

# **BLOCKCHAIN APPROACH TO CYBER SECURITY VULNERABILITIES (DDoS MITIGATION)**

**A PROJECT REPORT**

*for*

**INFORMATION SECURITY MANAGEMENT (CSE3502)**

*in*

**B.Tech. – Information Technology and Engineering**

*by*

**KUSHAGRA AGARWAL (18BIT0231)**

**PRIYAL BHARDWAJ (18BIT0272)**

**ROHAN JAIN (18BIT0429)**

*Under the Guidance of*

**Dr. I. SUMAIYA THASEEN**

Associate Professor, SITE



**VIT<sup>®</sup>**  
**Vellore Institute of Technology**  
(Deemed to be University under section 3 of UGC Act, 1956)

**School of Information Technology and Engineering**

June, 2021

Google Drive Link: [Click Here](#)

## **Abstract**

Akamai predicts that by 2021 the average DDoS attack will generate 1.5 Tbps of network traffic. Hence for the scope of this project, our aim is to mitigate DDoS attacks using Blockchain. Blockchain is a decentralized network that allows independent parties to communicate without any third-party involvement. In order to protect networks from DDoS attacks, organizations can be made distributed between multiple server nodes that provide high resilience and remove the single point of failure. The reason why we are using blockchain is because it can be used to deploy a decentralized ledger to store blacklisted IP's and eliminate the risk of a single point of failure.

**Keywords** – DDoS, Blockchain, Mitigation, Decentralized Ledger, IP

## **I. INTRODUCTION**

Over the past few years, a new variant of user has entered the world of the internet, commonly known as a smart device, also referred to as Internet of Things (IoT) devices, monitoring the functionality and operations of connected utilities. Despite of enough use cases, attackers are making use of them to launch some massive cyber attacks called distributed denial-of-service (DDoS) attacks.

### **Challenges with current DDoS solutions:**

In the past few years, a rise in DDoS attacks has been observed. As per the recent reports, 43% of organizations experienced burst attacks, but the rest were unaware of whether they were attacked. Attackers are adapting several emerging techniques and complex tactics to compromise the target network. On February 28, 2018, GitHub, was hit with the largest-ever DDoS attack, recorded at 1.35 Tbps. As DDoS attack falls under cyber threat category that makes it unfeasible to deploy any security prevention mechanism as system vulnerabilities are under control of organizations but threats can't be controlled.

### **Challenges with existing Machine Learning based solutions:**

- It is not a decentralized system like blockchain, so we have to inform each server/system individually about the attack. This can be time consuming. e.g. Banks

- If appropriate prediction model is not chosen then the rate of false positives can be high. This could lead to blacklisting genuine IP's.
- It is a time-consuming approach due to dataset size, training and testing time etc.

### **Suggested Methodology:**

Blockchain can mitigate DDoS protection. It is a decentralized network that allows independent parties to communicate without any third party involvement. In order to protect networks from DDoS attacks, organizations can be made distributed between multiple server nodes that provide high resilience and remove the single point of failure.

There are two main reasons why we are using blockchain:

- Blockchain technology can be used to deploy a decentralized ledger to store blacklisted IPs.
- Blockchain technology eliminates the risk of a single point of failure.

Thus, we will be using blockchain to mitigate DDoS attacks.

### **Software Requirements:**

- Operating System : Ubuntu
- TCP Flood DDoS Attack : hping3
- Network Analysis : tcptrack
- Blockchain : Ethereum testchain – Remix IDE/Ganache CLI
- Other packages : nmap, iptables

## **II. Literature Survey**

### **[1]. A Secure and Effective Construction Scheme for Blockchain Networks**

Blockchain technology has emerged as a novel distributed ledger technology, facilitating data sharing and system management securely and efficiently without interventions from a central authority. However, blockchain technology alone is not suitable for enterprise-class applications, mainly due to the limitations in capacity expansion and verification speed of blockchain systems. This paper proposes a secure and effective construction scheme for blockchain networks to improve performance and address the effective management concerns of blockchain data based on transaction categories.

## **[2]. Cyber Security through Blockchain Technology**

Blockchain technology has seen adoption in many industries and most predominantly in finance through the use of cryptocurrencies. However, the technology is viable in cybersecurity. This paper looks at several use cases of Blockchain in the cybersecurity industry as envisioned by 30 researchers. It found that most researchers are concentrating on the adoption of Blockchain to protect IoT (Internet of Things) devices, networks, and data. The paper examined the ways highlighted by previous researchers through which Blockchain can afford security to the three problematic areas in IT. Lastly, the paper recommended that future researchers focus on a single Blockchain on which to develop cybersecurity applications to allow for integration and uniformity among solutions.

## **[3]. An Improved Blockchain-Based Authentication Protocol for IoT Network Management**

In the last two decades, numerous IoT solutions have been developed by small, medium sized, and large enterprises to make our lives easier. The rapid expansion of IoT solutions accompanies numerous security concerns because the underlying IoT protocols and communication technologies have not considered security. Recently, blockchain has emerged to become one of the promising technologies that might overcome some of the IoT limitations (security limitations, in particular). This paper surveyed recent security advances to overcome IoT limitations using blockchain. In this paper, it is shown how blockchain attempts to overcome IoT limitations that are related to cyber security have been classified into four categories: end-to-end traceability; data privacy and anonymity; identity verification and authentication; and confidentiality, data integrity, and availability (CIA).

## **[4]. Blockchain as a Service for Software Defined Networks: A Denial of Service Attack Perspective**

Software defined networking (SDN) is one of the most popular network technologies which provides an adaptive, agile and flexible network management and visibility. Although SDN architecture provides manifold benefits but on the same time its dependence on a logically centralized controller leads to the single point of failure. An attacker can easily capture the any forwarding device and restrict the availability of the controller using different prevalent attacks. Distributed denial of service (DDoS) is one of the most popular attack of this category which is quite prevalent in SDN. Here, the aim of the attackers is to inject false script in the open flow tables through malicious switches which multiply exponentially. Therefore, in this paper, a

blockchain as a service framework has been presented wherein BlockSDSec model is designed to provide security as a separate service for the SDN architecture.

#### **[5]. DDoS Mitigation: Decentralized CDN Using Private Blockchain**

Distributed Denial of Service (DDoS) attacks are intense and are targeted to major infrastructure, governments and military organizations in each country. There are a lot of mitigations about DDoS, and the concept of Content Delivery Network (CDN) has been able to avoid attacks on websites. However, since the existing CDN system is fundamentally centralized, it may be difficult to prevent DDoS. This paper describes the distributed CDN Schema using Private Blockchain which solves the problem of participation of existing transparent and unreliable nodes. This will explain DDoS mitigation that can be used by military and government agencies.

#### **[6]. Distributed Denial of Service (DDoS) Mitigation Using Blockchain—A Comprehensive Insight**

Distributed Denial of Service (DDoS) attack is a primary threat impeding service to real requests on any network. It is estimated that these attacks will double, in the upcoming 2 years. Blockchain has emerged as a promising and viable technology for DDoS mitigation. The integral and fundamental characteristics of blockchain such as decentralization, immutability, integrity, anonymity and verifiability have proven to be strong candidates, in tackling this cyber threat. This paper discusses different approaches for DDoS mitigation using blockchain in varied domains to date. The paper aims at providing a comprehensive review, highlighting all essential details, strengths, challenges and limitations of different approaches.

#### **[7]. A Study on DDoS Attacks, Danger and its Prevention**

The cause for internet insecurity is related with its design because the primary concern is its functionality rather than its security. Hence many types of attacks and threats are reason for apprehension towards security of internet. Among all the attacks, DDoS (Distributed Denial of service) attacks are those which cause problems to clients, users to access all the benefits of services available to them from server side. The number of DoS and DDoS attacks on the Internet Service Providers has increased sharply in the past few years.

#### **[8]. Impact Analysis of Dos & DDos Attacks**

Denial of Service attacks constitutes one of the major threats, which poses immense threats to the Internet. The researchers have to find details of these type of attacks to avoid the damaging reputation issues. In this paper, an overview of Denial-of-Service (DoS) & Distributed denial-of-service (DDoS) attacks are provided. Real distributed denial-of-service incidents with their financial impact are analysed and finally DOS and DDOS solution is highlighted.

#### **[9]. A Study on Recent Approaches in Handling DDoS Attacks**

In this paper, authors have presented a study on the approaches in handling Distributed Denial of Service (DDoS) attacks. DDoS attack is a type of attack to hamper the accessibility of Internet services and resources. A DDoS attack can initiate from anywhere in the network and usually overcomes the victim server by sending a vast number of packets. A new type of DDoS attack is known as DDoS Reflector attack. This kind of attack is hard to defend as the victim computer is flooded with traffic from other Internet servers, which were not even compromised. This attack exploits the SYN ACKs in response to the TCP SYN requests and other TCP packets. Mostly, attackers misuse this acknowledging packet as a reflector. Quite a few counteractive measures have been proposed by various researchers. This paper tries to discuss the recent contributions to handle the DDoS attacks.

#### **[10]. A study on the impacts of DoS and DDoS attacks on cloud and mitigation techniques**

One of the challenges in cloud computing is providing secure and reliable services. From the past few years, Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are the major threats in cloud security. DoS and DDoS attacks are started by the hacker to make an online service engaged by compromising it with massive traffic from various sources. The aim of this paper is to study the impacts of several types DoS/DDoS attacks in the cloud, and the procedures that they use to affect the services' availability.

#### **[11]. A Lightweight Blockchain based Cybersecurity for IoT environments**

Blockchain which promise to address the security and the privacy preservation issues along with the IoT environments which is also restricted resources i.e. limited capabilities in terms of computation, storage and the energy that deter to adopt the Blockchain. The paper aims to address the mentioned issues and have introduced the Lightweight Blockchain bases Cybersecurity (LBC) for IoT environments. The main aim for the paper is to mitigate the heavy

computational cost that is required in a consensus algorithm to meet the IoT requirements. To provide scalability to the proposed method EBM (Edge Block Manager) and ABM (Aggregation Block Manager) have been introduced.

#### **[12]. Blockchain Based DDoS Mitigation Using Machine Learning Techniques**

In recent times, there are various types of attack like DDoS attack and each attack uses a different protocol and the attacker uses a botnet to execute such attacks. Hence it becomes difficult for organisations to secure themselves from such attacks. The proposed algorithm in the paper will be used in order to eliminate the third parties. The proposed system uses machine learning algorithms to identify the incoming packet is malicious or not and use the Blockchain technology to store the Blacklist. The benefit of Blockchain is that the IP that are blacklisted are stored efficiently. The paper has used three different ML algorithms such as the KNN classifier, Decision tree Classifier and the Random Forest Algorithm.

#### **[13]. Distributed Denial Of Service(DDoS) Mitigation in Software Defined Network using Blockchain**

Now a days the attackers use the DDoS attack methods to target someone as the DDoS attack can generate GBs to TBs of the random data to flood the target. The existing methods for the mitigation have lack of resources and not having the flexibility to prevent from the attack by itself so they are ineffective. The paper proposed an architecture where a smart contract is being deployed in the private blockchain that will also facilitates the DDoS mitigation across multiple network domain. The Blockchain is also used to give additional security service. With Blockchain the rules are distributed among all the hosts with the smart contracts and the shared protection is enabled among all hosts. The idea presents SDN domain and describes how a blockchain with the help of the SDN combination facilitates to avoid the attack like DDoS on the network. This paper also present the collaboration with blockchain and SDN through which a DDoS attack can be mitigated and make the system highly secure and transparent.

#### **[14]. Blockchain for Cybersecurity: A Comprehensive Survey**

Blockchain is a modern-day technology which is used for secure exchange of the digital currency and to perform deals etc. This Paper has given the blockchain architecture that can be used to explain the concept, characteristics and the need of Blockchain for the security. It also highlights the importance and the role of blockchain in the field of cybersecurity for shaping the future. The paper also explains the need of blockchain technology in various fields and to show the advantages over the conventional system. The main purpose of this research paper is

to provide guidance in the field of blockchain technology and to implement blockchain technology in the field of Cyber Security, Cryptocurrency and IoT.

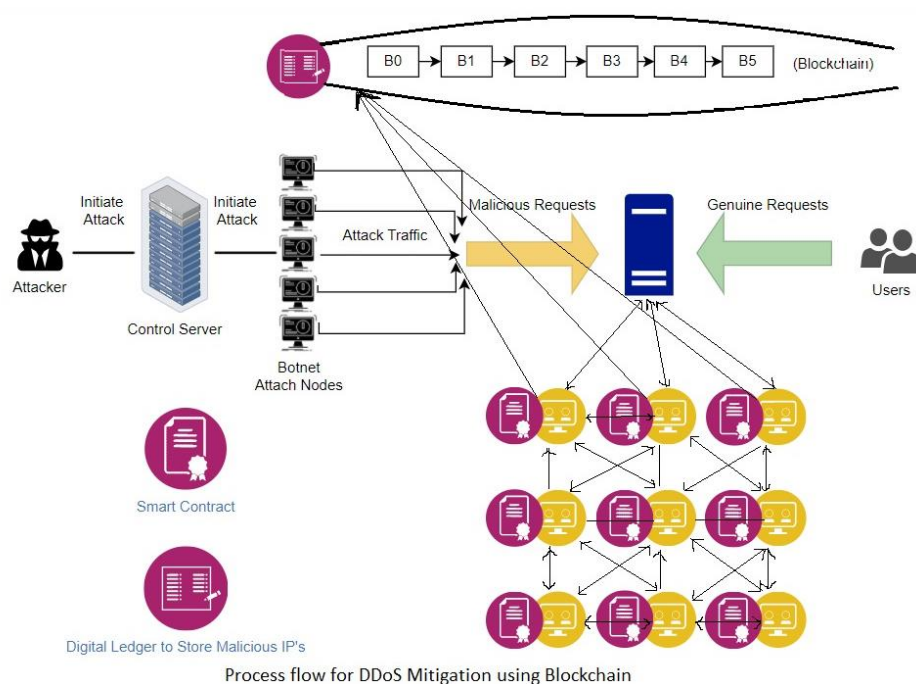
### [15]. A systematic literature review of blockchain cyber security

This research helps in identification of peer to peer reviewed literature that is to utilize blockchain technology for cyber security purposes. It also presents a systematic analysis for blockchain as the most frequently adopted security application. The authors find that the Internet of Things (IoT) have a novel blockchain application and it is same with the networks and the machine visualization. Blockchain also works good with the public key cryptography, web applications and the certification schemes and also the secure storage of the PII (Personal Identification Information). The paper as the systematic review has also shows the future direction for research in blockchain and also education and practices in blockchain like the cybersecurity space such as the security of the blockchain in IoT. Security of blockchain for AI data and the sidechain security. This paper also focus on the existing literature that highlight the use of blockchain as a supporting technology for cyber security applications, which include the areas of privacy, security, integrity and accountability of data, as well as its use in securing network devices.

## REVIEW - 2

### III. PROPOSED MODEL

**High Level Design:** (Tool used: IBM's online.visual-paradigm.com)



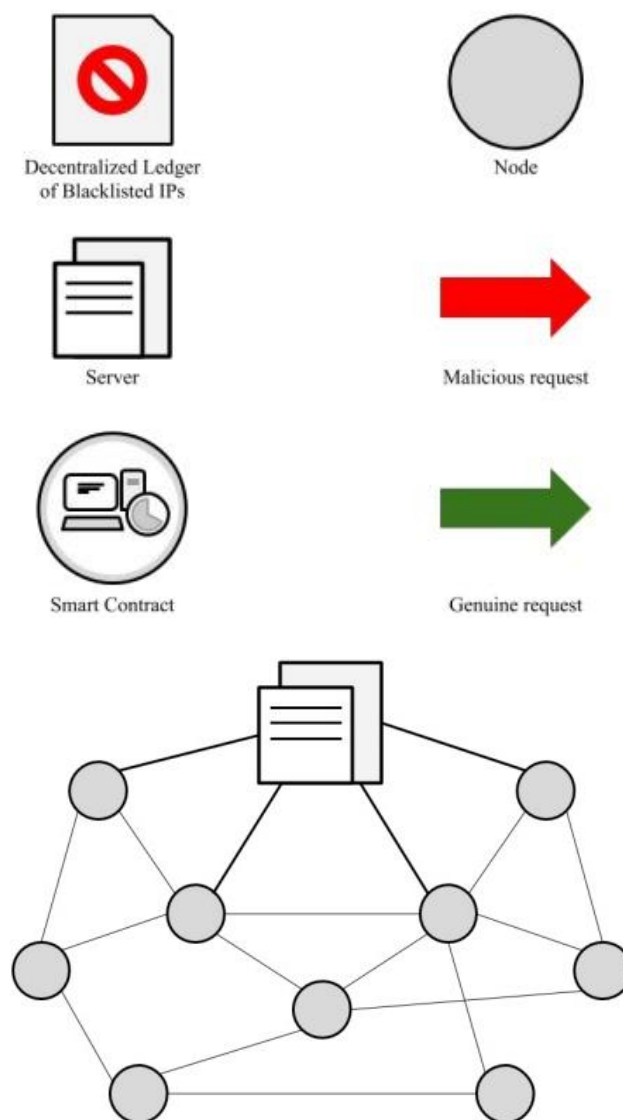


The proposed solution has multiple adjacent nodes to the main server. When under attack, the main server reroutes both the genuine and malicious requests to its adjacent nodes which in turn distributes the requests among their peers.

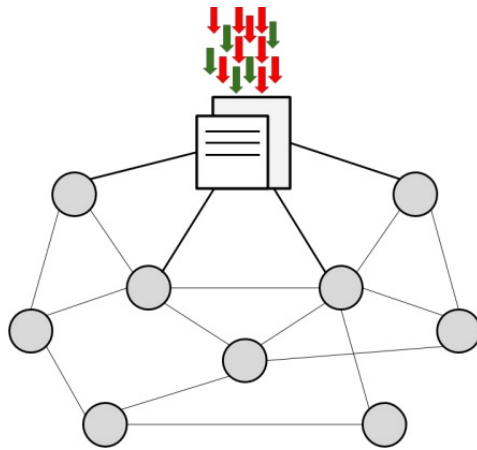
The decentralized Blockchain network then filters out the genuine requests in real time and sends them back to the server, while logging the malicious IPs to be blocked for a certain predefined period of time along with their timestamp in the decentralized ledger.

Every node having access to the decentralized ledger can effectively block requests in future by fetching (IP, timestamp) tuples from the ledger.

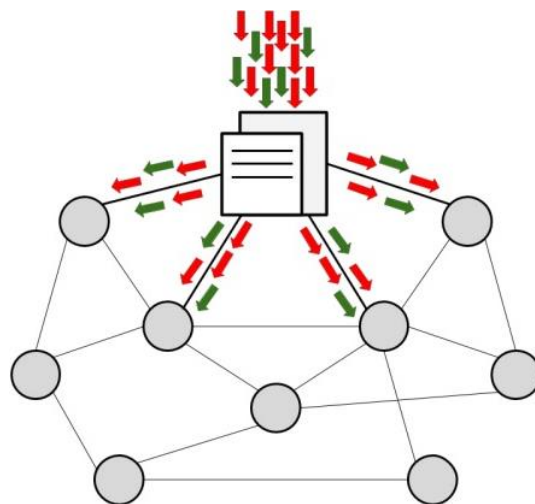
### Low Level Design:



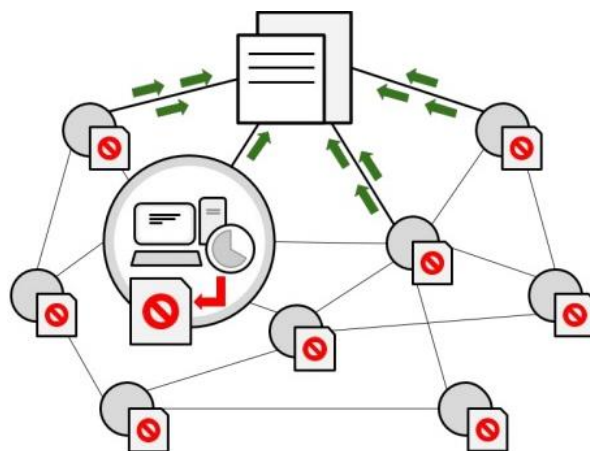
Basic network architecture with main server and multiple adjacent nodes.



Main server faces an attempt of DDoS attack with the number of malicious requests overwhelming the genuine requests.



Main server distributes the requests among its adjacent nodes which in turn distribute them among their decentralized network peers.



Each node has a running smart contract on the Ganache CLI/ Remix IDE blockchain which filters out the malicious requests and logs their IPs in their copy of the decentralized ledger.

## Analyse before hping

kushagra@kushagra-Ubuntu-PC: ~/Desk

kushagra@kushagra-Ubuntu-PC: ~/Desktop/ISM Proj... × kushagra@kushagra-Ubuntu-PC: ~/Desktop/

Client	Server	State	Idle	A	Speed
192.168.0.101:50790	34.107.221.82:80	ESTABLISHED	4s	0	B/s
192.168.0.101:50788	34.107.221.82:80	ESTABLISHED	4s	0	B/s
192.168.0.101:52534	117.18.237.29:80	ESTABLISHED	5s	0	B/s
192.168.0.101:52536	117.18.237.29:80	ESTABLISHED	7s	0	B/s

TOTAL  
Connections 1-4 of 4 Unpaused Unsorted 0 B/s

## Analyse after hping

kushagra@kushagra-Ubuntu-PC: ~

kushagra@kushagra-Ubuntu-PC: ~/Desktop/ISM Proj... × kushagra@kushagra-Ubuntu-PC: ~/Des

Client	Server	State	Idle	A	Speed
117.232.114.117:31558	192.168.0.1:80	SYN_SENT	7s	0	B/s
152.82.114.240:41738	192.168.0.1:80	SYN_SENT	30s	0	B/s
116.29.243.236:87	192.168.0.1:80	SYN_SENT	20s	0	B/s
129.232.51.209:28034	192.168.0.1:80	SYN_SENT	9s	0	B/s
129.152.206.229:214	192.168.0.1:80	SYN_SENT	20s	0	B/s
33.87.46.50:655	192.168.0.1:80	SYN_SENT	20s	0	B/s
1.0.230.225:2675	192.168.0.1:80	SYN_SENT	19s	0	B/s
242.76.252.177:200	192.168.0.1:80	SYN_SENT	20s	0	B/s
51.133.107.42:42051	192.168.0.1:80	SYN_SENT	3s	0	B/s
0.119.228.49:204	192.168.0.1:80	SYN_SENT	20s	0	B/s
81.46.48.252:84	192.168.0.1:80	SYN_SENT	20s	0	B/s
125.218.240.105:88	192.168.0.1:80	SYN_SENT	20s	0	B/s
163.23.220.158:2542	192.168.0.1:80	SYN_SENT	19s	0	B/s
213.105.186.82:10887	192.168.0.1:80	SYN_SENT	16s	0	B/s
117.42.77.215:20869	192.168.0.1:80	SYN_SENT	12s	0	B/s
170.87.46.137:31440	192.168.0.1:80	SYN_SENT	8s	0	B/s
233.51.59.207:82	192.168.0.1:80	SYN_SENT	20s	0	B/s
176.117.36.226:3679	192.168.0.1:80	SYN_SENT	18s	0	B/s
218.102.186.224:742	192.168.0.1:80	SYN_SENT	20s	0	B/s
189.181.192.142:1957	192.168.0.1:80	SYN_SENT	19s	0	B/s
255.217.49.74:13100	192.168.0.1:80	SYN_SENT	15s	0	B/s
230.99.242.46:8406	192.168.0.1:80	SYN_SENT	17s	0	B/s
208.35.255.148:42051	192.168.0.1:80	SYN_SENT	30s	0	B/s
0.94.144.243:100	192.168.0.1:80	SYN_SENT	20s	0	B/s
64.162.5.74:1708	192.168.0.1:80	SYN_SENT	19s	0	B/s
46.202.148.74:47116	192.168.0.1:80	SYN_SENT	28s	0	B/s
162.154.107.196:171	192.168.0.1:80	SYN_SENT	20s	0	B/s
131.23.207.58:1284	192.168.0.1:80	SYN_SENT	20s	0	B/s
209.0.29.149:26323	192.168.0.1:80	SYN_SENT	10s	0	B/s
105.152.251.236:14859	192.168.0.1:80	SYN_SENT	14s	0	B/s
208.226.7.185:85	192.168.0.1:80	SYN_SENT	20s	0	B/s
51.137.195.105:97	192.168.0.1:80	SYN_SENT	20s	0	B/s
144.105.118.118:58202	192.168.0.1:80	SYN_SENT	23s	0	B/s

TOTAL  
Connections 1-33 of 32231 Unpaused Unsorted 0 B/s

## ifconfig

```
kushagra@kushagra-Ubuntu-PC:~/Desktop$ ifconfig
eno1: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether 04:0e:3c:91:50:e0 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 808 bytes 84327 (84.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 808 bytes 84327 (84.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

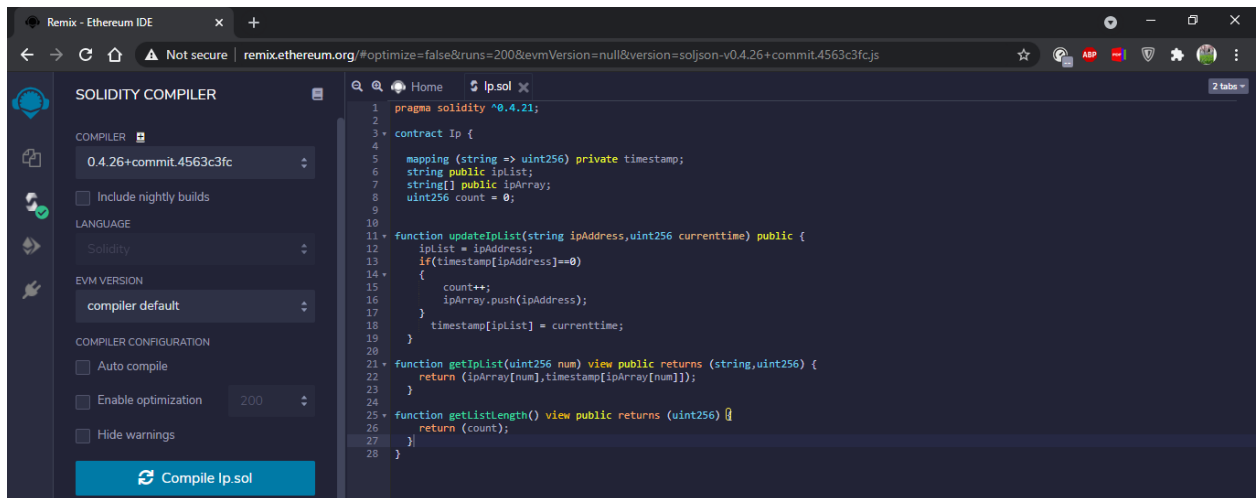
wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.101 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::9b99:2b24:4ca0:17be prefixlen 64 scopeid 0x20<link>
    ether ac:d5:64:ae:de:5d txqueuelen 1000 (Ethernet)
    RX packets 42837 bytes 59505384 (59.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 147455 bytes 13890389 (13.8 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

## hping

```
kushagra@kushagra-Ubuntu-PC: ~/Desktop/ISM Project$ sudo bash flood.sh
[sudo] password for kushagra:
enter interface name
wlo1
enter destination port
80
enter destination ip
192.168.0.1
HPING 192.168.0.1 (wlo1 192.168.0.1): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
--- 192.168.0.1 hping statistic ---
128982 packets transmitted, 0 packets received,
round-trip min/avg/max = 0.0/0.0/0.0 ms
kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project$
```

Compiling the Solidity file in Remix IDE.

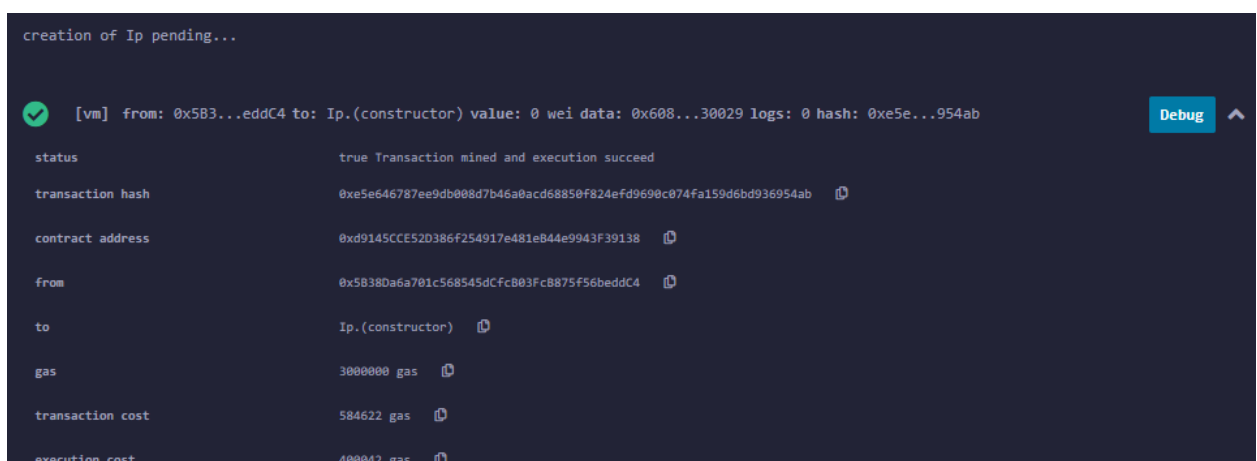
It automatically assigns the compiler version



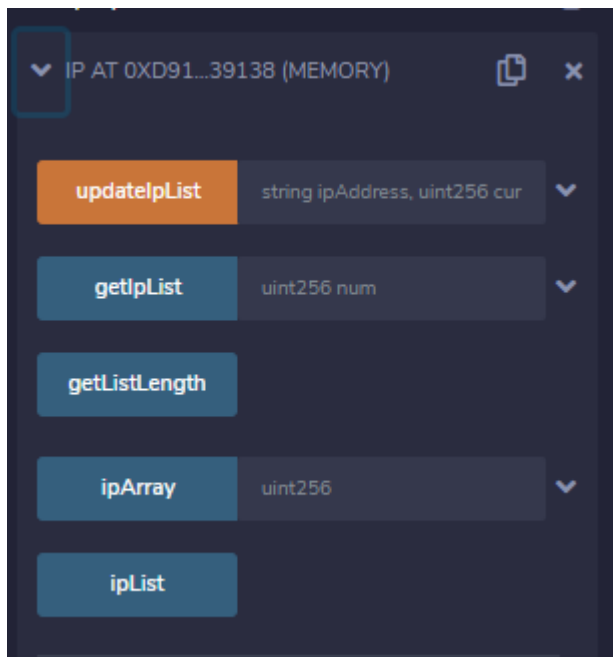
Deployed the file on the address provided by Remix IDE.



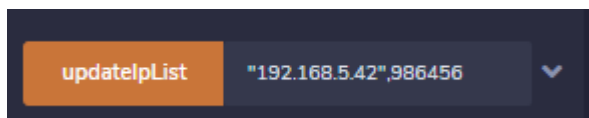
A transaction hash is created after we deploy the contract.



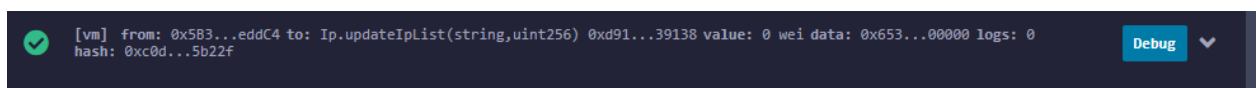
After Deploying we can see and use the function written in the solidity file.



Now we will put one of the IP as INPUT in the updateIpList function and that IP will be added to the smart contract.

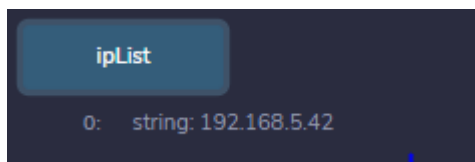


After we execute this function a transaction hash is created and this IP is added to the smart contract.



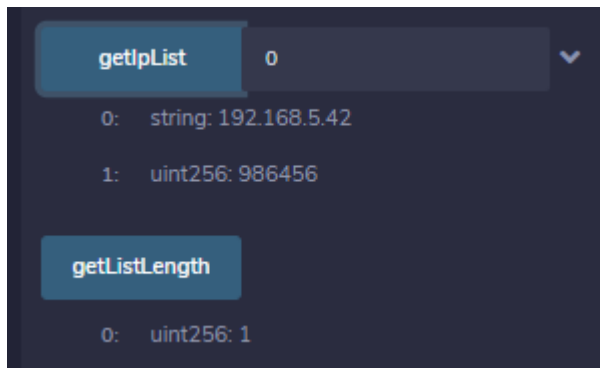
Transaction hash for the above function.

We can see the list of IP in the smart contract from the IpList function



For every function we execute a transaction hash is created.





The getIpList will return the Ip and the timestamp at the specified location and the getListLength returns the number of IP in the smart contract.

#### IV. PLAN FOR REVIEW 3

- Ganache CLI: In review 2 we have used Remix IDE to deploy the solidity file. In the review-3 we will be using Ganache CLI for the same.
- Truffle, Web3js: Truffle and Web3js creates the necessary files and directories required to run a web application.
- Rerouting the Packets, Blocking Malicious IPs: We have successfully implemented our attack and also the solidity file to mitigate it in review 2. In review 3, we will be combining these functionalities by making the code for rerouting the genuine packets and blocking malicious IPs on the basis of our solidity code.
- Web Interface using HTML, CSS: Finally, we will be adding an interface where we can easily see the blacklisted IPs as well as genuine IP requests.

### REVIEW – 3

#### Rerouting Packets:

SYN flooding is detected which means that we have detected DDoS attack flooding and have rerouted the packets successfully.

```
kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project/Review 2$ python3 div_reroute.py
Enter interface name
wlo1
enter incoming traffic port
80
SYN Flooding detected
```

#### Ganache-CLI:

Started ganache which is listening on 127.0.0.1:8545

```
kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project$ node_modules/.bin/ganache-cli
Ganache CLI v6.12.2 (ganache-core: 2.13.2)

Available Accounts
=====
(0) 0x93011069Fe03ccc0D94dcE4bB615F7238B0F96cb (100 ETH)
(1) 0xf71384270b630d31F09E21d40783F701cBf895E0 (100 ETH)
(2) 0xd877AE06c765c5d3F120C166C7969332fa676292 (100 ETH)
(3) 0xd996f58db3F6d18206982428037a6eC309Bae7Ef (100 ETH)
(4) 0x28a06e5fB6da9d491FBb8eAb1fF6c4f0E315380f (100 ETH)
(5) 0xEcd19b1977C9709F9F86854Bf458356e63261E78 (100 ETH)
(6) 0x249188eC132bd8c299495609708b16E994539c81 (100 ETH)
(7) 0x7CcAB82453290B80C7A9254db7361d4001AADCC2 (100 ETH)
(8) 0x53475eBE1CdbDC3D860D194B13445333CEe7254 (100 ETH)
(9) 0xF7221Fc6ad281CE21f4C0Ed94fE68d351A140d60 (100 ETH)

Private Keys
=====
(0) 0x092eb1cb3782460d4eec86ee67ae12d6970597e8a36d639883aa9d236736fe42
(1) 0xda688cc6e3d63c1914834a4567a003c91acf61d4b2b41a8dc51ed0c25b5e49b1
(2) 0x628097713cc0506affaa790b516778d51ca4a6c1fdcd9ec4dc43bdf5afb5067d
(3) 0x8c8a5a6be5c5fc53bbe67fad69a82124473fdb9107304be9a084036b3e7f41f
(4) 0xcada39b670f8a5ec883dba1c65cf40f4a543569585bc4b3d8f063a2b5781a105
(5) 0x87add0f32dc71f9213658c492acc9f20a13c975ac28398ad4e2ed4f3bf70b112
(6) 0x3fdb85b18ef07c8cfff9005c511da52130dc60c3cd22083253ca9f1535862e0
(7) 0x4a153bdc0fdc7ae41440d09b36f7df231b60ba383a62fb163f991a8f4d77a740
(8) 0x31bd4d07f119cf51d0545e41526e82d5669291c069717f554453ce99c9cd64d8
(9) 0x7272b3e9ade7ab732ec57306485d762162a15deba6b69a35d73300b43f755553

HD Wallet
=====
Mnemonic:      brother tube amused lonely tennis skin prize biology vacuum dentist interest reduce
Base HD Path:  m/44'/60'/0'/0/{account_index}

Gas Price
=====
20000000000

Gas Limit
=====
6721975

Call Gas Limit
=====
9007199254740991

Listening on 127.0.0.1:8545
```

Initialized web3 and checked whether it is interacting with ganache.

```
> web3.eth.getAccounts(console.log)
Promise {
  <pending>,
  domain:
    Domain {
      domain: null,
      _events:
        [Object: null prototype] {
          removeListener: [Function: updateExceptionCapture],
          newListener: [Function: updateExceptionCapture],
          error: [Function: debugDomainError] },
      _eventsCount: 3,
      _maxListeners: undefined,
      members: [],
      [Symbol(kWeak)]: WeakReference {} } }
> null [ '0x93011069Fe03ccc0D94dcE4bB615F7238B0F96cb',
  '0xf71384270b630d31F09E21d40783F701cBf895E0',
  '0xd877AE06c765c5d3F120C166C7969332fa676292',
  '0xd996f58db3F6d18206982428037a6eC309Bae7Ef',
  '0x28a06e5fB6da9d491FBb8eAb1fF6c4f0E315380f',
  '0xEcd19b1977C9709F9F86854Bf458356e63261E78',
  '0x249188eC132bd8c299495609708b16E994539c81',
  '0x7CcAB82453290B80C7A9254db7361d4001AADCC2',
  '0x53475eBE1CdbDC3D860D194B13445333CEe7254',
  '0xF7221Fc6ad281CE21f4C0Ed94fE68d351A140d60' ]
```



Deployed our smart contract with ganache and web3

```
> deployedContract.deploy({
...   data: bytecode,
... }).send({
...   from: '0x93011069Fe03ccc0D94dcE4bB615F7238B0F96cb',
...   gas: 1500000,
...   gasPrice: web3.utils.toWei('0.00003', 'ether')
... }).then((newContractInstance) => {
...   deployedContract.options.address = newContractInstance.options.address
...   console.log(newContractInstance.options.address)
... });
Promise {
  <pending>,
  domain:
    Domain {
      domain: null,
      _events:
        [Object: null prototype] {
          removeListener: [Function: updateExceptionCapture],
          newListener: [Function: updateExceptionCapture],
          error: [Function: debugDomainError] },
      _eventsCount: 3,
      _maxListeners: undefined,
      members: [],
      [Symbol(kWeak)]: WeakReference {} } }
> 0x40884F7e798e9f4935D05B6944Bc086374b80BB8
```

Transaction created on ganache after deploying our smart contract

```
eth_sendTransaction

Transaction: 0xdc4b54f4202e808484ae639ee3684b9faa134e408b3f23489b453e1bd135ddc2
Contract created: 0x40884f7e798e9f4935d05b6944bc086374b80bb8
Gas usage: 516350
Block Number: 1
Block Time: Sat Jun 05 2021 14:36:17 GMT+0530 (India Standard Time)

eth_getTransactionReceipt
eth_getCode
```

Web interface for our Local System module

DDoS DApp

File | /home/kushagra/Desktop/ISM%20Project/index.html

### Blacklist IPs

221.221.221.220

458562

SUBMIT

### List of Blacklisted IP Addresses

IP ADDRESS	TIMESTAMP
12.34.56.78	123365
56.95.183.65	982564
221.221.221.220	458562

Calling our smart contract functions from ganache command line.

```
> deployedContract.methods.getIpList(1).call(console.log)
Promise {
  <pending>,
  domain:
    Domain {
      domain: null,
      _events:
        [Object: null prototype] {
          removeListener: [Function: updateExceptionCapture],
          newListener: [Function: updateExceptionCapture],
          error: [Function: debugDomainError] },
      _eventsCount: 3,
      _maxListeners: undefined,
      members: [],
      [Symbol(kWeak)]: WeakReference {} } }
> null Result { '0': '0x35362e39352e3138332e3635', '1': '982564' }

> web3.utils.hexToAscii('0x35362e39352e3138332e3635')
'56.95.183.65'
> deployedContract.methods.getListLength().call(console.log)
Promise {
  <pending>,
  domain:
    Domain {
      domain: null,
      _events:
        [Object: null prototype] {
          removeListener: [Function: updateExceptionCapture],
          newListener: [Function: updateExceptionCapture],
          error: [Function: debugDomainError] },
      _eventsCount: 3,
      _maxListeners: undefined,
      members: [],
      [Symbol(kWeak)]: WeakReference {} } }
> null '3'
```

## Truffle:

truffle compile: Compiling our contracts in the contracts folder

truffle migrate --reset: Deploying our contract and retrieving the contract address

```
kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project/Truffle$ truffle compile

Compiling your contracts...
=====
> Everything is up to date, there is nothing to compile.

kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project/Truffle$ truffle migrate --reset

Compiling your contracts...
=====
> Everything is up to date, there is nothing to compile.

Starting migrations...
=====
> Network name:      'development'
> Network id:       1622882150299
> Block gas limit:  6721975 (0x6691b7)
```

```

1_initial_migration.js
=====

Deploying 'Migrations'
-----
> transaction hash: 0xd4282e285c85ffd9b63d4684cff59a0f9e4ecdaccaebe033b54b57d7c5c2b6fa
> Blocks: 0 Seconds: 0
> contract address: 0x8aaB02659725A872740DB8A7915e066C0B427302
> block number: 1
> block timestamp: 1622882314
> account: 0x28aDA727518fe972fF86820F7326b61184C07AF5
> balance: 99.9967165
> gas used: 164175 (0x2814f)
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.0032835 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.0032835 ETH

```

```

2_deploy_contracts.js
=====

Deploying 'Ip'
-----
> transaction hash: 0x91f1b94bfe7f17baaf35d0a4b4077705dbddedbc1bba7172757553c17a68817
> Blocks: 0 Seconds: 0
> contract address: 0xEad832148085b9672427b7F1503CF432743000B4
> block number: 3
> block timestamp: 1622882314
> account: 0x28aDA727518fe972fF86820F7326b61184C07AF5
> balance: 99.98601958
> gas used: 492505 (0x783d9)
> gas price: 20 gwei
> value sent: 0 ETH
> total cost: 0.0098501 ETH

> Saving migration to chain.
> Saving artifacts
-----
> Total cost: 0.0098501 ETH

Summary
=====
> Total deployments: 2
> Final cost: 0.0131336 ETH

```

Got the methods from our smart console through truffle console

```

methods:
{ ipArray: [Function: bound _createTxObject],
  '0x6b6d6c92': [Function: bound _createTxObject],
  'ipArray(uint256)': [Function: bound _createTxObject],
  ipList: [Function: bound _createTxObject],
  '0x9558ef58': [Function: bound _createTxObject],
  'ipList()': [Function: bound _createTxObject],
  updateIpList: [Function: bound _createTxObject],
  '0x6537e7db': [Function: bound _createTxObject],
  'updateIpList(string,uint256)': [Function: bound _createTxObject],
  getIpList: [Function: bound _createTxObject],
  '0x5776466a': [Function: bound _createTxObject],
  'getIpList(uint256)': [Function: bound _createTxObject],
  getListLength: [Function: bound _createTxObject],
  '0xb65c531b': [Function: bound _createTxObject],
  'getListLength()': [Function: bound _createTxObject] },
events: { allEvents: [Function: bound ] },
_address: '0xEad832148085b9672427b7F1503CF432743000B4',

```

Transactions created in ganache simultaneously

```
Transaction: 0xd4282e285c85ffd9b63d4684cff59a0f9e4ecdaccaebe033b54b57d7c5c2b6fa
Contract created: 0x8aab02659725a872740db8a7915e066c0b427302
Gas usage: 164175
Block Number: 1
Block Time: Sat Jun 05 2021 14:08:34 GMT+0530 (India Standard Time)

eth_getTransactionReceipt
eth_getCode
eth_getTransactionByHash
eth_getBlockByNumber
eth_getBalance
eth_getBlockByNumber
eth_getBlockByNumber
eth_gasPrice
eth_sendTransaction
```

Started our web client on localhost

```
kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project/Truffle$ cd app
kushagra@kushagra-Ubuntu-PC:~/Desktop/ISM Project/Truffle/app$ npm run dev

> app@1.0.0 dev /home/kushagra/Desktop/ISM Project/Truffle/app
> webpack-dev-server

i [wds]: Project is running at http://localhost:8080/
i [wds]: webpack output is served from /
i [wds]: Content not from webpack is served from /home/kushagra/Desktop/ISM Project/Truffle/app/dist
i [wds]: Hash: ee0d88ebccf4d736fb85
Version: webpack 4.41.2
Time: 497ms
Built at: 05/06/2021 2:20:03 pm
    Asset      Size  Chunks             Chunk Names
index.html  5.18 KiB          0  [emitted]
index.js    363 KiB       main  [emitted]  main
Entrypoint main = index.js
[0] multi (webpack)-dev-server/client?http://localhost:8080 ./src/index.js 40 bytes {main} [built]
[./node_modules/ansi-html/index.js] 4.16 KiB {main} [built]
[./node_modules/html-entities/index.js] 231 bytes {main} [built]
[./node_modules/loglevel/lib/loglevel.js] 8.36 KiB {main} [built]
[./node_modules/querystring-es3/index.js] 127 bytes {main} [built]
[./node_modules/webpack-dev-server/client/index.js?http://localhost:8080] (webpack)-dev-server/client?http://localhost:8080 4.29 KiB {main} [built]
[./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/client/overlay.js 3.51 KiB {main} [built]
[./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client/socket.js 1.53 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/createSocketUrl.js] (webpack)-dev-server/client/utils/createSocketUrl.js 2.89 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/log.js] (webpack)-dev-server/client/utils/log.js 964 bytes {main} [built]
[./node_modules/webpack-dev-server/client/utils/reloadApp.js] (webpack)-dev-server/client/utils/reloadApp.js 1.59 KiB {main} [built]
[./node_modules/webpack-dev-server/client/utils/sendMessage.js] (webpack)-dev-server/client/utils/sendMessage.js 402 bytes {main} [built]
[./node_modules/webpack-dev-server/node_modules/strip-ansi/index.js] (webpack)-dev-server/node_modules/strip-ansi/index.js 161 bytes {main} [built]
[./node_modules/webpack-hot sync ^\\.\\./log$] (webpack)/hot sync nonrecursive ^\\.\\./log$ 170 bytes {main} [built]
[./src/index.js] 1.59 KiB {main} [built]
+ 18 hidden modules
i [wds]: Compiled successfully.
```

DDoS DApp

localhost:8080/#

## Blacklist IPs

213.57.74.89

654321

Submit

## List of Blacklisted IP Addresses

IP ADDRESS	TIMESTAMP
78.21.65.41	123456
213.57.74.89	654321

Command line interface for our truffle module

```
truffle(development)> IpInstance.updateIpList("18.57.127.0",134652)
{ tx:
  '0x98261b627dd9fff385f9b6105ce815eaa9ec27b9dad3a185622d2ec17b653bec',
  receipt:
    { transactionHash:
      '0x98261b627dd9fff385f9b6105ce815eaa9ec27b9dad3a185622d2ec17b653bec',
      transactionIndex: 0,
      blockHash:
        '0x2db51fb3e4455975fd154610bc18ff5578f4d5e1631d7daa3c691f6910dc6201',
      blockNumber: 7,
      from: '0x28ada727518fe972ff86820f7326b61184c07af5',
      to: '0xead832148085b9672427b7f1503cf432743000b4',
      gasUsed: 85234,
      cumulativeGasUsed: 85234,
      contractAddress: null,
      logs: [],
      status: true,
```

```
    logs: [] }
truffle(development)> IpInstance.getListLength()
<BN: 3>
truffle(development)> IpInstance.getIpList(0)
Result { '0': '78.21.65.41', '1': <BN: 1e240> }
truffle(development)> IpInstance.getIpList(1)
Result { '0': '213.57.74.89', '1': <BN: 9fbf1> }
truffle(development)> IpInstance.getIpList(2)
Result { '0': '18.57.127.0', '1': <BN: 20dfc> }
```

## Automation:

Created an instance of our contract

```
> contractInstance = VotingContract.at(deployedContract.address)
Contract {
  _eth:
    Eth {
      _requestManager:
        RequestManager { provider: [HttpProvider], polls: {}, timeout: null },
      getBalance:
        { [Function: send] request: [Function: bound ], call: 'eth_getBalance' },
      getStorageAt:
        { [Function: send] request: [Function: bound ], call: 'eth_getStorageAt' },
      getCode:
        { [Function: send] request: [Function: bound ], call: 'eth_getCode' },
      getBlock:
        { [Function: send] request: [Function: bound ], call: [Function: blockCall] },
      getUncle:
        { [Function: send] request: [Function: bound ], call: [Function: uncleCall] },
      getCompilers:
        { [Function: send] request: [Function: bound ], call: 'eth_getCompilers' },
```

Got the contract address to be used using the command shown

```
> module.exports = deployedContract.address
'0x8c95ddb6befbfff8e5eb17dec7ab0e67d894b5a70'
```

Details of our smart contract and the functions in it.

```
address: '0x8c95ddb6befbfff8e5eb17dec7ab0e67d894b5a70',
abi:
  [ { constant: true,
    inputs: [Array],
    name: 'getIpList',
    outputs: [Array],
    payable: false,
    stateMutability: 'view',
    type: 'function' },
    { constant: false,
    inputs: [Array],
    name: 'updateIpList',
    outputs: [],
    payable: false,
    stateMutability: 'nonpayable',
    type: 'function' },
    { constant: true,
    inputs: [Array],
    name: 'ipArray',
    outputs: [Array],
    payable: false,
    stateMutability: 'view',
    type: 'function' },
    { constant: true,
    inputs: [],
    name: 'ipList',
    outputs: [Array],
    payable: false,
    stateMutability: 'view',
    type: 'function' },
    { constant: true,
    inputs: [],
    name: 'getListLength',
    outputs: [Array],
    payable: false,
    stateMutability: 'view',
    type: 'function' } ],
```

Adding the blacklisted IPs to blockchain automatically through our index.js file

```
kushagra@kushagra-Ubuntu-PC:~/Automation/Project$ node index.js
Adding BlackListed Ip to blockchain :
41.24.111.67

Adding BlackListed Ip to blockchain :
53.57.198.219

Adding BlackListed Ip to blockchain :
63.106.243.81

3
ip:41.24.111.67
time:1622887304455
ip time:1622887300391
Needs to be blocked

Writing to file

ip:53.57.198.219
time:1622887305300
ip time:1622887300391
Needs to be blocked

Writing to file

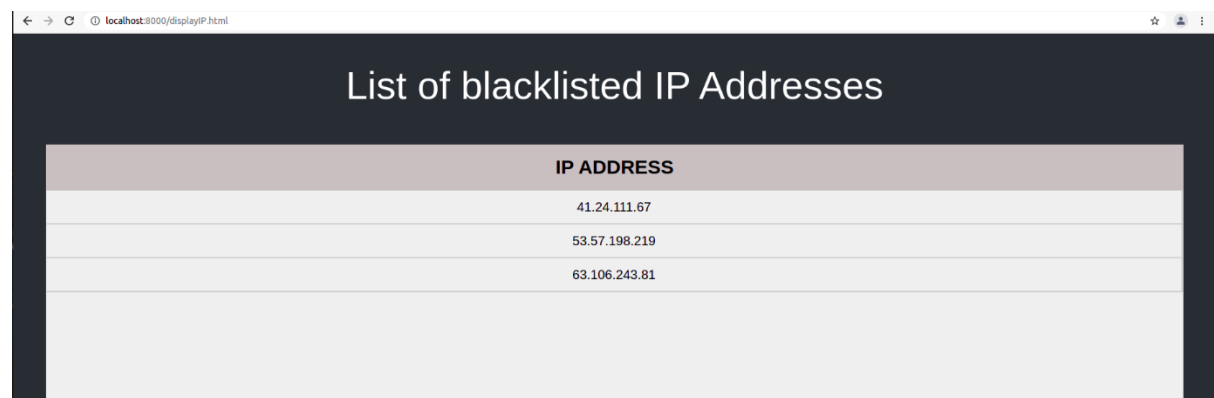
ip:63.106.243.81
time:1622887306224
ip time:1622887300391
Needs to be blocked

Writing to file
```

## Blocking the IPs

```
kushagra@kushagra-Ubuntu-PC:~/Automation/Project$ python3 blockip.py
[sudo] password for kushagra:
41.24.111.67
  is blocked
53.57.198.219
  is blocked
63.106.243.81
  is blocked
```

## Viewing which IPs have been blacklisted on a web interface



The screenshot shows a web browser window with the address bar displaying 'localhost:8000/displayIP.html'. The main content area has a dark background with the title 'List of blacklisted IP Addresses' in white. Below the title is a table with a single column header 'IP ADDRESS' and three rows of IP addresses: '41.24.111.67', '53.57.198.219', and '63.106.243.81'.

IP ADDRESS
41.24.111.67
53.57.198.219
63.106.243.81

## Transactions created on ganache in the background

```
Transaction: 0x07802a017c1e1432e82a42d533e1f9b7e92156366d7c128feb1e21b8942ea189
Gas usage: 130210
Block Number: 8
Block Time: Sat Jun 05 2021 15:31:41 GMT+0530 (India Standard Time)

eth_accounts
eth_sendTransaction

Transaction: 0xa21373af0b5d3423735b93847a21de89c418f083b7032e9aa0674438df3f8961
Gas usage: 86079
Block Number: 9
Block Time: Sat Jun 05 2021 15:31:42 GMT+0530 (India Standard Time)

eth_accounts
eth_sendTransaction

Transaction: 0x1dfaa5b880f3d02b466e161db065017f971baa65ff17538aa971292207abb437
Gas usage: 86079
Block Number: 10
Block Time: Sat Jun 05 2021 15:31:43 GMT+0530 (India Standard Time)
```

## Additional module as discussed in Review 3:

We have added a new parameter in the updateIpList function which will take noOfBytes as an input and then update in our blockchain ledger. This parameter will help and slow DDoS attack as well as for IP which are constantly flooding our network with traffic.

## Updated Solidity Code

```
pragma solidity ^0.4.21;

contract Ip {

    mapping (string => uint256) private timestamp;
    string public ipList;
    string[] public ipArray;
    uint256 count = 0;
    string public ByteList;
    string[] public ByteArray;

    function updateIpList(string ipAddress,uint256 currenttime,string noOfBytes) public {
        ipList = ipAddress;
        ByteList = noOfBytes;
        if(timestamp[ipAddress]==0)
        {
            count++;
            ipArray.push(ipAddress);
            ByteArray.push(noOfBytes);
        }
        timestamp[ipList] = currenttime;
    }

    function getIpList(uint256 num) view public returns (string,uint256,string) {
        return (ipArray[num],timestamp[ipArray[num]],ByteArray[num]);
    }

    function getListLength() view public returns (uint256) {
        return (count);
    }
}
```

## Transaction Created for the deployed Contract

```
✓ [vm] from: 0x583...eddC4 to: Ip.(constructor) value: 0 wei data: 0x608...a0029 logs: 0 hash: 0x69b...05f68
```

## Deployed Contract Functions

Deployed Contracts

▼ IP AT 0X7EF...8CB47 (MEMORY)

updateIpList string ipAddress, uint256 cur ▼

ByteArray uint256 ▼

ByteList

getIpList uint256 num ▼

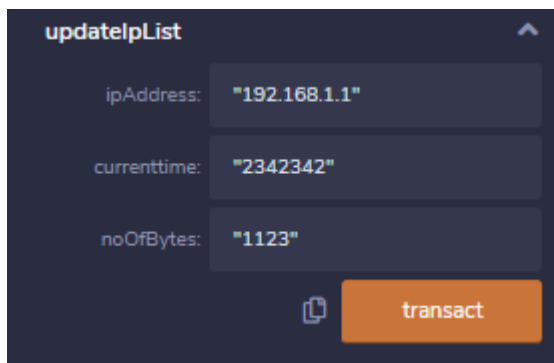
getListLength

ipArray uint256 ▼

ipList

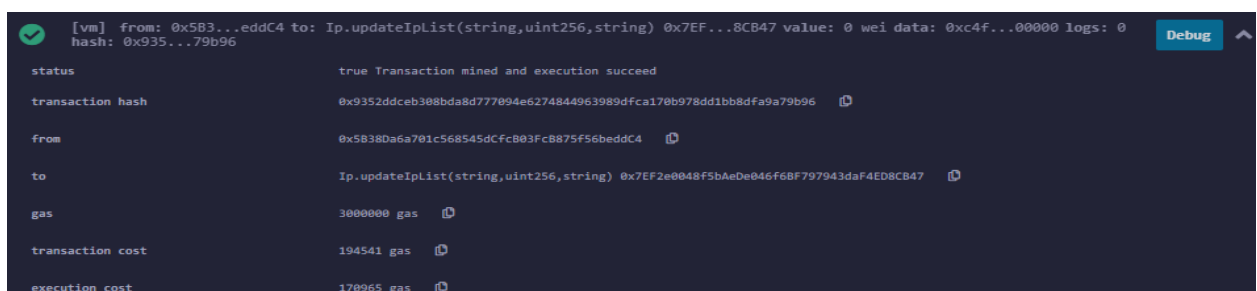


## updateIpList Function



The screenshot shows a function call for `updateIpList`. It has three input fields: `ipAddress` with the value `"192.168.1.1"`, `currenttime` with the value `"2342342"`, and `noOfBytes` with the value `"1123"`. Below these fields is a `transact` button.

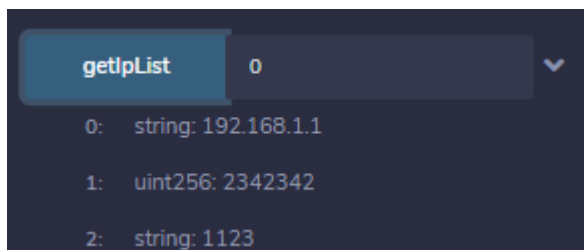
## Transaction for updateIpList Function



The screenshot displays a transaction log for the `updateIpList` function. It includes a status bar at the top with a green checkmark and a `Debug` button. The main content is a table with the following data:

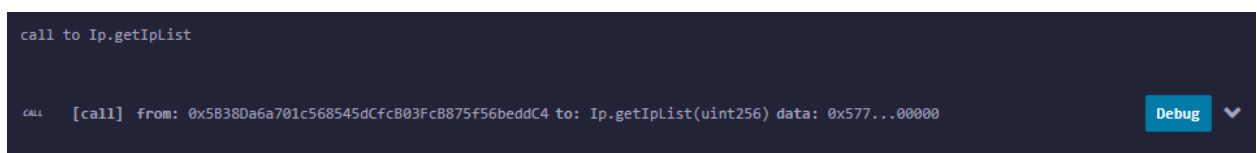
status	true Transaction mined and execution succeed
transaction hash	0x9352ddceb38bda8d777094e6274844963989dfca170b978dd1bb8dfa9a79b96
from	0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to	Ip.updateIpList(string,uint256,string) 0x7EF2e0048f5bAeDe046f68F797943daF4ED8CB47
gas	3000000 gas
transaction cost	194541 gas
execution cost	170965 gas

## GetIpList Function



The screenshot shows a function call for `getIpList`. It has a single input field labeled `0` with the value `0`. Below the field, the arguments are listed: `0: string: 192.168.1.1`, `1: uint256: 2342342`, and `2: string: 1123`.

## Transaction for GetIpList



The screenshot displays a transaction log for the `GetIpList` function. It includes a status bar at the top with a green checkmark and a `Debug` button. The main content is a table with the following data:

call to Ip.getIpList	
CALL	[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Ip.getIpList(uint256) data: 0x577...00000

## V. PERFORMANCE ANALYSIS/ DISCUSSION OF RESULTS

After performing a DDoS attack and rerouting the packets, we retrieved some malicious IPs and successfully added them to our blockchain through several methods, mainly using ganache and truffle.

```
kushagra@kushagra-Ubuntu-PC:~/Automation/Project$ node index.js
Adding BlackListed Ip to blockchain :
41.24.111.67

Adding BlackListed Ip to blockchain :
53.57.198.219

Adding BlackListed Ip to blockchain :
63.106.243.81

3
ip:41.24.111.67
time:1622887304455
ip time:1622887300391
Needs to be blocked

Writing to file

ip:53.57.198.219
time:1622887305300
ip time:1622887300391
Needs to be blocked

Writing to file

ip:63.106.243.81
time:1622887306224
ip time:1622887300391
Needs to be blocked

Writing to file
```

After adding the malicious IPs to the blockchain we were able to actually block them.

```
kushagra@kushagra-Ubuntu-PC:~/Automation/Project$ python3 blockip.py
[sudo] password for kushagra:
41.24.111.67
is blocked
53.57.198.219
is blocked
63.106.243.81
is blocked
```

Advantages of using Blockchain for mitigating DDoS:

- Transparency - Each and every machine in the network have an access to all the blacklisted IP addresses.
- Cost Effective - Our method is much cost effective in comparison to the solutions provided by other companies as it does not require any third party to provide DDoS Protection.

- Immutability - No attacker can try to change the IP addresses once they are stored in the blockchain as it's a distributed ledger.
- Time-efficient – Unlike Machine Learning solutions, the proposed method is not as time consuming in detecting and blocking the malicious IPs on all servers of the organization.

## **VI. CONCLUSION**

We have developed a model that will help companies to drop malicious and blacklisted IP based on the timestamp and the no of bytes of data. We have developed the system on the local system and on the localhost server. We have also implemented an automation system that will store the details of the network in the file and then use after re-routing we will use this file to take the IP and then block the IP and update in our Blockchain ledger. The web client that is linked to every module can be used to add IP in our blockchain ledger and then block the IP. The result after updating the blockchain ledger were displayed on the webpage for our every module.

**Google Drive Link: [Click Here](#)**

## VII. REFERENCES

1. Chaoxia Qin, Bing Guo, Yan Shen, Tao Li, Yun Zhang, Zhen Zhang, *A Secure and Effective Construction Scheme for Blockchain Networks*, Security and Communication Networks Volume, 2020
2. Alex R. Mathew, *Cyber Security through Blockchain Technology*, International Journal of Engineering and Advanced Technology (IJEAT), 2019
3. Mostafa Yavari, Masoumeh Safkhani, Saru Kumari, Sachin Kumar, Chien-Ming Chen, *An Improved Blockchain-Based Authentication Protocol for IoT Network Management*, Security and Communication Networks Volume, 2020
4. Arnab Bose, Gagangeet Singh Aujla, Maninderpal Singh, Neeraj Kumar, Haotong Cao, *Blockchain as a Service for Software Defined Networks: A Denial of Service Attack Perspective*, IEEE Intl Conf on Dependable, Jai Hind, Jai Bharat Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress, 2019
5. Kyoungmin Kim, Youngin You, Mookyu Park, Kyungho Lee, *DDoS Mitigation: Decentralized CDN Using Private Blockchain*, 10<sup>th</sup> International Conference on Ubiquitous and Future Networks (ICUFN), 2018
6. Wani, Sharyar; Imthiyas, Mohammed; Almohamedh, Hamad; Alhamed, Khalid M; Almotairi, Sultan; Gulzar, Yonis, *Distributed Denial of Service (DDoS) Mitigation Using Blockchain—A Comprehensive Insight*, Symmetry 13, no. 2: 227, 2021
7. Chakraborty, Sushmita & Kumar, Praveen & Sinha, Bhawna, *A Study on DDoS Attacks, Danger and its Prevention*, 10.1729/Journal.20847, 2019
8. Thilagavathi, M. and DrA. Saradha. *Impact Analysis of Dos & DDos Attacks*, IOSR Journal of Computer Engineering 16: 24-33, 2014
9. Mukhopadhyay, Debajyoti & Oh, Byung-Jun & Shim, Sang-Heon & Kim, Young-Chon, *A Study on Recent Approaches in Handling DDoS Attacks*, 2016
10. Balobaid, Awatef & Alawad, Wedad & Aljasim, Hanan, *A study on the impacts of DoS and DDoS attacks on cloud and mitigation techniques*, 416-421. 10.1109/CAST.2016.7915005, 2016

11. Omar Abdulkader, Alwi M. Bamhdi, Vijey Thayananthan, Fathy Elbouraey, Bandar Al-Ghamdi. *A Lightweight Blockchain based Cybersecurity for IoT environments*, 2019 6th IEEE International Conference on Cyber Security and Cloud Computing
12. V. V. S., & Maheswari, B. U., *Blockchain Based DDoS Mitigation Using Machine Learning Techniques*, 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)
13. Giri, N., Jaisinghani, R., Kriplani, R., Ramrakhyani, T., & Bhatia, V., *Distributed Denial Of Service(DDoS) Mitigation in Software Defined Network using Blockchain*, 2019 Third International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)
14. Bansal, P., Panchal, R., Bassi, S., & Kumar A., *Blockchain for Cybersecurity: A Comprehensive Survey*, 2020 IEEE 9th International Conference on Communication Systems and Network Technologies
15. Taylor, P. J., Dargahi, T., Dehghantanha, A., Parizi, R. M., & Choo, K.-K. R, *A systematic literature review of blockchain cyber security*, Digital Communications and Networks