Evaluating a Blockchain-based Cooperative Defense

Bruno Rodrigues, Lukas Eisenring, Eder Scheid, Thomas Bocek, Burkhard Stiller
Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH
E-mail: rodrigues,scheid,bocek,stiller@ifi.uzh.ch
lukas.eisenring@uzh.ch

***Abstract*—The volume of traffic generated by modern Distributed Denial-of-Service (DDoS) attacks suggests that centralized defenses are not the most effective approach to counter these attacks. An alternative to reduce the burden of detection and mitigation is to combine centralized defense systems, creating a global and cooperative protection system. However, existing approaches suffer from the complexity of deployment and operation across different systems. Blockchains appear in this scenario as an alternative to simplify the exchange of information in a cooperative defense. This work evaluates in both local and global experimentations the performance of the blockchain system proposed in [8] concerning the latency to perform the signaling of blacklisted addresses.**

***Index Terms*—Blockchain, Security, Management, Cooperative Defense, Distributed Denial-of-Service (DDoS),**

I. Introduction

Distributed Denial-of-Service (DDoS) attacks are a large-scale, coordinated attempt to make a target system's resources unavailable. Although being a widely known attack, it remains as one of the major causes of concern for service providers. The increasing number of connected devices (stationary and portable) allow attackers to control a vast amount of unsecured devices that range from cameras to smart fridges to generate malicious attacks. As result, traditional in-house (*e.g.,* firewall, deep packet inspection) or cloud-based (*e.g.,* services of CloudFlare or Akamai) defenses can become a communication bottleneck due to the need to download and process (*i.e.,* detect and mitigate) all traffic measurements at a single location. As a consequence, if an attack presents a massive volume of traffic or is highly sophisticated, legitimate users may be impaired until the attack stops.

As an alternative, the distributed nature of DDoS attacks suggests that a distributed and coordinated defense is necessary for a successful defense. Advantages of cooperative defenses have been widely recognized in the literature [7], [15]. For instance, it allows to combine the detection/mitigation capabilities of the cooperative entities; reduce the detection/mitigation overhead in a single entity, and to block malicious traffic near its source. However, still does not exist a widespread deployment of such systems because of their lack of effectiveness and implementation complexities. Among the challenges of existing approaches are the high complexity of operation and coordination, the need of trusted communication, and lack of incentives for the service providers to cooperate.

978-3-903176-15-7

Blockchains provides a trustworthy, decentralized, and publicly available data storage making it an interesting opportunity for organizations to increase business agility and reduce costs by removing intermediaries in distributed applications. For example, in the context of a DDoS cooperative defense, blockchain capabilities could be leveraged for signaling attacks as a mitigation requests across a blockchain network, and serve as an immutable platform for the exchange of mitigation services defined in smart contracts of different peers.

Previous work presented a system architecture to leverage blockchain capabilities towards a cooperative network defense [8], [9]. However, the evaluation was limited to local experiments in an environment under controlled conditions. This paper extend the performance assessment to a global sphere analyze and discuss the different aspects under a larger scale. Thus, eight instances at different geolocations were deployed to simulate a global scale DDoS attack and its cooperative signaling using the Blockchain Signaling System (BloSS).

The remainder of this work is organized as follows. Section II overviews the system. Section III presents the performance evaluations. Related work is presented in Section IV and, finally, Section V concludes the work.

II. Overview

The Blockchain Signaling System (BloSS) [8], [9] is a decentralized DDoS defense system where each AS (Autonomous System) taking part in the cooperative defense alliance and running the BloSS can post information about an ongoing attack to the Ethereum blockchain [13]. Attack information posted to the blockchain is not directly stored on the blockchain due to limited block sizes. For this purpose, IPFS (InterPlanetary File System) [1] is used as a decentralized and highly scalable storage solution to hold attack information. Each AS running the BloSS also maintains an IPFS node to enable the decentralized storage. Whenever a new set of attack information is posted to the blockchain, the data is first stored in IPFS, and only the hash as a unique identifier of the storage location within IPFS is stored in a block on the Ethereum blockchain.

Figure 1 shows a prototypical defense scenario involving a mitigator (BloSS on Mitigator AS) as well as target AS (BloSS on target AS). Attack detection is outside the scope of the BloSS, so the first step includes compiling the attack information and encrypting it to post the IPFS hash to the Ethereum blockchain later. The encryption of attack information posted to IPFS ensures the confidentiality as well as the

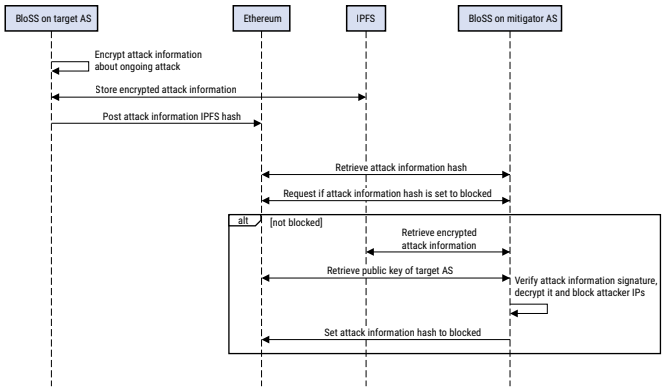


Fig. 1. Overview of BloSS

integrity of the attack information based on a per-message signature bundled with the attack information. Privacy is an essential attribute of the data exchange between ASes since the attack information can be sensitive in regards to implicating individuals both as victims of an ongoing DDoS attack or as the perpetrators of the attack. Verifying the integrity of the attack information allows to hold each AS accountable for the information posted to the blockchain and makes a forgery of attack information impossible. The integrity-check is enabled through a public key published by each AS to the blockchain and therefore available to all ASes participating in the BloSS defense alliance. Without this measure in place, forgery of attack information could allow a malicious party to indicate specific IP addresses as being the source of an ongoing attack, and in effect, blocking flows from these addresses to the target address specified in the attack information. Furthermore, to minimize access to IPFS as well as Ethereum to access attack information and the public key of the target AS the attack information hash is connected to a boolean indicating whether the mitigator has already obtained the information AS to block the attackers.

III. Experiments

Local and global experiments were conducted to evaluate the signaling delay of BloSS. Node configuration for both experiments are described in Section III-A. First, local experiments were performed at the University of Zürich (UZH) network to tune BloSS configurations and estimate its performance in a controlled environment. These experiments are described in Section III-B. Second, global experiments were performed to assess the performance of BloSS in a geographically distributed and heterogeneous environment. Section III-C describes these experiments.

A. Configuration

Instances used to perform the BloSS measurements are described in Table I. Local instances were configured with different Virtual CPU (vCPU) and RAM capacities ranging from 1 to 12 vCPUs and 1 GB to 32 GB of RAM. Instances marked with an asterisk processed parallel workload from other services, such as Graphical User Interfaces (GUI).

Local instances (UZH IfI and UZH Irchel) were configured with more processing power (vCPUs) and RAM than other nodes due to the availability of resources in the UZH infrastructure. However, nodes used in the global experiment, such as AWS, Azure, and DigitalOcean, were configured with similar specifications (1 GB of RAM for Amazon t2.micro, Microsoft Azure B1S, and DigitalOcean small instances). However, as observed during the experiments, due to low RAM in experimental runs instances were adjusted to 2 GB of RAM. Furthermore, while AWS and Azure were configured with 1 vCPU, DigitalOcean was configured with 2 vCPUs and thus, being able to handle more processes simultaneously. Also, it is important to note that real traffic was not generated in the experiments. Thus, different files with random IPv4 addresses were produced, and requests were submitted to the blockchain simulating the signaling of an attack. Figure 3 presents the distribution of the time required for this process during the experiments.

B. Local Experiments

Local experiments measured the performance of BloSS to exchange black-listed addresses with different file sizes. Six instances were deployed inside the UZH network: two at the Institute of Informatics (IfI) and the remainder at the UZH Irchel campus. The distance between these instances is 3 km, and the available bandwidth was 200 Mbit/s.

Figure 2 depicts the elapsed time for a transaction inside the UZH in four experiments. For each test, 100 measurements (blue dots) were performed, and the file size varied from 10 kB to 10 MB. The red line represents the median value of the corresponding experiment set. Also, results do not account for the constant 15 seconds necessary to create a block in the Ethereum Rinkeby blockchain. The average to perform the

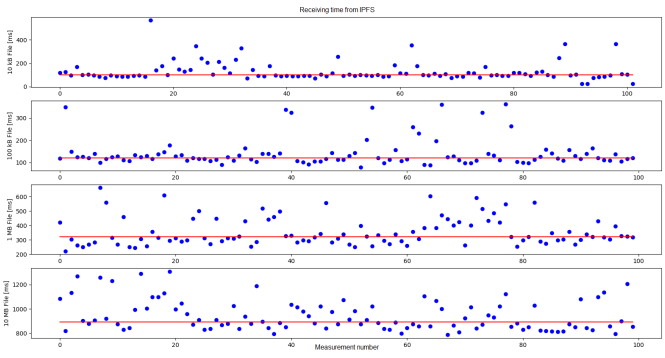


Fig. 2. Elapsed time for a transaction within the UZH network with different file sizes. From the bottom to the top: 10 MB, 1 MB, 100 kB, 10 kB.

signaling of a ten kB file was around 100 ms (first chart on top), and for a 100 kB file is 120 ms (second chart on top). For a 1 MB file, the delay was approximately 320 ms. Thus, a baseline time of 75 ms (processing of addresses) and an additional 25 ms per 100 kB could be observed to complete the signaling of an attack.

Also, it is important to note that the maximum block-size defined in IPFS is 1 MB. Thus, in the case of large-

TABLE I
CONFIGURATION OF THE INSTANCES USED IN THE EXPERIMENTS

Provider	Local	Local	Contabo	Amazon EC2	Amazon EC2	Microsoft Azure	Microsoft Azure	DigitalOcean	DigitalOcean
Type	UZH IfI*	UZH Irchel*	VPS M*	t2.micro	t2.small	B1S	B1MS	small	medium
vCPU	4	12	2	1	1	1	1	1	2
RAM	4 GB	32 GB	6 GB	1 GB	2 GB	1 GB	2 GB	1 GB	2 GB

TABLE II
ROUND TRIP TIME BETWEEN DEPLOYED INSTANCES

From \ To	Australia	Brazil	Germany	USA	Switzerland
Australia	-	344 ms	315 ms	248 ms	351 ms
Brazil (AWS)	337 ms	-	229 ms	130 ms	220 ms
Brazil (Azure)	366 ms	6 ms	224 ms	116 ms	204 ms
Germany	333 ms	227 ms	-	103 ms	21 ms
Asia	197 ms	326 ms	279 ms	227 ms	260 ms
Japan	153 ms	266 ms	334 ms	171 ms	170 ms
USA	247 ms	128 ms	104 ms	-	96 ms
Switzerland	347 ms	221 ms	20 ms	94 ms	-

scale attacks where a potential list of blacklisted addresses is above 1 MB a file is segmented into different blocks, which increases the signaling time. During local experiments, a file with 10 MB was created to evaluate the impact of such segmentation. Using a traditional file transfer method (*wget*), it was observed an average delay of 3,200 ms (10 times 320 ms) to complete. However, IPFS took approximately 900 ms due to the decentralization throughout multiple instances using IPFS. An additional experiment included ten files of 1 MB size simultaneously to IPFS, submitting the hash (generated when the blacklist is added to IPFS) value to the same Smart Contract, and retrieving this file on the mitigator instance. The measured time was 950 ms on average, which is slightly above the transmission time of a regular 10 MB file (900 ms).

C. Global Experiments

To evaluate BloSS in a geographically distributed environment, 8 instances in different countries were deployed to measure the data transfer time among themselves. Table III details the specification of such instances.

Two instances were located in Europe, Switzerland, and Germany. In Asia, two other instances were deployed in Singapore and Japan. Three separate instances were situated in America, one in the USA, close to New York, and two instances were located in Brasil near São Paulo. At the time of the experiment, these main cloud providers do not provide instances in Africa. Finally, instances were hosted by different cloud providers (AWS, Azure, DigitalOcean, and Contabo) and configured with similar compute and network capacities.

Bandwidth was measured using the package `speedtest-cli` from the Ubuntu repository and measurements performed based on the instance with the lowest latency from the list of `speedtest.net`. Values represent the approximated minimum values upload and download times in a batch of 10 measurements. The lowest bandwidth measured is from Contabo with 90 Mbit/s, which restricts available bandwidth to a maximum of 100 Mbit/s. All other instances were configured with at least a bandwidth of 170 Mbit/s. Differences in available bandwidth were not significant for lists of addresses of under 200 kB in the experiments.

The Round Trip Time (RTT) provides an estimated lower bound for the time needed to transfer a file between two instances. The measurements presented in Table II represents an average value of 100 requests at the interval of 10 seconds.

It is worth mentioning that Microsoft Azure does not allow incoming ICMP traffic. Therefore, ping requests could not be answered by instances deployed in this cloud provider. The measurements shows approximately symmetric results, *i.e.,* the RTT time from Brazil to Switzerland is approximately the same as from Switzerland to Brazil.

1) Signaling Latency: The latency between a request of mitigation service (signaling) and the mitigation is an important metric in a cooperative defense system. Experiments consisted in executing the sequence of steps in Figure 1 to evaluate the delay from the beginning of an attack until the attack has been mitigated as well as the CPU usage of each BloSS instance throughout the entire mitigation.

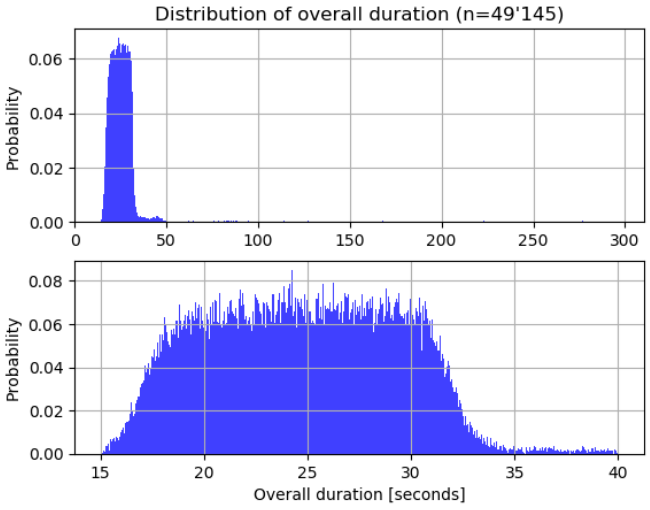


Fig. 3. Probability distribution of overall elapsed time in the upper histogram and range between 15 and 40 seconds zoomed the bottom histogram.

More than 95% of the measured values are between the 15 to 35 seconds range. The lowest signaling time measured was 14.1 seconds, while the highest was 3 minutes. Considering the measured values and that a block is created at every 15 seconds in the Ethereum Rinkeby blockchain, most transactions were

TABLE III
SPECIFICATIONS OF INSTANCES USED FOR THE GEOGRAPHICALLY DISTRIBUTED EVALUATION

Location	Australia	Brazil	Brazil	Germany	Singapore	Japan	USA	Switzerland
Instance Type	AWS t2.small	Azure B1MS	AWS t2.small	Contabo VPS M	Azure B1MS	Azure B1MS	DigitalOcean Medium	Local
Bandwidth	350 Mbit/s	225 Mbit/s	180 Mbit/s	90 Mbit/s	400 Mbit/s	180 Mbit/s	170 Mbit/s	310 Mbit/s

mined in the first or second block after a transaction was submitted.

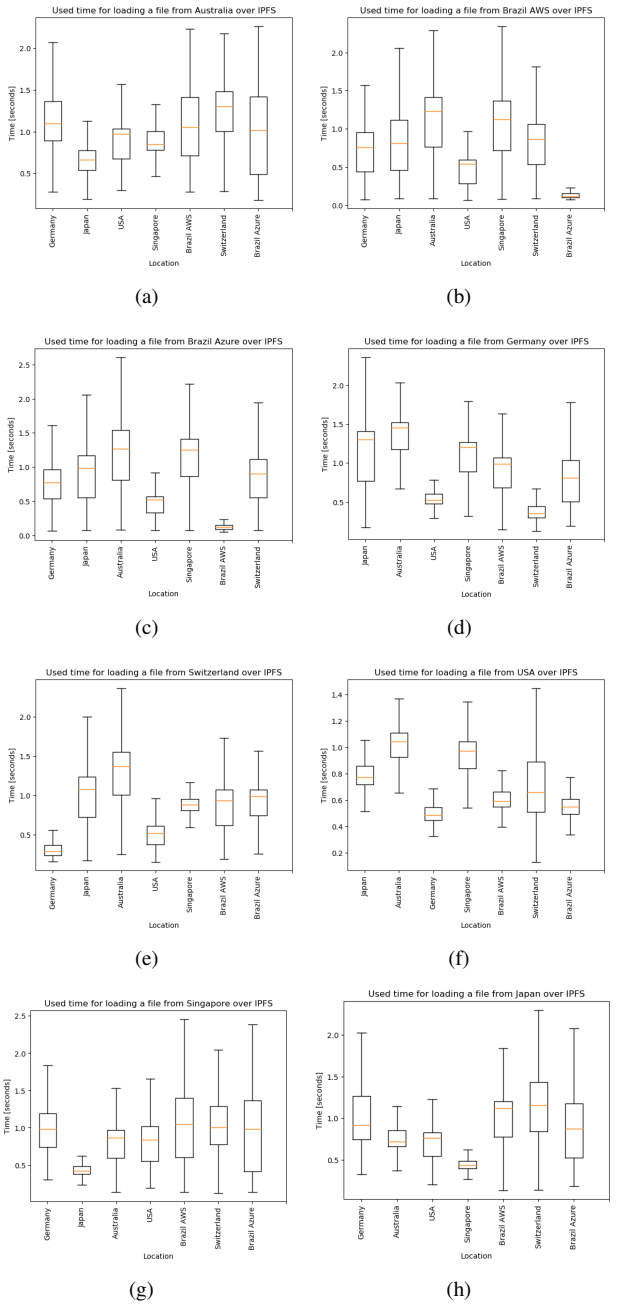


Fig. 4. (a) Australia; (b) Brazil AWS; (c) Brazil Azure; (d) Germany; (e) Switzerland; (f) USA; (g) Singapore; (h) Japan

Figure 4 depicts the results of the measurements of the time required for successfully transferring a file from a sender to receivers. Every file in the experiment contains 10,000 randomly generated blacklisted IPv4-addresses, resulting in a file size of 173 kB. Files were published by BloSS-enabled instances, with an attacked network assigned and other BloSS instances retrieving the file to support the mitigation of the attack. In general, proportions between transfer times are distributed as shown in RTT measurements Table II. Transmission times of near-located instances is, in general, lower than between farther distanced instances, but not symmetric in all cases. Differences in observed transfer times mostly relates to the geographical proximity of instances and the availability of content hosted in the IPFS nodes. For example, during the experiments some instances closely located (*e.g.*, Switzerland-Germany, Brazil-Brazil) resulted - as expected - in low signaling delays in contrast to others. Furthermore, some instances could benefit from a direct link connection between the same cloud provider (*e.g.*, Azure instances Brazil-Japan).

It should be noted that measurements representing from system failures were not considered. Thus, outliers ranging from more than 1.5 interquartile than the second and third quartile were discarded. Furthermore, a significant variation in data transference times (delay between instances) was observed. The maximum delay observed was 2.6 seconds while the average values by instance regardless varied between 0.9 and 1.2 seconds. The fastest transmission between instances was in Brazil where, due to their proximity, the transference time was below 200 ms. In general, to retrieve signaled addresses represents the overall time an instance uses to receive a block from the blockchain, retrieve the file from the DHT (Distributed Hash Table) and its own operations to decrypt the file.

2) Large Datasets: In large attack occasions, such as the DDoS attack on the French provider OVH in 2016 where over 150'000 IoT devices were involved [6], larger files have to be handled. Figure 5 (a) shows the measured values of four instances in comparison of 150'000 to 10'000 IP addresses as in the previously mentioned tests.

Figure 5 (a) presents mean values for the larger file, these values are between four and ten times higher for transmitting data 15 times larger. On one hand, files of over 1 MB size delivers the advantages of the decentralized approach of IPFS by splitting these files into multiple hosts. For example, for a blacklist published in Australia, the instance in Switzerland can retrieve the first block from the USA, which already has loaded the block, some blocks from Australia, and other blocks from Germany. This behavior allows to minimize the usage of distant instances and interacts with close-by instances more often.

On the other hand, the spread of measurements is higher.

In the best case, blocks are available from a close-by instance and the transfer time will be relatively low. All instances can try to receive the data at the same time from one instance which gets congested at this moment and can serve only two requests simultaneously. In this case, some instances have to wait until the resources of the sender are available. As a result, the used time increases strongly.

3) HTTP Comparison: To compare the results of IPFS as the data channel to exchange blacklisted addresses measurements using the data transmission over HTTP were performed. Figure 5 (b) depicts the time taken to load the data using HTTP and IPFS.

Measurements were performed using the instance in Germany as file provider, in which instances requested the target file 120 times. In general, the transmission over HTTP was slower than over IPFS. The difference between the time taken over HTTP and IPFS is increased by the distance between the two instances. The difference in the spread of the two instances in Brazil is noticeable probably caused by the different overseas cable of the providers.

4) CPU Overhead: In order to better understand the variations in CPU usages between the two scenarios (with and without encryption), minimum, maximum, average and median CPU usage across all instances have been compiled. Each entry represents the average over 10 runs for the specific statistical indicator. Table IV shows statistics with data encryption turned on turned off.

TABLE IV
CPU OVERHEAD MEASUREMENTS

CPU Usage	Encrypted	Not Encrypted
Min (%)	5.8%	4.8%
Max (%)	21.2%	18.2%
Average (%)	13.2%	10.5%
Median (%)	13.4%	10.3%

By collecting CPU usage information, it can be shown that the attack itself does not contribute to a considerable increase in CPU usage, except for short bursts to post attack reports and retrieve them from the blockchain. The baseline CPU usage of around 15% has to be attributed toward the Python programming language, which is an interpreted language, therefore creating a small overhead when running programs written in that language. Apart from this, the BloSS requires to periodically analyze traffic information as well as request attack reports from the blockchain, which both contribute toward the baseline CPU usage. Since 15% of average CPU usage is not a negligible amount of processing resources consumed, this value could be improved by reducing the periodic frequency tasks. However, this would result in increased delays to mitigate ongoing attacks.

D. Discussion

While local experimentation helped to create the basis of the system and reveal first aspects of configuration, the global evaluation exposed the system to real-world problems. During the experiments, it can be seen that the off-chain tasks involving file management, have a more significant influence on transfer latency than the steps in the blockchain. This influence varies according to the file size and distance between IPFS gateways, being able to observe a transfer time increased to 20 seconds in the global experiments with large datasets (150'000 IPv4 addresses) in contrast to the payload of 10,000 IPv4 addresses.

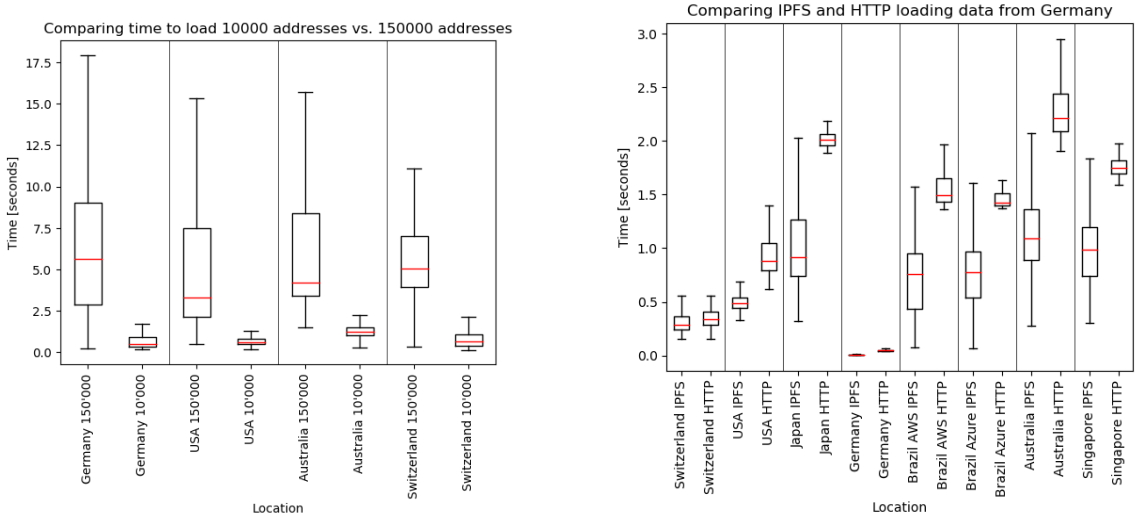
Also, during the local experiments, it was possible to transfer files up to 4 GB size and ten files simultaneously without failures. However, in the global test with instances running on cloud servers from various providers some issues were found. Firstly, it is necessary to observe that BloSS uses IPFS and the Ethereum blockchain for signaling attacks. Thus, it is required an available communication channel to send/answer requests during an attack. Secondly, the use of relatively new technologies such as IPFS and Blockchain may result in failures during its operation. For example, IPFS experienced problems of excessive memory consumption in instances running on Amazon EC2 t2.small, Microsoft Azure B1S and DigitalOcean small instances with 1 GB of RAM, as shown in Table I, which needed to be upgraded to Amazon EC2 t2.medium, Microsoft Azure B1MS and DigitalOcean medium instances providing 2 GB of RAM.

The distribution of the used memory (besides the portion allocated to the operating system) is spread in the following proportion: two parts for the IPFS daemon and one part for the Ethereum client with a deviation of up to 15% of this ratio. During block indexing periods, the Ethereum client allocated more memory for this process. The developers of IPFS acknowledge the memory usage problem on Github [11]. At the moment, a more stable system can be built starting the IPFS daemon as a service. However, this restarts the daemon by itself after killing the process, resulting in an IPFS outage of about 10 seconds.

IV. RELATED WORK

The field of DDoS defense system contains an increasing number of approaches to solve the multi-domain DDoS mitigation problem. The BloSS is best comparable with the decentralized systems, which mainly differ in the communication mechanism they employ. DOTS [4] proposes to standardize intra-organization and inter-organization communications. To support the exchange of data within constrained instances, DOTS use the CoAP (Constrained Application Protocol) [12]. CoAP is a specialized web transfer protocol enabling constrained nodes to exchange data, combining high interoperability and low communication overhead. However, DOTS presents a complex architectural design, which hinders your deployment without the complete standardization of the DOTS protocol.

Compared to these two approaches, the BloSS is more akin to DefCOM [5] than DOTS since it also builds on an existing system in the form of the Ethereum blockchain instead of developing a new communication approach from scratch. Leveraging the highly distributed nature and secure ledger aspects of blockchains allows the BloSS to securely and easily



(a) (b)

Fig. 5. (a) Large Dataset; (b) HTTP Comparison.

scale to the demand of a modern distributed DDoS defense system. DefCOM on the other hand relies on complex peer to peer message exchanges that are more prone to failure than the consensus-based blockchain system utilized by the BloSS.

A different approach is seen in [10], proposing a collaborative framework that allows the customers to request DDoS mitigation services from other domains. However, the solution is based on East-West bound protocols used in Software Defined-Network (SDN) controllers. These SDN controllers are then deployed at the customer side and interfaced with the SDN controller on the mitigation provider, which can change the label of the anomalous traffic and redirect them to security middle-boxes. Although SDN offers greater flexibility in network management, its restriction to SDN domains can make the solution not widely adopted.

V. Conclusions and Future Work

This paper presented an analysis of the blockchains applicability in a collaborative defense and provided a global evaluation regarding latency for signaling the respective addresses. While the analysis of application requirements and requirements justifying the use of blockchain points out that a private-permissioned blockchain is more appropriate for a cooperative defense, those performance results indicate that the approach in [8] performs well (less than 20 seconds in the worst case) for signaling large-scale attacks.

As future work, further improvements on the stability and security aspects need to be considered (*i.e.,* reputation mechanism to increase trust). Also, the integration of an incentive scheme is an essential feature to encourage stakeholders to collaborate in the defense. The BloSS source-code and deployment instructions are available.[1]

[1]https://gitlab.ifi.uzh.ch/rodrigues/bloss

References

[1] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," 2014. [Online]. Available: https://goo.gl/YBtGgc

[2] M. Feldman, K. Lai, I. Stoica, and J. Chuang, "Robust Incentive Techniques for Peer-to-peer Networks," *Proceedings of the 5th ACM conference on Electronic commerce*, pp. 102–111, 2004.

[3] G. Greenspan, "Avoiding the Pointless Blockchain Project," 2015. [Online]. Available: https://www.multichain.com/blog/2015/11/avoiding-pointless-blockchain-project/

[4] K. Nishizuka, L. Xia, J. Xia, D. Zhang, L. Fang, and C. Gray, "Inter-organization Cooperative DDoS Protection Mechanism," Internet-Draft, Draft, December 2016. [Online]. Available: https://goo.gl/Z2yMD8

[5] G. Oikonomou, J. Mirkovic, P. Reiher, and M. Robinson, "A Framework for a Collaborative DDoS defense," in *Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual Conference*. IEEE, 2006, pp. 33–42.

[6] P. Paganini. (2016, Sep.) 150,000 IoT Devices Behind the 1Tbps DDoS attack on OVH. [Online]. Available: https://goo.gl/NYDgi9

[7] T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of Network-based Defense Mechanisms Countering the DoS and DDoS Problems," *ACM Computing Surveys (CSUR)*, vol. 39, no. 1, p. 3, 2007.

[8] B. Rodrigues, T. Bocek, A. Lareida, D. Hausheer, S. Rafati, and B. Stiller, "Blockchain-based Architecture for Collaborative DDoS Mitigation with Smart Contracts and SDN," *11th International Conference on Autonomous Infrastructure, Management and Security (AIMS 2017)*, pp. 16–29, July 2017.

[9] B. Rodrigues, T. Bocek, and B. Stiller, "Enabling a Cooperative, Multi-domain DDoS Defense by a Blockchain Signaling System (BloSS)," *42nd IEEE Local Computer Networks (LCN). Demonstration Track*, pp. 1–3, Oct 2017. [Online]. Available: https://goo.gl/5TMFUt

[10] R. Sahay, G. Blanc, Z. Zhang, and H. Debar, "Towards Autonomic DDoS Mitigation using Software Defined Networking," *NDSS Workshop on Security of Emerging Networking Technologies*, 2015.

[11] H. Sanjuan. (2016, Dec.) IPFS Daemon Memory Usage Grows Overtime: Killed by OOM After a 10-12 Days Running. [Online]. Available: https://github.com/ipfs/go-ipfs/issues/3532

[12] Shelby, Zach and Hartke, Klaus and Bormann, Carsten, "The Constrained Application Protocol (CoAP)," 2014.

[13] G. Wood, "Ethereum: A Secure Decentralised Generalised Transaction Ledger," *Ethereum Project Yellow Paper*, vol. 151, jan 2014. [Online]. Available: https://goo.gl/LG7adX

[14] K. Wüst and A. Gervais, "Do you need a Blockchain?" 2017. [Online]. Available: https://eprint.iacr.org/2017/375.pdf

[15] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 2046–2069, April 2013.