The Knowledge Engineering Review, Vol. 35, e16, 1 of 17.
doi:10.1017/S0269888920000259

Orchestrating DDoS mitigation via blockchain-based network provider collaborations

ADAM PAVLIDIS[1], MARINOS DIMOLIANIS[1], KOSTAS GIOTIS[2], LOUKAS ANAGNOSTOU[2], NIKOLAOS KOSTOPOULOS[1], THEOCHARIS TSIGKRITIS[2], ILIAS KOTINAS[2], DIMITRIOS KALOGERAS[1], and VASILIS MAGLARIS[1]

[1]*Network Management and Optimal Design Laboratory (NETMODE), National Technical University of Athens, Athens, Greece*
e-mails: apavlidis@netmode.ntua.gr, mdimolianis@netmode.ntua.gr, nkostopoulos@netmode.ntua.gr, dkalo@netmode.ntua.gr, maglaris@netmode.ntua.gr
[2]*Technology R&D, PCCW Global, Athens, Greece*
e-mails: kyiotis@pccwglobal.com, lanagnostou@pccwglobal.com, ttsigkritis@pccwglobal.com, ikotinas@pccwglobal.com

Abstract

Network providers either attempt to handle massive distributed denial-of-service attacks themselves or redirect traffic to third-party scrubbing centers. If providers adopt the first option, it is sensible to counter such attacks in their infancy via provider collaborations deploying distributed security mechanisms across multiple domains in an attack path. This motivated our work presented in this paper. Specifically, we investigate the establishment of trusted federations among adjacent and disjoint network domains, that is, autonomous systems (ASes) that collectively mitigate malicious traffic. Our approach is based on Distributed Ledger Technologies for signaling, coordination, and orchestration of a collaborative mitigation schema via appropriate blockchain-based smart contracts. Reputation scores are used to rank ASes based on their mitigation track record. The allocation of defense resources across multiple collaborators is modeled as a combinatorial optimization problem considering reputation scores and network flow weights. Malicious flows are mitigated using programmable network data paths within the eXpress Data Path (XDP) framework; this enables operators with enhanced packet processing throughput and advanced filtering flexibility. Our schema was implemented in a proof-of-concept prototype and tested under realistic network conditions.

1. Introduction

Distributed denial-of-service (DDoS) attacks constitute major cybersecurity threats. To counter them, network providers either contract third-party organizations to scrub offending traffic or implement mitigation solutions within their own networking domain. However, the sheer volume of present-day cyber threats may overwhelm an individual provider, thus the pressing need for collaborative mitigation efforts. In particular, DDoS attacks are better pinpointed near the victim and more efficiently mitigated closer to their sources. However, defense collaborations might be hindered by operator concerns such as unwillingness to share victim-related information to preserve sensitive client data, lack of incentives for cooperation, and shortcomings of incident handling mechanisms.

In this paper, we propose an automated mechanism to orchestrate the collaborative mitigation of distributed attacks. Our schema fits best to Network and Wholesale Network Providers that share relevant and credible incident reports within a trust federation. Such insights are pieced together with (i) network programmability incorporated in next-generation networks, (ii) smart contracts (SCs) defining behavioral rulesets for all participants via blockchain networks, and (iii) procedures to evaluate and verify security

services. Our mechanism could also introduce a novel paradigm in business relationships with profitable interactions and reciprocal digital settlements among providers.

Network providers serve client organizations in their domains via interconnected autonomous systems (ASes). Our proposed solution enables an AS serving the victim of a distributed attack to receive and evaluate offers from collaborating domains found in the path of the attack, mitigating all or part of the offending traffic. This work extends our early reporting (Giotis *et al.*, 2018) in which we introduced a generic platform to properly identify appropriate mitigation offers, aiming to fully mitigate malicious traffic before reaching the premises of the victim. Our extensions enable a member of the trust federation under attack to (i) process all available mitigation offerings pertaining to a particular attack scenario, (ii) identify the optimal sets of flows that should be mitigated based on costs and historical records, that is, reputation scores, and (iii) announce these sets by issuing SCs toward the appropriate mitigation collaborators. This was modeled as a combinatorial optimization problem (generalized assignment problem—GAP) aiming at distributing defense resources in multi-domain attack paths.

The actual mitigation is undertaken by appropriately deployed filtering appliances within each collaborating domain. Notably, we introduce a reputation score that is calculated and maintained to evaluate the behavior of the respective collaborator on past incidents, while updating its credentials for future incident handling. We have also implemented a verification mechanism, capable of maintaining and exposing the necessary information for dispute settlement.

This paper is structured as follows: In Section 2, we discuss related work, while Section 3 offers a high-level architectural overview of the proposed solution. Section 4 analyses implementation details and related interactions between distinct architectural components. Section 5 presents experimental evaluations employing datasets of benign and malicious (DDoS) traffic. Finally, Section 6 highlights future steps and directions.

2. Related work

Various approaches have been proposed for the collaborative mitigation of massive DDoS attacks. Indicatively, CoFence (Rashidi *et al.*, 2017) is a framework that enables collaborating Network Functions Virtualization-enabled infrastructures (i.e., Internet Service Provider) to mitigate DDoS attacks using available compute and network resources. These are allocated in a reciprocal manner based on past mitigation collaborations. In Giotis *et al.* (2016), an Software-Defined Networking approach is proposed, featuring inter-domain collaboration via the exchange of Incident Object Description Exchange Format (The Incident Object Description Exchange Format, 2007) messages on top of Border Gateway Protocol (BGP); reputation score for neighbors is evaluated via the Beta Reputation system (Josang & Ismail, 2002). 3DCoP (2016) is also a P2P system for DDoS detection and mitigation whereby network domains collaborate to provide monitoring, alerting, and ultimately mitigation of malicious flows. Internet Engineering Task Force proposed the DDoS Open Threat Signaling (DOTS) protocol (Mortensen *et al.*, 2019) that specifies interactions between domains under attack and potential mitigators, while considering adverse network conditions and related limitations of the signaling channel. The establishment of business relationships and collaboration incentives is not a main objective in DOTS activities.

Approaches based on Distributed Ledger Technologies (DLTs) have recently been proposed as a promising way to enhance the coordination between collaborators for detection and mitigation of security incidents. Indicatively, Malomo *et al.* (2018) propose a federation, whereby monitoring data are exchanged to collaboratively detect network anomalies. The federation is based on a permissioned blockchain framework that ensures transparency and business regulation. Similarly, in Kim *et al.* (2018), the use of a private blockchain is proposed to avoid verification delays commonly occurring in public ledgers. Another approach aims at providing DDoS mitigation services to third parties via sharing of user resources; this is orchestrated in Gladius, an Ethereum-based platform (Decentralized CDN, WAF, and DDoS protection, 2018) that verifies web requests and drops illegitimate ones.

In Rodrigues *et al.* (2017), a cross-domain collaborative schema for DDoS mitigation is introduced, whereby the cooperation signaling relies on an Ethereum network. Malicious IP addresses are advertised in blockchain-based SCs issued by the victim. These are retrieved by interested ASes which in turn may

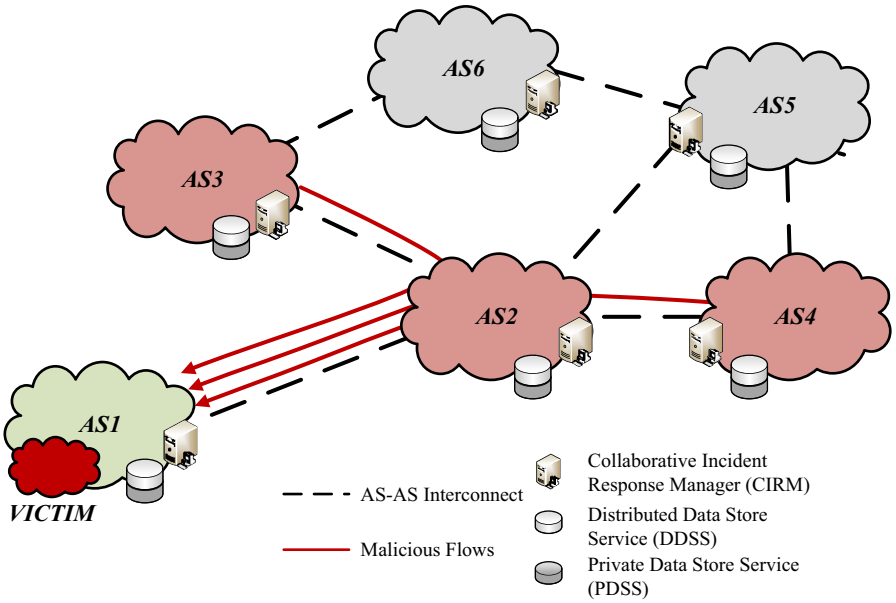

Figure 1 High-level overview of our use case

trigger mitigation actions. However, inserting large blocks of IP addresses in the blockchain may introduce significant latencies to the mitigation process. As an extension to Rodrigues *et al.* (2017), authors presented in Gruhler *et al.* (2019) a reputation scheme based on the Beta Reputation system in order to rate mitigation services and prevent abuse/misuse by modeling customer strategies. Additionally, Mannhart *et al.* (2018) explore mechanisms for verifying an attack was indeed mitigated to prevent false reporting.

In our early work (Giotis *et al.*, 2018), we provided an initial design for blockchain-based Federation of Collaborating Autonomous Systems. Inspired by Giotis *et al.* (2016), Rodrigues *et al.* (2017), and Gruhler *et al.* (2019), we designed and implemented a modular collaborative, Blockchain-powered platform for mitigating massive DDoS attacks. Our approach primarily employs Blockchain as a formal communication channel between network providers in an attempt to solidify business relationships.

Similarly to related efforts on collaborative DDoS defense (Giotis *et al.*, 2016; Gruhler *et al.*, 2019), we adopted the Beta Reputation system and appropriately modified it to weigh partner contributions in the mitigation process (i.e., number of malicious sources blocked). Furthermore, other approaches (e.g., Rashidi *et al.*, 2017; Giotis *et al.*, 2016) employ scores that characterize a domain's past behavior to decide on how to allocate mitigation resources or whether to provide assistance. Alternatively, given specific offerings from potential mitigators, we focus on enabling the victim to distribute mitigation actions (i.e., sources to be blocked) to the most reliable (e.g., higher reputation) AS/ASes. The distribution process is formulated as a cost optimization problem that considers the reputation of potential mitigators and may also include additional operational parameters and business policies. Finally, disputed mitigation claims are settled via a verification procedure relying on network monitoring data as provided by the victim and mitigator ASes; other mechanisms have been considered in Mannhart *et al.* (2018) that require infrastructure access and specialized equipment for verification purposes. Note that we considered a trusted federated environment whereby collaborators follow agreed upon admission procedures and adhere to standards as in Mutually Agreed Norms for Routing Security, 2016, a global initiative among network providers.

3. Overview and baseline design

3.1 Design principles

A high-level overview of our schema is depicted in Figure 1. Malicious flows (continuous lines) that collectively form a DDoS attack originate from—or transit through—interconnected federated ASes (dashed

lines). Instances of the Collaborative Incident Response Manager (CIRM), deployed in each AS, are responsible for coordinating the mitigation of the malicious flows and registering all interactions within two distinct data stores based on DLT. The entire process is orchestrated using data (logged messages, encrypted sensitive information) stored in a federation-wide Distributed Data Store Service (DDSS). CIRM also takes advantage of *ad hoc* instances, realizing the respective Private Data Store Services (PDSS), between sets of ASes. These are used to support the exchange of additional (private and/or sensitive) information and to automatically settle the resulting agreements. We assume that malicious flows are accurately detected by the victim or the network provider of the victim (*AS1*) via an independent detection mechanism; this process is considered to be outside the scope of this paper.

Upon detecting a distributed attack, the *victim AS* (*vAS* henceforth) issues an SC for the incident; this digital agreement includes among others (i) the adjacent neighboring domains that forward attack traffic, (ii) a URL pointing to a document containing the DDoS attack sources stored within a distributed file storage system (we adopted Inter-Planetary File System—IPFS 2015), and (iii) the encrypted IP address(es) of the victim. ASes update the agreement above to include their adjacent ASes located in the attack path. Once the initial SC is issued, each AS located in the path of the attack is considered as a possible *mitigator AS* (*mAS* henceforth) and may offer via relevant SCs to fully or partially block malicious traffic in exchange for reciprocal benefits. The *vAS* is then able to coordinate with the federated ASes leveraging the various types of SCs available in DDSS and PDSS. The resulting mitigation plan is obtained by feeding appropriate data to the cost optimization and reputation algorithms implemented as applications on top of each CIRM. The different types of SC implemented in our framework, as well as details on the related applications, are further documented in Section 4.

The design principles of our federated schema are as follows.

- *Privacy-aware propagation of malicious network events and out-of-band communication:* Our schema enables a *vAS* to (i) report a detected network attack respecting the privacy of sensitive information and (ii) propagate the report to ASes along the attack path. We consider that ASes under attack potentially face adverse network conditions (e.g., link saturation), severely impairing communication. To that end, we assume that participants maintain dedicated channels used for collaboration.
- *Collaborative mitigation of security incidents:* Federated partners are able to collaboratively deploy appropriate mitigation mechanisms. Our decentralized approach tends to push filtering rules near attack sources thus alleviating the victim domain from the total burden of a highly distributed attack. This schema refines commonly used blackholing techniques that may lead to blind service disruption.
- *Reputation-based federation:* Our proof-of-concept implementation reflects a federated multi-domain network environment whereby partners are rated by reputation scores for past incident handling. The reputation score depicts the quality of the mitigation service provided, that is, the capability of a collaborator to consistently block malicious flows.
- *Accountability and consensus:* Collaboration schemas need to account for scenarios whereby partners may have diverging incident handling priorities or unintentionally report inaccurate information. As such, federation members should be able to (i) log transactions, (ii) verify logged transactions, (iii) enforce service-level agreements, and (iv) settle disputes via appropriate verification mechanisms.

Our vision is that such capabilities will introduce business workflows with potential financial benefits (rewards), thus incentivizing members of the federation to deliver high-quality mitigation services.

3.2 Architectural components

Figure 2 presents an overview of our proposed framework, as well as the interactions between the main components, namely (i) the CIRM and related applications built on top, (ii) the distributed ledgers realizing the Data Store Services (i.e., DDSS and PDSS), and (iii) the attack mitigation appliance (AMA).

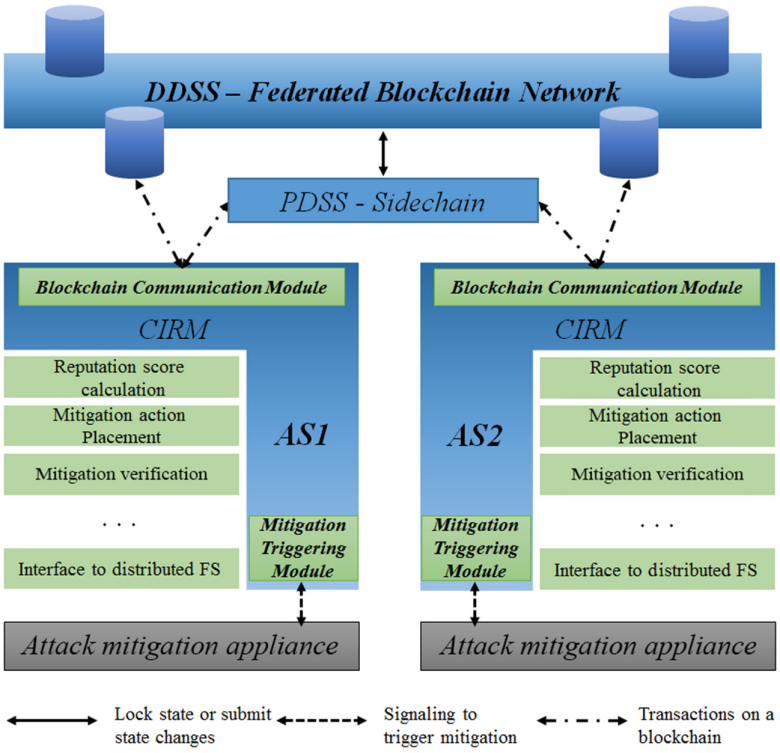

Figure 2 Proposed framework and component interactions

3.2.1 Collaborative Incident Response Manager

CIRM components are responsible for coordinating the mitigation effort within the federation. A CIRM is initially triggered by alerts, generated by DDoS detection schemas, for example, Giotis *et al.* (2015). Our approach assumes that incident alert messages contain the following information: (i) a set of network flows related to the incident extracted via NetFlow/sFlow (Claise, 2004; Phaal & Lavine, 2004) (i.e., a tuple of source IP, destination IP, source port, destination port, and protocol) and (ii) the initial incident timestamp. CIRM then creates an attack information document (AID) containing the incident flows, as well as its SHA-256 digest to be used for AID integrity check. This is stored off-chain, in a distributed file storage system (IPFS) accessible by all federation members.

The CIRM design is modular, consisting of two core elements, namely, the Blockchain Communication Module (BCM) and the Mitigation Triggering Module (MTM). Specifically, BCM is used to register and perform transactions to the DDSS and the relevant PDSS instances in order to receive, acknowledge, or forward mitigation requests (as well as the related information) to other ASes. The communication between CIRM and the Distributed Data Stores is facilitated by deploying or updating instances of the various types of SCs we have implemented, as described in Section 4. MTM is responsible for the communication with mitigation appliances upon the conclusion of transactions in DDSS and PDSS blockchain networks, triggered by a particular incident. In this paper, MTM conveys malicious sources to an XDP-based mitigation appliance that implements the actual mitigation process.

Apart from the above core modules, the proposed framework relies on the following set of support applications, built on top of CIRM, that enable *vAS* to select the appropriate collaborators within the federated network and orchestrate the mitigation:

- *Reputation score calculation:* The cooperation track record of each AS in the federation is evaluated using SC attributes related to DDSS and PDSS past transactions. These are supplemented with monitoring metrics (e.g., verified blocked flows per incident) and dispute resolution results.
- *Mitigation action placement—cost optimization*: *vASes* are able to retrieve and evaluate offers posted in the PDSS for the mitigation of malicious sources related to a specific incident. The evaluation is

formulated as a GAP tailored to the flows to be dropped and the respective reputation scores of a collaborating *mAS*.

- *Mitigation verification*: This application facilitates business-oriented verification, that is, verify that the appropriate flows were actually blocked in our specific use case, indicating whether the victim should consider the respective SC as fulfilled and consequently close it.
- *Interface to distributed file storage:* Large blocks of information are stored and retrieved from an external distributed file storage (e.g., IPFS) interworking with blockchain-based architectures to separately store files such as AIDs which may include hundreds of thousands of malicious sources. The CIRM application facilitates the communication with the distributed file storage system.

3.2.2 Data store service

Our architecture requires a form of distributed transactional data store available to all participants, accompanied by a consensus mechanism. As mentioned, the platform is not open to the public Internet but only to approved members of the federation. Thus, there is no specific need to use complicated or computationally intensive consensus mechanisms (e.g., proof of work). We adopted a permissioned blockchain in which predefined miners (i.e., federated participants) are authenticated and authorized within the blockchain infrastructure. To that end, our implementation is based on a blockchain-based federation (Buterin, 2015) using the Proof-of-Authority (PoA) (2017) consensus mechanism, to fulfill the requirements mentioned above. Another objective is to preserve the victim's privacy (e.g., IP address) even within a trusted federation, while enabling on-demand reward and settlement of verifying transactions. This is achieved via dedicated *ad hoc* communication channels between collaborating ASes, forming private blockchain networks, that is, Sidechains (Back *et al.*, 2014). Such Sidechains enable exchanging and updating of information related with an SC stored on the public blockchain network (Mainchain henceforth). Note that in our architecture, Mainchain realizes the DDSS component, while each Sidechain is a PDSS instance between two collaborating ASes. Recall that, both types of blockchain networks use dedicated out-of-band connections.

3.2.3 Attack mitigation appliance

The AMA represents a major component of the proposed architecture, responsible for the actual mitigation of malicious traffic. While any mitigation appliance could be an appropriate candidate, we implemented and deployed a flexible fine-grained and high-performance mitigation mechanism, based on the XDP (Høiland-jørgensen *et al.*, 2018). Triggered by the MTM, this component is deployed in each AS and drops all IP sources described by MTM as malicious, as further described in Section 4.

4. Proposed architecture: implementation details

4.1 Blockchain-based SCs

The proposed approach leverages both public and private distributed ledger instances (i.e., Mainchain and Sidechains) as transactional distributed data stores. For our implementation, we selected Ethereum (2015) as the underlying platform of our distributed application, mostly due to the sizable resources and community (Decentralized CDN, WAF, and DDoS protection, 2018; ConsenSys – Harness the power of Ethereum, 2014) surrounding the project. However, the proposed framework is not conceptually intertwined with a particular DLT.

Upon initialization, all federation members have a unique Ethereum address and a private key, authenticating them against their Ethereum account. Their public Ethereum address is explicitly configured within the Genesis block of the network (i.e., a JavaScript Object Notation file containing static configuration parameters related to the network). Regarding the Mainchain, all federation members assume the role of a *Sealer*, that is, they can validate any network-generated block. Notably, in a PoA deployment such as the one described for Mainchain, only accounts specified on the Genesis block as *Sealers* are eligible for block verification. The Ethereum architecture also utilizes the *Bootnode* that assists the respective BCM nodes of each member in discovering the DDSS network. Thus, each BCM can connect

to the DDSS as an authorized member of the network. The PDSS is deployed following the same principles as the DDSS, with the only difference being that each PDSS is a PoA network between only two ASes, and only these two are configured as *Sealers*.

The business logic in both DDSS and PDSS networks is introduced via different archetypes of SCs developed in *Solidity* Programming Language (2019). Once an SC is deployed to the appropriate PoA network via the BCM, it is assigned a unique address in the form of a hexadecimal number. In order to interact with an SC, Ethereum defines an *Application Binary Interface* (*ABI*), as the data encoding scheme. Thus, SC data are encoded according to their type, based on the information described in the respective ABI. The ABI specification is shared among all federation participants upon initialization of a blockchain network, since without this information the BCMs would not be able to decode and interact with any deployed SCs.

We implemented four types of SCs: (i) $SC^{(A)}$ is issued (owned) by the victim AS and contains basic information about the incident; (ii) $SC^{(B)}$ is used to recursively notify potential mitigators; (iii) $SC^{(C)}$ is issued by transit ASes, offering to block malicious flows; and (iv) $SC^{(D)}$ is issued by the victim AS to finalize collaborative mitigation assignments. All four SCs are used to orchestrate collective actions across an incident attack path.

In Table 1, we present specific attributes of the aforementioned SC types. Specifically, the current status attribute may be modified by the *vAS* and/or *mAS* as denoted in the table. In the last column, we indicate the specific component of the Data Store Services (DDSS or PDSS) that each SC type should be deployed on.

4.2 Orchestration workflow

The steps described in this subsection constitute a recursive process between adjacent ASes within a federation-wide attack graph. This includes (i) the notification mechanism of all ASes in the graph and (ii) the propagation mechanism of decryption keys.

The CIRM instance advertises the necessary information (e.g., malicious flows within *AID*, hiding privacy sensitive victim identification) to DDSS via an $SC^{(A)}$. Subsequently, the CIRM deploys $SC^{(B)}$ instances on the PDSS of each adjacent AS found in the path of the attack; these include the decryption key for the victim IP. Initially, the decryption key is sent by the victim to adjacent nodes, and then, it is recursively propagated to all potential mitigators (i.e., adjacent ASes that forward attack traffic). Each $SC^{(B)}$ is fulfilled upon acknowledgement by the adjacent AS.

As mentioned, federated members additionally maintain secure out-of-band communication channels to exchange signaling messages. Optionally, we could also employ asymmetric encryption (e.g., Public Key Infrastructure, GNU Privacy Guard) to encrypt and distribute the decryption key. Note that, the decryption key itself is kept secure within the infrastructure of a federated domain, using well-known and accepted security practices (e.g., hardened systems, firewalls, physical/logical access control, and auditing).

ASes in the attack path should update the disclosed attack graph within $SC^{(A)}$, to include their relevant neighbors that forward malicious traffic. The attack graph is represented and stored as an array of linked lists. Each array record represents a given AS of the federation, while each list represents a valid branch of the attack graph. In addition to $SC^{(B)}$-related communication, each AS may retrieve blockchain-verified $SC^{(A)}$ instances via their respective BCM. Thus, by inspecting the attack graph, an AS may decide on their role in forwarding ongoing attacks (i.e., whether they belong to the attack path).

ASes on the attack path may offer to assist (placing a bid) the *vAS* in the mitigation of the attack by deploying an $SC^{(C)}$. This SC maintains an attribute pointing to a list of source IP addresses that can be blocked by the bidding AS. The list of source IP addresses that can be blocked by the *mAS* is also stored in the IPFS (2015). All $SC^{(C)}$ instances are stored within the PDSS; they may be retrieved by the CIRM and conveyed to the Mitigation Action Placement application to select appropriate bids.

The original $SC^{(A)}$ owner will create $SC^{(D)}$ in the DDSS for all the approved $SC^{(C)}$ bids. Upon the creation of the relevant $SC^{(D)}$, each *mAS* should start the mitigation process and update the status attribute of the $SC^{(D)}$ to 'Fulfilled'. All fulfilled $SC^{(D)}$s should then be verified by the owner of $SC^{(A)}$ in order to

Table 1 Types of smart contracts

SC type	Attributes	Short description	Deployed
SC$^{(A)}$	Owner	Public Ethereum address	DDSS
	Status	Current status—modification rights • Seek assistance—*vAS* • Attack graph update—*vAS*, *mAS* • Bidding closed—*vAS* • Under mitigation—*vAS* • Under verification—*vAS* • Fulfilled—*vAS* • Not fulfilled—*vAS*	
	vIP	Encrypted victim IP address	
	Graph	Attack graph	
	AID_URL	IPFS URL to retrieve AID	
	AID_Digest	AID SHA-256 digest	
SC$^{(B)}$	SCA_Address	Address of relevant SC$^{(A)}$	PDSS
	Owner	Public Ethereum address	
	Status	Current status—modification rights • Sent—*vAS* • Acknowledged—*mAS*	
	M_Address	DDSS address of a potential mitigator	
	D_KEY	Decryption key	
SC$^{(C)}$	SCA_Address	Address of relevant SC$^{(A)}$	PDSS
	Owner	Public Ethereum address	
	Status	Current status—modification rights • New bid—*mAS* • Approved—*vAS* • Rejected—*vAS*	
	SIP_URL	IPFS URL for the list of source IP addresses to block	
	Reward	Requested reward for mitigation	
SC$^{(D)}$	SCA_Address	Address of relevant SC$^{(A)}$	DDSS
	Owner	Public Ethereum address	
	Status	Current status—modification rights • Offer—*vAS* • Fulfilled—*mAS* • Verified- *vAS*	
	M_Address	DDSS address of a potential mitigator	
	Reward	Reward for mitigation	

complete the transaction allowing bidders to claim the agreed reward. Finally, once all the SC$^{(D)}$s of the initial SC$^{(A)}$ have been verified, the SC$^{(A)}$ is fulfilled and updated accordingly in the DDSS.

4.3 Reputation schema for collaborating entities

Each AS is characterized by a reputation score, representing the quality of mitigation service provided to members of the federation, pertaining to past incidents. The proposed reputation schema is based on

the *Beta(a,b)* distribution (Josang & Ismail, 2002). The reputation value after n incidents is equal to the mean value of the beta distribution, $rep_n\,(AS) = a_n/\ (a_n + b_n)$ and represents a 'forecast' for the outcome of future mitigation tasks. Such information is available to all ASes in the federation and may be used to appropriately assign mitigation tasks. Similar approaches (Giotis *et al.*, 2016; Gruhler *et al.*, 2019) that utilize Beta distribution consider only whether an AS provides mitigation service, disregarding the magnitude of its contribution. Hence, we extended the update process for the values of shape parameters, a_n and b_n, quantifying an AS's assistance, according to the following equations:

$$a_{n+1} = \gamma \cdot a_n + B_n,\ b_{n+1} = \gamma \cdot b_n + N_n$$

.The variables B_n and N_n represent, respectively, the flows blocked and not blocked, for a given instance n of an $SC^{(D)}$. Both values are based on the flows that a specific AS is assigned to block via $SC^{(D)}$ and not the total flows involved in the attack incident. Note that γ is a discount factor affecting past reputation scores; $\gamma = 1$ means that the reputation score incorporates the cumulative contribution of a contributor along the past n incidents, whereas $\gamma = 0$ indicates that only the last mitigation service offered may affect the reputation score. We used the former in our experiments.

In a typical scenario, B_n should be equal to the length of the list of flows the respective $SC^{(D)}$ is pointing to. However, in case of disputes between an *mAS* and *vAS*, the two ASes calculate and compare the exact number of blocked sources for an *ad hoc* bilateral settlement. This is further discussed in Section 4-6.

4.4 Cost optimization —mitigation action assignment

We treat the aforementioned evaluation of available offerings to assist in the mitigation of an attack event as a GAP with the following integer programming formulation:

$$maximize \sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}x_{ij}$$

Subject to

$$\sum_{j=1}^{n} w_{ij}x_{ij} \le C_i,\ i = 1, ..., m$$

$$\sum_{i=1}^{m} x_{ij} = 1,\ j = 1, ..., n$$

$$x_{ij} \in \{0, 1\},\ i = 1, ..., m\ and\ j = 1, ..., n$$

From the *vAS* perspective, the GAP is considering the assignment of n items to m bins. Assignment of an item j to bin i yields a reward r_{ij} and carries a weight of w_{ij}. A feasible solution dictates the total weight of assigned items for each bin should not exceed the capacity of the bin (C_i), and each item should be assigned to exactly one bin. The optimal solution is an assignment maximizing the sum of all the rewards. In our case, each item represents the filtering of a malicious flow via a respective entry on a mitigation appliance (e.g., a Ternary Content-Addressable Memory entry; Giotis *et al.*, 2015) with weight equal to one. Network providers within the described federation are represented by bins, while their capacity denotes the available mitigation resources. Finally, the GAP reward r_{ij} is calculated as the product of reputation score of a collaborator i and the importance of the flow j, where the importance denotes the amount of bytes corresponding to the flow j. Then,

$$r_{ij} = rep(i) \cdot flow\_imp(j).$$

Intuitively, our approach tends to assign the most important flows, for example, heavy hitters, to the collaborators with the higher reputation values. Consequently, the victim tends to receive better mitigation service for a specific incident. In turn, collaborators apart from any direct rewards as part of the agreement establish and maintain a high reputation score increasing the possibility to be selected in the future.

GAP is an NP-hard problem, and its input size affects the time needed to be solved. In our previous work (Dimolianis *et al.*, 2019), we considered methods for reducing the input size of the algorithm using grouping and/or prefix aggregation techniques. These methods are also applicable to the aforementioned formulation and can be used for reducing the execution time of the solver, that is, Dippy integer programming tools (O'Sullivan *et al.*, 2011).

4.5 Implementation of mitigation mechanisms

We have considered a number of solutions that can be signaled by the MTM to block a set of malicious source IP addresses. Each *mAS* maintains one or more AMAs capable of filtering malicious sources. The MTM will be responsible to trigger the communication with an AMA via protocols including but not limited to BGP, Network Configuration Protocol, and OpenFlow. Protocols such as BGP Flowspec (Marques *et al.*, 2009) and OpenFlow (McKeown *et al.*, 2008) might be used to disseminate forwarding rules to appropriately configured devices offering fine-grained matching capabilities and actions. Network traffic matching those rules could be either redirected to in-house or cloud-based scrubbing mechanisms for further analysis or blocked immediately. The approaches above offer different capabilities in terms of packet matching (coarse-grained and fine-grained), communication method (BGP, Secure Shell/HTTP, and OpenFlow), and available actions (drop, rate limit, and redirect). Another emerging technology, the XDP (Høiland-jørgensen *et al.*, 2018), is an in-kernel Linux framework that enables on-demand packet processing at the lowest layers of the OS stack. This allows Common Off the Shelf hardware, for example, Linux machines, to execute programs written in C programming language, at the Network Interface Card driver level providing high-performance packet dropping capabilities; XDP is considered a promising candidate for inline DDoS mitigation and customized monitoring solutions (Bertin *et al.*, 2017).

We implemented the AMA leveraging on the XDP capabilities that allow matching of specific packet fields in the filtering process. Specifically, our implementation parses packet headers, matches the source IP addresses of malicious actors coupled with the destination IP of the victim, and drops it, whereas legitimate traffic is forwarded normally to its destination. Note that we could utilize the capabilities offered by XDP to implement a DDoS mitigation solution that is not dependent only on IPs but combines other packet features, tailored to specific attack vectors.

4.6 Verification of mitigation agreements

The proposed framework relies on the existence of a trusted federation, composed of ASes operating according to a predefined policy (e.g., Mutually Agreed Norms for Routing Security, 2016). Hence, we assume that mishaps in contract fulfillment should not occur due to malicious intent but as a result of defense mechanism malfunction and/or misconfigurations. Consequently, operating in this environment cannot completely alleviate the need for mitigation verification of $SC^{(D)}$s.

To that end, we consider an approach that leverages monitoring data collected from the peering (interconnection) links of the ASes. Specifically, each *mAS* should maintain flow information for the egress traffic directed toward the *vAS*. At the same time, *vAS* should maintain the respective data for ingress traffic on each corresponding link. Malicious flows pertaining to a particular incident are extracted from appropriate monitoring data and are stored in distinct timeframes (i.e., 60 seconds); subsequently, they can be retrieved upon request, for dispute settlement purposes. Note that flows are identified by their source IP; additional information may include destination IP, protocol number, and TCP/UDP ports.

Moreover, aiming to tackle potential storage issues that might arise in cases of extremely distributed attacks, we opted for maintaining these malicious flows within a Bloom Filter probabilistic data structure (Bloom, 1970; Van Rijswijk-Deij *et al.*, 2019). We based this selection on the observation that the actual counters are not required for the verification process. Hence, the malicious flows related to an attack incident are hashed in space-efficient Bloom Filters, each pertaining to a particular time window of an incident. Indicatively, a Bloom Filter of 176 KBytes and 10 Hash Functions would be sufficient to accurately store 100°000 malicious sources (e.g., Memcached DDoS attack, 2018), with a false positive probability of 0.1%.

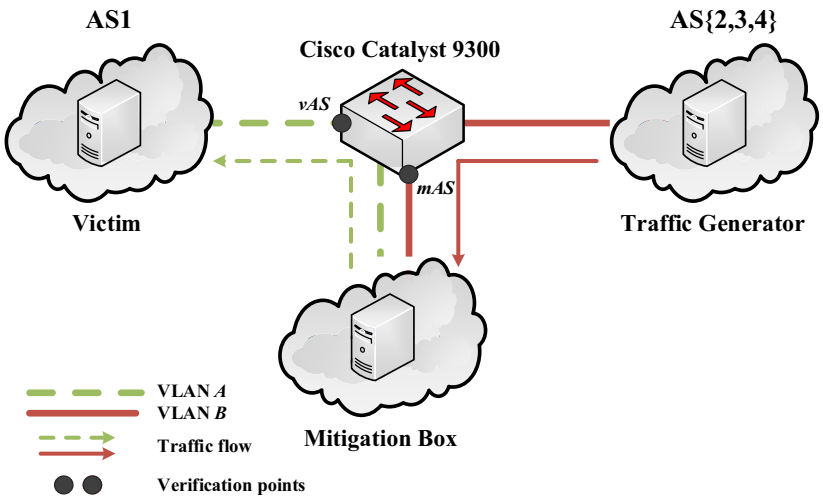

Figure 3 Proof-of-concept testbed setup

Typically, network operators employ widely adopted protocols for network monitoring, such as NetFlow (Claise, 2004) or sFlow (Phaal & Lavine, 2004); these are usually combined with sampling mechanisms to alleviate load on network devices. We employed NetFlow monitoring data with varying sampling rates as further discussed in Section 5. Since there is a trade-off between sampling rate and monitoring accuracy, there might be inconsistencies in the data collected that are not due to faulty mitigation but to visibility limitations of the sampling process. However, we do not expect the accuracy under massive attacks to be impacted by sampling (Giotis *et al.*, 2015).

5. Experimental evaluation

5.1 Proof-of-concept experimental setup

We implemented a proof-of-concept testbed for experimental evaluation deployed at the NETMODE Laboratory of the National Technical University of Athens, Greece. We considered the following areas for experimentation: (i) Reputation Score Calculation, (ii) Mitigation Action Placement, and (iii) Mitigation Verification.

Our testbed incorporates three Linux machines operating as distinct elements: (i) the Traffic Generator, (ii) the Mitigation Box (emulating the AMA), and (iii) the Victim. As depicted in Figure 3, machines are connected in a star-like topology, with a Cisco Catalyst 9300 Multilayer Switch at the center. All the links operate at 10G.

The Traffic Generator is responsible for replaying and multiplexing benign and malicious network traffic, emulating traffic originating from *AS2*, *AS3*, and *AS4* as depicted in Figure 3. The traffic is directed to the victim via the Mitigation Box with two bridged interfaces, each in a separate Virtual Local Area Network. Appropriate XDP programs, applied in the ingress interface of the Mitigation Box, are used to block malicious sources while allowing benign traffic. Verification points (*vAS*, *mAS*) are implemented as separate flow exporters on the NetFlow-enabled switch.

In our experiments, we used both benign and malicious traffic. The legitimate traffic is based on the *CAIDA Anonymized Internet Traces 2016* (The CAIDA UCSD Anonymized Internet Traces, 2016) dataset. The attack traces we used (Santanna *et al.*, 2015) contain real traffic generated from DDoS-as-a-Service platforms (commonly referred to as Booters or Stressers). These traces were captured during a controlled experiment at the University of Twente in collaboration with SURFnet, the Dutch National Research and Education Network. We synthesized a total mixture of traffic that emulates a production network environment under attack; it has an approximately 1.77 ratio of malicious to benign traffic, with the total average packet rate approximately at 584 Kpps. In our experiments, malicious flows are aggregated via their source IP attribute; we further assume that each AS blocks a flow (source) assigned

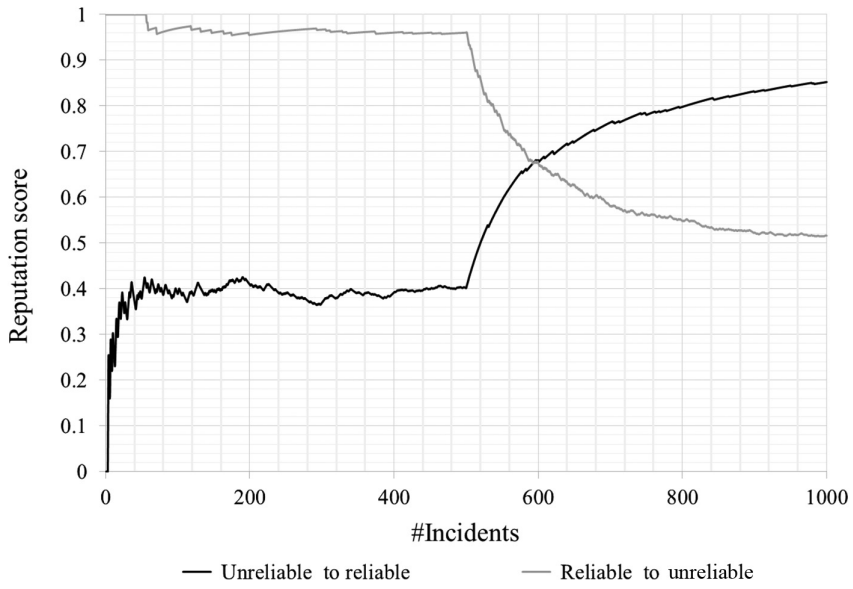

Figure 4 Reputation score during 1000 consecutive incidents for different types of federated collaborators

to it with a probability equal to its reputation score since the latter ranks an AS mitigation service quality. We expect some incidents for which the mitigation fails to meet the promised requirements. As discussed in Section 4, we considered that such incidents might occur due to human errors (misconfiguration of the security appliance) or hardware malfunction.

Moreover, aiming to evaluate the DLT-based orchestration of the proposed federated schema, we deployed an Ethereum blockchain network consisting of six nodes, based on the *Go Ethereum* implementation (Go Ethereum, 2019). Along with each node, we deployed a CIRM instance and a *Sealer* service, with the authority to validate each block on the blockchain network. Additionally, we deployed two more nodes for coordination and monitoring purposes: (i) *Bootnode* that facilitates the interconnectivity of the blockchain nodes by providing them with the necessary details to connect with the correct blockchain network; (ii) *Monitoring node* that retrieves statistics and displays information about the current status of the blockchain network as in Ethereum Network Intelligence API (2016) and Ethereum Network Stats (2016).

5.2 Reputation score calculation

To evaluate our proposed reputation mechanism, we selected three types of federated collaborators: (i) a *reliable* mitigator blocking a high percentage of the agreed malicious flows, (ii) an *unreliable* mitigator blocking a low percentage of the agreed malicious flows, and (iii) an *unpredictable* mitigator, whose performance might vary both regarding the success rate and the volume of the promised flows. We considered two simulation experiments to assess the impact of mitigation service to the reputation score.

In the *first experiment*, we assume that a *reliable* mitigator AS (*mAS*) blocks the agreed malicious sources (i.e., 1000 flows) with a probability ranging from 90% to 100% for the first 500 incidents. For the next 500 incidents, the *mAS* behaves as an *unreliable* mitigator, blocking the agreed sources with a probability ranging from 40% to 50%. We also consider the opposite, that is, an unreliable mitigator that turns reliable. In both cases, we calculate the reputation score and present its evolution during 1000 consecutive attack incidents.

As expected, Figure 4 indicates that the reputation score closely resembles the performance of the *mAS* during past attacks. Note that, the proposed schema allows collaborators to recover from low reputation scores in case they start providing high-quality mitigation services, while low-quality ones result to a significant drop in the reputation score. The rate of recovery depends not only on their reliability but

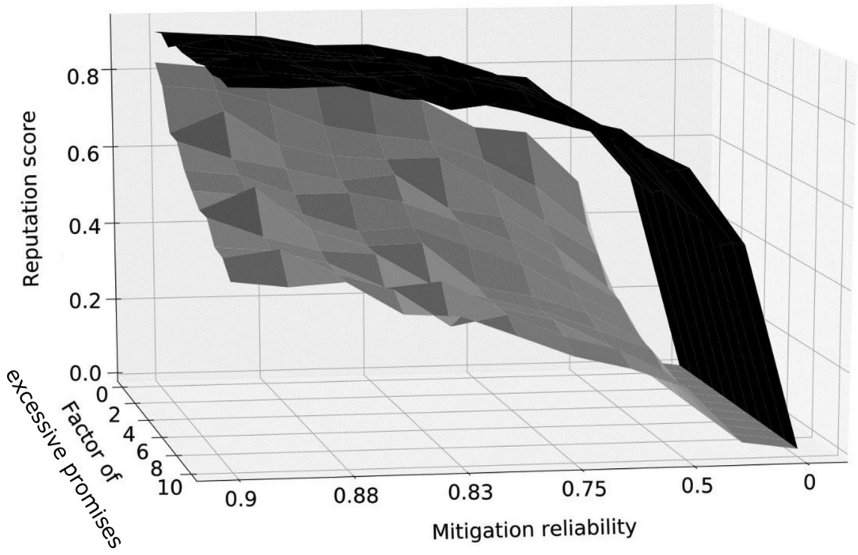

Figure 5 Reputation score comparison—binary reputation schema (Black) and proposed approach (Gray)

also on the volume of promised flows. Thus, an AS is able to establish a high reputation score faster by seeking and fulfilling SCs for incidents involving large numbers of malicious sources.

In the *second experiment*, we compare our proposed schema against approaches that consider only binary outcomes, that is, a collaborator is providing assistance or not (Giotis *et al.*, 2016; Gruhler *et al.*, 2019). Specifically, we showcase the reputation score of the *unpredictable* collaborator mentioned above that exhibits inferior performance in terms of (i) mitigation reliability and (ii) inability to fulfill excessive mitigation promises in an SC. The former indicates the probability that an attack will be mitigated; for simplicity, we consider that the entire batch of agreed flows in an SC is either mitigated successfully or not. The latter defines a multiplier for the number of flows the collaborator promised but was not able to block. Specifically, in our simulation experiments, we used 1000 flows as the base value.

In Figure 5, we compare the 'binary' reputation schema (black surface) and our approach (gray surface) that considers the number of malicious flows promised but not blocked. Each point in the figure is the average reputation score calculated after 1000 consecutive DDoS attacks emulated in our testbed. The binary approach is only affected by the mitigation reliability and yields significantly higher reputation scores. Our approach also accounts for the actual number of mitigation actions successfully deployed by a collaborator and yields lower reputation scores, especially when the *mASes* fail to block large numbers of malicious flows (i.e., high factor of excessive promises). As an example, a mitigation reliability of 0.9 and a factor of excessive promised flows of 10 mean that a collaborator successfully mitigates an attack blocking 1000 flows with probability 0.9, while it fails to block excessive numbers of promised flows (10 000 flows) with probability 0.1. For this case, our schema would rank the *mAS* with a reputation score of 0.5, whereas 'binary' reputation schemas would yield a reputation score of 0.9.

5.3 Mitigation actions placement

To evaluate the mitigation actions placement on collaborating ASes, we considered the topology depicted in Figure 1. We analyzed the 'Booters 1' attack dataset (Santanna *et al.*, 2015) and allocated all IP sources within distinct /24 subnets, distributed uniformly to *AS2*, *AS3*, and *AS4*. This coupled with the actual attack dataset resulted to attack traffic emanating from the three ASes in proportion to 33%, 31%, and 36% of the total attack traffic. Note that, attack traffic originating from *AS3* and *AS4* may be also blocked by *AS2*. In our experimentation process, the *mASes* offer to block the malicious traffic that flows through their network and the *vAS* assigns mitigation actions based on the aforementioned approaches. The cumulative capacity of all *mASes* is sufficient to collectively block the attack flows.

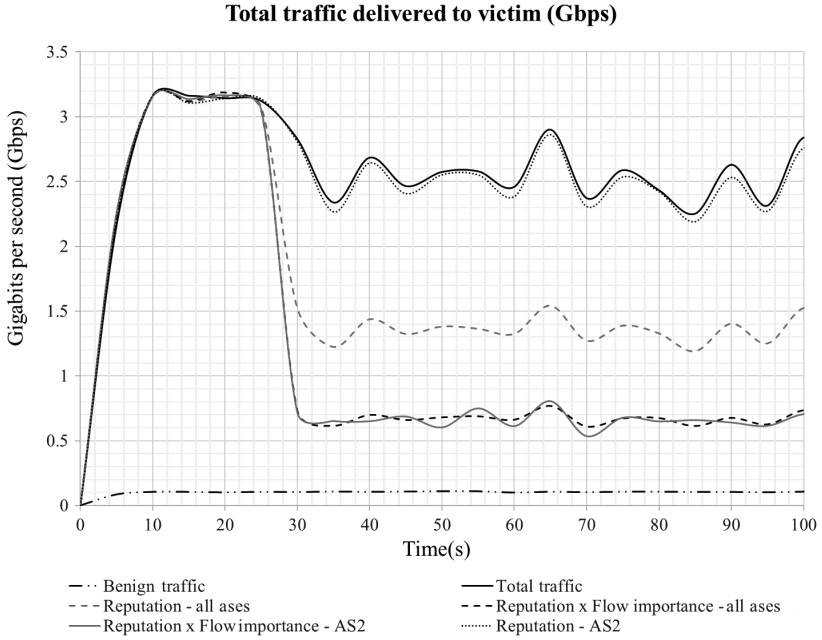

Figure 6 Total malicious and benign traffic reaching the victim

We consider two options for the mitigation actions placement using the following reward functions: $r_{ij}^{a} = rep\,(i)$ and $r_{ij}^{b} = rep\,(i) \times flow\_imp\,(j)$. Note that, in both cases, all malicious flows are assigned to some *mASes* but possibly in a different manner: The former tends to assign mitigation actions to the *mASes* with the best reputation score. The latter additionally considers the importance of a flow for the mitigation process (i.e., flow importance). Flow importance is a metric identifying the danger a specific flow represents and may be calculated via sophisticated Intrusion Detection and Intrusion Prevention systems that evaluate multiple parameters across various attack vectors. This formulation enables operators to define policies focusing on traffic features they consider as the highest risks to their infrastructures. In our experiments, we adopted the total amount of bytes that correspond to each malicious flow, considering typical volumetric DDoS attacks.

We showcase the mitigation of a real-time attack scenario, employing again the same traffic mixture of benign and malicious traffic. We consider four different mitigation approaches: (i) *vAS* assigns mitigation actions to all ASes based on: r_{ij}^{b}, (ii) *vAS* assigns mitigation actions to all ASes based on: r_{ij}^{a}, (iii) *vAS* assigns mitigation tasks only to *AS2* based on: r_{ij}^{b} (i.e., top 1000 sources), and (iv) *vAS* assigns mitigation tasks only to *AS2* based on: r_{ij}^{a}. The last approach is a random assignment, since all rewards r_{ij} are equal; we illustrate the worst case scenario in which the assigned 1000 malicious sources have the least contribution to the attack.

For approaches (i) and (ii), we assumed that *AS2*, *AS3*, and *AS4* have a reputation equal to 0.9, 0.6, and 0.5, respectively. For (iii) and (iv), the reputation score of *AS2* is assumed to be 1, thus blocks every flow assigned to it. The attack is detected, and countermeasures are deployed after 30 seconds. This accounts for solving the GAP and applying the mitigation actions to the XDP-enabled mitigation box.

In Figure 6, we illustrate the amount of traffic that is delivered to the victim. The attack traffic ranges between 2.5 and 3 Gbps. When multiple ASes collaborate, that is, approaches (i) and (ii), the assignment of mitigation actions based on r_{ij}^{a} is significantly inferior to r_{ij}^{b} that incorporates flow importance. As shown in the figure, approach (i) results to twice the volume of the malicious traffic reaching the victim, as opposed to (ii). Similarly, this is also illustrated in approaches (iii) and (iv), whereby only *AS2* is contributing to the attack mitigation. Assigning mitigation tasks in a random fashion as in approach (iv) might lead to an unsuccessful mitigation, while considering flow importance in approach (iii) mitigates a larger volume of malicious traffic. Note that, both approaches (iii) and (ii) perform comparably well in terms of the total attack traffic reaching the victim. However, approach (iii) assumes that there exists a

Table 2 Percentage of malicious sources observed under varying sampling rates and mitigation performance

	Successful mitigation		Unsuccessful mitigation	
Experiment parameters	**Set**	**Bloom filter**	**Set (%)**	**Bloom filter (%)**
mAS – sampling 1/100 packets	0 sources	0 sources	88.86	88.9
vAS—sampling 1/100 packets	0 sources	0 sources	88.62	88.74
mAS—sampling 1/1000 packets	0 sources	0 sources	78.02	78.1
vAS—sampling 1/1000 packets	0 sources	0 sources	76.8	76.88

top notch strategically placed collaborator that is able to consistently block all malicious sources and has adequate capacity to sustain massive DDoS attacks. On the contrary approach, (ii) relies on collaborative efforts and smart distribution of mitigation actions across multi-domain attack paths. Thus, we consider approach (ii) as the most appropriate for our collaborative schema.

5.4 Mitigation verification

We conducted a series of experiments to evaluate the mitigation verification process, detailed in Section 4.6. This relies on malicious flows as monitored by SC participants, both *vAS* and *mASes*. We used NetFlow data exported by separate flow exporters within the multilayer switch, sampled at two different rates: (i) 1 out of 100 and (ii) 1 out of 1000. NetFlow data are sent to a collector machine running two distinct processes of the *nfdump* toolset (Netflow Processing Tools – nfdump, 2018).

In Table 2, we represent the percentage of identified sources out of the agreed 1000, for successful and unsuccessful mitigation considering different sampling rates. The exact number for which assistance has been requested is 1000 (the top 1000 hitters in Booter-1 dataset).

As expected, none of the 1000 sources are identified when mitigation is performed successfully, since the Bloom Filters, as described in Section 4.6, only maintain the agreed upon malicious sources which in this case do not reach the victim. In the unsuccessful mitigation scenario, we expect most of the sources that were not mitigated to be hashed in the *vAS* Bloom Filters. However, packet sampling limits our *vAS* identification rates for non-mitigated malicious sources to 88.62% for a sampling rate of 1/100 and 76.8% for a sampling rate of 1/1000. In massive DDoS attacks though, we expect the identification of flows at the *vAS* to be closer to 100%, since the probability of malicious traffic being selected in the sampling process would be significantly higher. Note that the small discrepancies between the *vAS* and *mAS* identification rates are due to hardware limitations in our testbed (e.g., flow processing performance and cache size).

Apart from the sampling process, Bloom Filters also affect the accuracy of our identification process due to their inherent limitation of producing some false positives (Bloom, 1970). Thus, a flow might be identified as present (not blocked), although it has been successfully blocked. Note that false positive probabilities in Bloom Filters can be reduced to acceptable levels with very modest space consumption. Thus, we selected this option to fulfill our major concern, that is, the verification of flows blocked by collaborating ASes. Specifically in our experiments for 1000 distinct sources, we opted for a fairly small false positive of 1% that according to Broder and Mitzenmacher (2004) is attainable with a Bloom Filter of 1.17 Kbytes and seven hash functions.

6. Conclusion: future work

In this paper, we provide an integrated framework based on DLTs for Wholesale Network Providers, enabling them to collaboratively mitigate massive DDoS attacks. We extended our preliminary work in Giotis *et al.* (2018) by introducing network programmability tools, efficiently distributing defense resources across attack paths and formalizing digital agreements as SCs amongst collaborating domains. Our schema is geared toward coordination and orchestration that ultimately facilitates collaboration

and accountability for incident handling within a reputation-based, reward-driven federation of network providers.

Our proof-of-concept implementation is based on the Ethereum blockchain network; appropriately deployed SCs are used to propagate reported security incidents and streamline the collaborative mitigation of malicious flows. Defense resources (i.e., mitigation actions) are appropriately assigned on collaborators considering available mitigation offerings; this has been formulated as a GAP. Offered security services are ranked by reputation scores factoring in the actual contribution of collaborators. The filtering of malicious traffic is implemented on XDP leveraging on the capabilities of programmable data planes. SCs pertaining to the actual mitigation process can be managed and verified via automated mechanisms.

As future work, we plan to investigate advanced capabilities promised by next-generation networks in order to efficiently offload monitoring tasks on programmable data planes and increase accountability. Specifically, we aim to create special purpose XDP-based appliances that extract monitoring data pertaining to network anomalies and also mitigate these anomalies based on distinct traffic characteristics. These portable appliances could be seamlessly deployed by various ASes within the attack path, following the same mechanisms described in this paper. Another important area is to extensively measure performance metrics such as packet throughput and deployment times. We further aim to incorporate detection mechanisms based on emerging advances in Federated Learning (Konečný *et al.*, 2016); this may empower collaborators to collectively detect complex network anomalies while preserving client privacy. Finally, we plan to look into sophisticated arbitration mechanisms to settle disputes amongst Wholesale Network Providers, tokenizing shared assets, charging for security services, and settling multilateral agreements thus forming new business opportunities.

Acknowledgements

This work was partially supported by the European Commission Horizon 2020 Framework Programme for Research and Innovation, Grant Agreement No. 856726 (GN4-3). Line of research and development works within PCCW Global, the international operating division of Hong Kong Telecom (https://www.pccwglobal.com/en/about).

References

3DCoP: DDoS Defense for a Community of Peers. 2016. available at: https://galois.com/project/3dcop-ddos-defense/

Back, A., Matt, C., Luke, D., Mark, F., Gregory, M., Andrew, M., Andrew, P., Jorge, T. & Pieter, W. 2014. "Enabling blockchain innovations with pegged sidechains", available at: http://www.opensciencereview.com/papers/123/enablingblockchain-innovations-with-pegged-sidechains

Bertin, G. 2017. "XDP in practice: Integrating XDP into our DDoS Mitigation Pipeline", https://netdevconf.org/org/2.1/papers/Gilberto_Bertin_XDP_in_practice.pdf

Bloom, B.H. 1970. "Space/Time Trade-offs in Hash-Coding with Allowable Errors", in *Communications of the ACM* **13**(7), 422–426.

Broder, A. & Mitzenmacher, M. 2004. "Network Applications of Bloom Filters: A Survey", *Internet Mathematics* **1**(4), 485–509.

Buterin, V. 2015. "On Public and Private Blockchains", available at: https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/

Claise, B., Ed., 2004. "Cisco Systems NetFlow Services Export Version 9", October.

ConsenSys – Harness the power of Ethereum. 2014. available at: https://new.consensys.net/

Decentralized CDN, WAF, and DDoS protection. 2018. available at: https://gladius.io

Dimolianis, M., Pavlidis, A., Kalogeras, D. & Maglaris, V. 2019. "Mitigation of Multi-vector Network Attacks via Orchestration of Distributed Rule Placement", in proc. of the IFIP/IEEE International Symposium on Integrated Network Management (IM 2019), Washington D.C., USA, pp. 162–170, April.

Ethereum Network Intelligence API. 2016. available at: https://github.com/cubedro/eth-net-intelligence-api

Ethereum Network Stats. 2016. available at: https://github.com/cubedro/eth-netstats

Ethereum Project. 2015. available at: https://github.com/ethereum/

Giotis, K., Androulidakis, G. & Maglaris, V. 2015. "A Scalable Anomaly Detection and Mitigation Architecture for Legacy Networks via an OpenFlow Middlebox", in *Security and Communication Networks*, pp. 1958–1970.

Giotis, K., Apostolaki, M. & Maglaris, V. 2016. "A Reputation-based Collaborative Schema for the Mitigation of Distributed Attacks in SDN domains", in proc. of the IEEE/IFIP Network Operations and Management Symposium, pp. 495–501, April.

Giotis, K., Pavlidis, A., Anagnostou, L., Dimolianis, M., Tsigkritis, T., Kalogeras, D., Kostopoulos, N., Kotinas, I. & Maglaris, V. 2018. "Blockchain-based Federation of Network Providers for Collaborative DDoS Mitigation", 3rd Symposium on Distributed Ledger Technology, Gold Coast, Australia, November.

Go Ethereum. 2019. available at: https://github.com/ethereum/go-ethereum

Gruhler, A., Rodrigues, B. & Stiller, B. 2019. "A Reputation Scheme for a Blockchain-based Network Cooperative Defense" in proc. of the IFIP/IEEE International Symposium on Integrated Network Management (IM 2019), Washington D.C., USA, pp. 71–79, April.

Høiland-jørgensen, T., Borkmann, D., Fastabend, J., Herbert, T., Ahern, D. & Miller, D. 2018. "The eXpress Data Path: Fast Programmable Packet Processing in the Operating System Kernel", in proc. of the 14th ACM International Conference on emerging Network Experiments and Technologies (CoNEXT '18), pp. 54–66, December.

InterPlanetary File System (IPFS). 2015. available at: https://ipfs.io/

Josang, A. & Ismail, R. 2002. "The Beta Reputation System", in proc. of the 15th Bled Electronic Commerce Conference. 5, 2502–2511, June.

Kim, K., You, Y., Park, M. & Lee, K. 2018. "DDoS Mitigation: Decentralized CDN Using Private Blockchain" in Tenth International Conference on Ubiquitous and Future Networks (ICUFN), July.

Konečný, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T. & Bacon, D., 2016. "Federated Learning: Strategies for Improving Communication Efficiency", available at: https://arxiv.org/pdf/1610.05492.

Malomo, O. O., Rawat, D. & Garuba, M. 2018. "Next-generation cybersecurity through a blockchain-enabled federated cloud framework", *The Journal of Supercomputing* 1–28, May.

Mannhart, S., Rodrigues, B., Scheid, E., Kanhere, S. S., & Stiller, B. 2018. "Toward Mitigation-as-a-Service in Cooperative Network Defenses," in 2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th International Conference on Pervasive Intelligence and Computing, 4th International Conference on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), pp. 362–367, August.

Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J. & McPherson, D. 2009. "Dissemination of Flow Specification Rules", RFC 5575, available at: http://www.ietf.org/rfc/rfc5575.txt

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shrenker, S. and Turner, J. 2008. "OpenFlow: enabling Innovation in Campus Networks", in *ACM SIGCOMM Computer Communication Review* **38**(2), 69–74.

Memcached DDoS Attacks: 95,000 Servers Vulnerable to Abuse. 2018. available at: https://www.bankinfosecurity.com/memcached-ddos-attacks-95000-servers-vulnerable-to-abuse-a-10705

Mortensen, A., Andreasen, F., Reddy, T., Gray, C., Compton, R. & Teague, N. 2019. "DDoS Open Threat Signaling (dots)", available at: https://datatracker.ietf.org/wg/dots/

Mutually Agreed Norms for Routing Security. 2016. available at: https://www.manrs.org/

Netflow Processing Tools – nfdump. 2018. https://github.com/phaag/nfdump

O'Sullivan, M., Lim, Q. S., Walker, C., Dunning, I. & Mitchell, S. 2011. "Dippy: A Simplified Interface for Advanced Mixed-integer Programming", Report 685, University of Auckland Faculty of Engineering.

Phaal, P. & Lavine, M. 2004. "sFlow Version 5", available at: https://sflow.org/sflow_version_5.txt

Proof-of-Authority Chains. 2017. available at: https://wiki.parity.io/Proof-of-Authority-Chains

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.

Rodrigues, B., Bocek, T., Lareida, A., Hausheer, D., Rafati, S. & Stiller, B. 2017. "A Blockchain-Based Architecture for Collaborative DDoS Mitigation with Smart Contracts", in IFIP International Conference on Autonomous Infrastructure, Management and Security, pp. 16–29, June.

Santanna, J. J., van Rijswijk-Deij, R., Hofstede, R., Sperotto, A., Wierbosch, M., Granville, L. Z, & Pras, A. 2015. "Booters—An Analysis of DDoS-as-a-Service Attacks", Integrated Network Management (IM), in proc. of the 2015 IFIP/IEEE International Symposium, pp. 243–251.

Solidity Programming Language. 2019. available at: https://github.com/ethereum/solidity

The CAIDA UCSD Anonymized Internet Traces 2016. available at: http://www.caida.org/data/passive/passive_2016_dataset.xml

The Incident Object Description Exchange Format 2007. https://tools.ietf.org/html/rfc5070

Van Rijswijk-Deij, R., Rijnders, G., Bomhoff, M. & Allodi, L. 2019. "Privacy-Conscious Threat Intelligence Using DNSBLOOM", in proc. of the IFIP/IEEE International Symposium on Integrated Network Management (IM 2019), Washington D.C., USA, pp. 98–106, April.