent Advances in Intrusion Detection. *Lecture Notes in Computer Science, vol 6961. Springer* (pp. 161-180). Berlin: Heidelberg.

Mininet: An Instant Virtual Network on your Laptop. (2022). Retrieved 08 07, 2022, from http://mininet.org/

Open vSwitch. (2022). Retrieved 08 07, 2022, from https://www.openvswitch.org/

Rashidi, B., Fung, C., & Bertino, E. (2017). A Collaborative DDoS Defence Framework Using Network Function Virtualization. *IEEE Transactions on Information Forensics and Security, 12*(10), 2483-2497. doi:10.1109/TIFS.2017.2708693
Rodrigues, B., Bocek, T., Lareida, A., Hausheer, D., Rafati, S., & Stiller, B. (2017). A blockchain-based architecture for collaborative DDoS mitigation with smart contracts. *IFIP International Conference on Autonomous Infrastructure, Management and Security.* Cham Springer. doi:https://doi.org/10.5167/uzh-146085
Scapy. (2022). Retrieved 08 07, 2022, from https://scapy.net/
Shannon, C. (1951). Prediction and entropy of printed English. *The Bell System Technical Journal, 30*(1), 50-64. doi:10.1002/j.1538-7305.1951.tb01366.x
Simpson, S., Shirazi, S. N., Marnerides, A., Jouet, S., Pezaros, D., & Hutchison, D. (2018). An Inter-Domain Collaboration Scheme to Remedy DDoS Attacks in Computer Networks. *IEEE Transactions on Network and Service Management, 15*(3), 879-893. doi:10.1109/TNSM.2018.2828938
Solidity. (2022). Retrieved 08 07, 2022, from https://docs.soliditylang.org/en/develop/
Spathoulas, G., Collen, A., Pandey, P., Nijdam, N. A., Katsikas, S., Kouzinopoulos, C. S., . . . Tzovaras, D. (2018). Towards Reliable Integrity in Blacklisting: Facing Malicious IPs in GHOST Smart Contracts. *2018 Innovations in Intelligent Systems and Applications (INISTA)* (pp. 1-8). Thessaloniki, Greece: IEEE. doi:10.1109/INISTA.2018.8466327
Stankovic, J. A. (2014). Research Directions for the Internet of Things. *IEEE Internet of Things Journal, 1*(1), 3-9. doi:10.1109/JIOT.2014.2312291
Steinberger, J., Kuhnert, B., Sperotto, A., Baier, H., & Pras, A. (2016). Collaborative DDoS defense using flow-based security event information. *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium* (pp. 516-522). Istanbul, Turkey : IEEE. doi:10.1109/NOMS.2016.7502852
Tool Documentation. (2022). Retrieved 08 07, 2022, from https://www.kali.org/tools/hping3/
Truffle Suite: Sweet Tools for Smart Contracts. (2022). Retrieved 08 07, 2022, from https://trufflesuite.com/
Vailshery, L. S. (2022). *Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2030*. Retrieved 08 01, 2022, from https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/
Vermouth, A., Aniket, Disu, D., Mertens, F., Izang, J., Husikyan, L., . . . Levreau, Y. (2022). *Remix Project*. Retrieved from Remix: https://remix-project.org/
Virtual Box. (2022). Retrieved 08 07, 2022, from https://www.virtualbox.org/
Wang, R., Jia, Z., & Ju, L. (2015). An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking. *TRUSTCOM '15: Proceedings of the 2015 IEEE Trustcom/BigDataSE/ISPA - Volume 01* (pp. 310-317). USA: IEEE Computer Society.
Welcome to Ethereum. (2022). Retrieved August 01, 2022, from https://ethereum.org/en/
Yu, M., Jose, L., & Miao, R. (2013). Software defined traffic measurement with OpenSketch. *nsdi'13: Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation* (pp. 29-42). Berkeley,CA,USA : USENIX Association.
Zawar, S., Ullah, I., Li, H., Levula, A., & Khurshid, K. (2022). Blockchain Based Solutions to Mitigate Distributed Denial of Service (DDoS) Attacks in the Internet of Things (IoT): A Survey. *Sensors, 22*(3). doi:10.3390/s22031094
Zhang, C., & Green, R. (2015). Communication security in internet of thing: Preventive measure and avoid DDoS attack over IoT network. *Simulation Series, 47*, 8-15.