Pay the Piper: DDoS mitigation technique to deter financially-motivated attackers

Anonymous Author(s)

ABSTRACT

Distributed Denial of Service attacks against the application layer (L7 DDoS) are among the most difficult attacks to defend against because they mimic normal user behavior. Some mitigation techniques against L7 DDoS, e.g., IP blacklisting and load balancing using a content delivery network, have been proposed; however, unfortunately, these are symptomatic treatments rather than fundamental solutions. In this paper, we propose a novel technique to disincentivize attackers from launching a DDoS attack by increasing attack costs. Assuming financially motivated attackers seeking to gain profit via DDoS attacks, their primary goal is to maximize revenue. On the basis of this assumption, we also propose a mitigation solution that requires mining cryptocurrencies to access servers. To perform a DDoS attack, attackers must mine cryptocurrency as a proof-of-work (PoW), and the victims then obtain a solution to the PoW. Thus, relative to attackers, the attack cost increases, and, in terms of victims, the economic damage is compensated by the value of the mined coins. On the basis of this model, we evaluate attacker strategies in a game theory manner and demonstrate that the proposed solution yields negative economic impact to attackers. Moreover, we implement a prototype to evaluate performance, and we show that this prototype demonstrates practical performance.

CCS CONCEPTS

• **Security and privacy → Denial-of-service attacks**; **Denial-of-service attacks**; **Economics of security and privacy**;

1 INTRODUCTION

Distributed Denial of Service (DDoS) attacks are a serious but unsolved problem. Broadly, there are two main types of DDoS attacks, i.e., (1) saturation attacks against communication bandwidth and (2) resource exhaustion attacks against servers. Relative to the bandwidth saturation attack, a recent attack recorded 1.7 Tbps [22]. For attacks against servers, the largest attack was greater than two million HTTPS requests per second [28]. In this paper, we focus on DDoS attacks against servers.

DDoS attacks have been commoditized, and many DoS tools can easily be found on the web [27] and in underground markets. In this sense, malicious organizations, small groups, and individuals can provide DDoS-as-a-Service [17] as their business model. Using such tools and services, people, including non-experts, can easily perform DoS/DDoS attacks. In other words, attack costs, e.g., computer and network resources, and the required expertise have become low in recent years.

Regarding countermeasures against DDoS attacks, many mitigation methods have been proposed. Recently, Content Delivery Network (CDN)-based defenses have become common. A CDN is a load-balancing system that uses geographically distributed cache servers for efficient distribution of content. CDNs mitigate DDoS attacks by absorbing the large number of layer 7 requests (e.g., HTTP GET flood and HTTP PUT flood) at the network edge to reduce traffic to an origin server. Moreover, a CDN provides a Web Application Firewall that monitors HTTP protocols and blocks invalid requests. Similar mitigations against DDoS are provided by Internet service providers (ISP) and network security companies. Another mitigation technique is SYN cookies, which have been deployed to mitigate SYN flooding attacks. SYN flooding is a layer 4 DoS attack that consumes server resources due to the large number of created connections. A SYN cookie encodes a client IP address into a TCP sequence number of a SYN/ACK packet (second packet of a TCP handshake), and a server checks the consistency of a SYN packet (third packet of a TCP handshake). However, this does not work when attackers correctly perform a three-way handshake. As a simple network defense against layer 4 and 7 attacks, ISPs and telecom carriers enforce an IP black hole that simply drops packets to specific IP addresses targeted by the DDoS attack; however, this solution also drops legitimate traffic.

There are many other mitigation strategies [32], but DDoS attacks remain an unsolved problem because, essentially, the requests for DDoS and requests for legitimate access cannot be distinguished according to request metadata, such as IP addresses, or by the content of the requests. For example, considering HTTP-flooding attacks, attackers can easily send legitimately formed requests to victim servers, and the difference between DDoS and legitimate requests is only the volume of traffic. Moreover, the request-per-host is low in DDoS attacks from a large botnet; thus, per-host traffic shaping is ineffective.

The problem with existing solutions is that they attempt to mitigate damage caused by the DDoS attack and do not consider attacker motivations. Specifically, existing solutions, such as filtering/shaping, are passive and do not reduce attackers' incentives. Therefore, attackers can perform DDoS attacks without any drawbacks because the countermeasures are not effective deterrents. Moreover, it is difficult to mitigate DDoS using only technological solutions because differentiating legitimate and DDoS requests is difficult.

To address this problem, we introduce an economic solution that reduces the incentives of financially motivated DDoS attackers. In this paper, we focus on attackers whose ultimate goal is to gain profit rather than cause damage. From an attacker perspective, if attack costs are greater than the attack profit, the attackers have little or no motivation to launch an attack. Instead, an attacker can use their infrastructure (e.g., a botnet) to obtain revenue via other methods, such as joining a pool to mine cryptocurrencies. From victim's perspective, if losses suffered from an attack are compensated, victims are not motivated to consider such attacks as a serious threat. On the basis of this assumption, our goal is to

Anonymous Submission to XXXX, ,
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

reduce DDoS attackers' incentives. Specifically, we aim to maximize attack costs while simultaneously minimizing victim losses.

To this end, we propose an architecture that requires clients accessing a server to mine cryptocurrency coins. In the proposed architecture, a gateway is deployed between clients and servers. This gateway issues cryptocurrency mining requests to clients seeking access to the servers. Then, only clients who perform mining are permitted to access the servers. In terms of an economic model, this gateway increases the attack costs because attackers must invest more resources to mine coins. Moreover, the victims can obtain profit while their services are under attack, which means the attacker's intent (e.g., economic profit by ransom) is not realized. Thus, DDoS attackers would face a dilemma, i.e., perform a DDoS attack at the cost of mining coins as a proof-of-work (PoW), thereby providing victims with coins via the PoW solution. This dilemma is considered an effective deterrent against DDoS attacks.

We evaluate this architecture and conclude that there is no economic benefit when targeting small and medium businesses. The reason for this is that victims are not required to pay ransom money to avoid a DDoS attack because the damage from the DDoS is compensated by the obtained cryptocurrency income. Moreover, in some cases, directly mining coins using a botnet is more profitable than a DDoS attack using the same botnet. We implement a gateway prototype and evaluate its performance. The results demonstrate that the prototype's performance is practical. We believe that the proposed model and architecture provides a new perspective relative to mitigating DDoS attacks.

Our primary contributions are summarized as follows.

- We present an economic model that reduces the incentives of DDoS attackers by requiring them to mine cryptocurrency coins when accessing a server. The effectiveness of this economic model is evaluated using game theory.
- We propose an architecture that reduces the motivation of DDoS attackers based on the economic model.
- The implementation of an initial prototype acting as an HTTP reverse proxy does not require any modification to existing HTTP servers.

The remainder of this paper is organized as follows. In Section 2, we define the problem. Then, we propose the economic model and architecture using the economic model in Sections 4 and 5, respectively. In Section 6, we evaluate the performance of the proposed architecture. We discuss a security analysis of the proposed architecture and discuss its limitations in Sections 7 and 8, respectively. The paper is concluded in Section 10.

2 PROBLEM DEFINITION

Our primary goal is mitigation of DDoS attacks by reducing the financial incentives of DDoS attackers, which can be achieved by maximizing attack costs and minimizing damages to victims.

2.1 DDoS economic setting

In recent years, the cost of a DDoS attack has reduced dramatically due to improved attack techniques. Specifically, DoS attack tools [27] are available, and such tools can be used by both experts and non-experts. With these tools, attackers only need to specify the target IP address and push a start button to perform a DoS attack. Moreover, malicious organizations, small groups, and individuals provide extremely low-cost DDoS-as-a-Service [17]. For example, the cheapest plan can be acquired for five euros per month [20].

Furthermore, attackers frequently use large botnets to perform DDoS attacks. For example, the Mirai botnet [5] can perform DDoS attacks using IoT devices that have been exploited extensively. Due to the large size of this botnet, its attacks have disrupted the operations of the target and caused major collateral damages due to the unavailability of the targets' services. However, the costs to create such a botnet are small because it compromised devices using default credentials widely available on the Internet.

We can use some techniques to increase attack costs. One technique is a PoW-based defense [10], which requires solving a computational puzzle to access a server. In this case, attackers must have very powerful machines or a large number of attack hosts to achieve the required high request speed. However, considering recent large botnets, this method would be insufficient to make attackers stop DDoS attacks.

The above techniques introduce additional attack costs; however, the conditions that would make DDoS attacks unviable remain unclear. Moreover, our solution requires cryptocurrency mining as a PoW (Section 5); thus, an attacker loses opportunities to obtain profit from mining. This scheme provides a negative impact to an attacker, and the conditions become strict because of the negative impact. Therefore, our first goal is *identification of conditions under which DDoS attacks have no benefit to attackers.*

2.2 Minimizing economic damage of DDoS victims

We can identify some damage mitigation solutions; however, these solutions are insufficient. For example, IP blacklisting blocks requests from specified IP source addresses. Unfortunately, in case of L7 DDoS attacks, legitimate and DDoS requests have the same form and cannot be distinguished; thus, a DDoS victim cannot add the IP addresses of the DDoS attackers to a blacklist. IP whitelisting also does not work when considering general services where the IP addresses of legitimate users cannot be specified in advance. In addition, deep packet inspection does not work because attackers simply send legitimately formed requests to victims, and the legitimate and DDoS attack requests cannot be differentiated effectively by inspecting their contents.

As described above, existing technical solutions are limited relative to mitigating DDoS attacks. Therefore, our second goal is *mitigation of DDoS attack damage beyond what is achievable with existing solutions.*

3 BACKGROUND

The proposed architecture leverages a cryptocurrency to create a disincentive for the attacker to a launch a DDoS attack. In this section, we briefly introduce how cryptocurrencies work. We also introduce some basic terminology related to game theory, which we use later to examine the effectiveness of the proposed economic model and architecture.

3.1 Cryptocurrency

Here, we explain the basics of Bitcoin [24] as an example cryptocurrency. Bitcoin leverages a blockchain to record transactions between accounts, such as sending and receiving money. A blockchain is a set of entries associated by hash values. Specifically, a block in a blockchain contains a hash value of its previous block. Therefore, modification of a block can be detected by verifying its hash value and the hash value stored in the next block. As a result, modifying a block requires an update to the hash values stored in all subsequent blocks, and this mechanism makes malicious falsification difficult.

Any new block added to the chain must satisfy the following equation.

$$\text{hash}(\text{nonce} + \text{newblock}) < \text{difficulty}. \quad (1)$$

Due to the hash function, there is no efficient algorithm to satisfy this equation; thus, the client must perform a brute force scan. This task is referred to as a PoW, which ensures that attackers cannot falsify the blockchain if their computational power is less than that of benign users. Specifically, if a blockchain has multiple branches, the longest branch is considered the valid branch. Thus, if the attacker has greater than 51% of the entire computational power, they can create a fake longest branch that is considered the valid branch. The PoW is also called *mining* because a client that finds a solution to the PoW (nonce) and adds a new block obtains coins as a reward. Note that clients that perform mining are referred to as miners.

Bitcoin transactions are recorded by the blocks, where each block contains transactions, and the integrity of the transactions is ensured by the blockchain.

3.2 Mining pool

In the Bitcoin case, the difficulty of mining is adjusted such that a new block is added every 10 min, which means that a miner has little chance of obtaining coins when many miners are online. A mining pool has been developed to share the chance. The mining pool separates the search space of the nonce and distributes search spaces to clients in the pool. The mining pool has its own difficulty, which is less than the original mining difficulty. When a client hits a coin, the coin is shared with the clients in the mining pool. Here the mined coin is shared on the basis of the computational power of each client. The computational power of a client is measured by the number of PoW solutions the given client sends to the pool.

3.3 DDoS attacks and game theory

Game theory is an analysis framework to analyze decision-making among players. A basic analysis method includes defined player strategies, and then the payoff of each strategy is evaluated. Some studies have analyzed the actions of an attacker and a defender in a DDoS from a game theory perspective. Spyridopoulos et al. proposed an attack and defense strategy when using a firewall rate limit [34], and Narasimhan et al. applied game theory to puzzle-based DDoS mitigation [25].

Different from DDoS attack strategies discussed in the above study, we focus on high-level interaction among players where DDoS attacks will only be launched if they are profitable financially. Specifically, the differences in the game configuration are summarized as follows.

- Existing studies have focused on fine-grained strategies to maximize DDoS efficiency after an attacker decides to perform an attack. In contrast, we focus on a high-level decision-making process to determine whether an attacker should (or should not) perform a DDoS attack.
- Existing studies assume a game with infinite rounds; however, this is unrealistic because an attacker moves to other victims rather than staying focused on a specific attack target. Thus, we adopt a two-round game.
- The existing studies assume a zero-sum game where the attacker's profit is equal to the victim's damage. However, this is unrealistic relative to the actual flow of money between the attacker and victim. Thus, we assume DDoS for ransom money, which has actually been performed by malicious organizations and individuals.

4 ECONOMIC MODEL

We propose an economic model to reduce the attacker incentive by increasing their costs. Here, we focus on attackers that are motivated by financial gain. Other attack motivations, e.g., hacktivists and pranksters, are beyond the scope of this paper. Specifically, we assume attackers that demand ransom money.

In PoW-based mitigation, a computational puzzle must be solved to gain access to servers. In the proposed model, mining a cryptocurrency is the required computational puzzle. Differing from the computational puzzle, the victims of a DDoS attack can obtain profit from the mining; thus, the victim's damage (opportunity loss) is reduced by this profit. The attackers then face a dilemma. Specifically, attackers seek to gain profit through DDoS attacks; however, the attackers must mine cryptocurrencies coins as a PoW, and victims receive the coins when the PoW is solved. Thus, in the proposed architecture, the economic motivation of the attackers is reduced. In the following, we discuss the economic model of the proposed architecture.

We can identify two cases where an attacker gains profit through DDoS attacks. The first case is the zero-sum game, where the attacker's profit is equal to the damage done to the victim. For example, the victim's service is down because of a DDoS attack, and the victim's customers buy products from the attacker's service rather than the victim's service. The second case is ransom money. Here, the attacker demands ransom money to gain profit.

The zero-sum game case would not be a realistic situation because it cannot be ensured that the customers of the victim's service move to the attacker's service when considering similar services from third parties. In contrast, real-world DDoS ransom cases have been reported (Section 9.1). Thus, we focus on the ransom money case and examine attacker and defender strategies based on payoff from a game theory perspective.

4.1 Money flow analysis

Here, we analyze how money flows between an attacker, a victim, and legitimate users (Figure 1). Between the attacker and the victim, DDoS damage is compensated by the mined coin, and, between

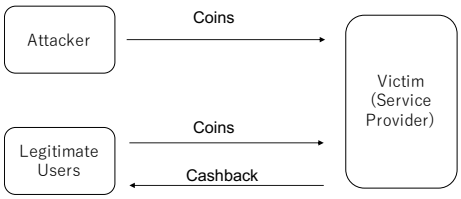

Figure 1: Money flow between attacker, victim, and legitimate users

Table 1: Notation

Symbol	Definition
C_d	Cost of defense mechanism
C_a	Attack cost
C_m	Mining cost
P_m	Mined coins
V_v	Value of DDoS victim's service
α	A parameter of ransom money

legitimate users and the victim, the legitimate users must also mine coins and give the coins to the victim to access the victim's service. To be fair to the legitimate users, revenue generated by the coin should be returned to the user. In the case of a service with authentication, the victim identifies the users paying the coins using authentication information and returns a profit of the coin from the users, e.g., by issuing a special discount ticket. Thus, in terms of money flow, only the DDoS attacker receives a negative impact by our mechanism. In addition, the DDoS victim receives a positive impact because the victim obtains coins mined by the DDoS attacker. In case of services without authentication, legitimate users paying the coins cannot be identified; thus, for transparency, the victim should clearly indicate that the PoW solution has been applied to DDoS defense.

4.2 Attacker model

We assume that DDoS attackers send legitimately formed requests and that legitimate users and malicious attackers cannot be distinguished. We further assume powerful attackers that can employ a large botnet to perform DDoS attacks.

In addition, we assume rational attackers performing DDoS attacks for financial gain. Specifically, the goal is the maximization of profit gained by DDoS attacks. In case that no profit is gained by the DDoS attacks, attackers do not perform attacks. Moreover, we assume a two-round decision process where the attacker initially decides to demand ransom money and decides to perform a DDoS attack. The reason for the two-round decision process is that most attackers are expected to change the attack target if a DDoS attack is not worth the cost. The attacker strategy is described in Section 4.4.

4.3 Economic model of DDoS ransom

We define the model wherein the attacker demands $\alpha \times V_v$ as ransom money. Here, V_v denotes the value of the victim's service.

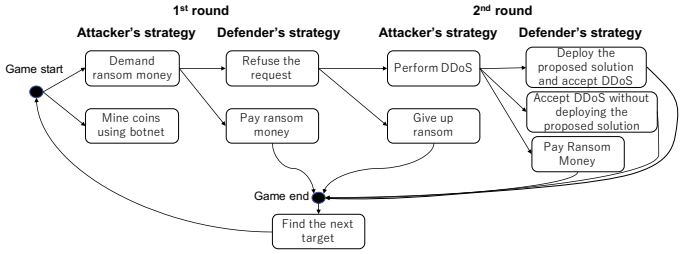

Figure 2: Two-round game tree

Specifically, the value of the victim's service is the profit the service is expected to earn during a DDoS attack. Furthermore, parameter α deals with some types of attacks. For example, in a zero-sum game case, $\alpha = 1$, which means that the attacker's profit is equal to the victim's economic damage.

Here, we consider the model with an example ransom DDoS. Specifically, we assume that an attacker demands $\alpha \times V_v$ as ransom money from the victim. Note that the range of α is $[C_a/V_v, 1]$ because the attacker's profit becomes negative where $\alpha \leq C_a/V_v$. In addition, the victim is not motivated to pay the ransom if $\alpha \geq 1$, which means that the ransom money is greater value than service.

4.4 Strategies and payoffs for attackers and defenders

Here, we investigate attacker and defender strategies using game theory based on the proposed economic model. We define the strategies and payoffs of the attacker and defender based on the proposed economic model. Here, we assume a two-round game where (1) the attacker determines whether to demand ransom money, (2) the defender determines whether to accept the ransom demand, (3) the attacker then determines whether to perform a DDoS attack, and finally, (4) the defender determines whether to pay the ransom (Figure 2). The payoff of each strategy is described in the following.

(1) First round: Attacker strategy. First, the attacker can adopt one of two strategies, i.e., to demand a ransom (or not).

When the attacker decides to not perform a DDoS attack, the attacker can mine coins using their botnet. The attacker's payoff is calculated from the profit of mined coin P_m minus mining cost C_m. Here, the defender's payoff is zero because there is no money flowing between the attacker and defender.

$$\text{Attacker's payoff: } P_m - C_m \tag{2}$$

$$\text{Defender's payoff: } 0 \tag{3}$$

When the attacker decides to demand ransom money, the game moves to the defender's turn.

(2) First round: Defender strategy. The defender can adopt one of two strategies when the attacker demands ransom money.

- Pay the ransom money. When the defender adopts this strategy, the game ends. Here, the payoffs of the attacker and defender are equal to the ransom money.

$$\text{Attacker's payoff: } \alpha \times V_v \tag{4}$$

$$\text{Defender's payoff: } -\alpha \times V_v \tag{5}$$

- Refuse the ransom demand. When the defender adopts this strategy, the game turns to the attacker (second round).

(3) Second round: Attacker strategy. When the defender refuses the ransom demand, the attacker either abandons the ransom DDoS attack or performs the DDoS attack.

- Abandon DDoS. The game ends when the attacker adopts this strategy. The payoff of both players is zero because there is no flow of money. The attacker finds another target and begins a new game.
- Perform DDoS attack. When the attacker performs a DDoS attack, the game proceeds to the defender's final turn.

(4) Second round: Defender strategy. The defender must decide whether to pay the ransom (or not). For each strategy, the payoffs of the attacker and defender are determined as follows.

- Deploying proposed solution and refusing ransom demand. To perform a DDoS attack, attacker must solve the PoW. Thus, in addition to attack cost C_a, the attacker must pay mining cost C_m. Moreover, the attacker must give the mined coin P_m to the defender. Relative to the defender, the defender gains the profit of mined coins; however, they must pay defense cost C_d and suffer damages caused by the downtime of the defender's service V_v. Therefore, the payoff of the attacker and defender are given as follows.

$$\text{Attacker's payoff: } -C_a - P_m - C_m \tag{6}$$

$$\text{Defender's payoff: } P_m - V_v - C_d \tag{7}$$

- Not deploying proposed solution and paying ransom. The payoffs of the attacker and defender are the same as the case where the defender pays the ransom money (Equations 4 and 5).
- Not deploying PoW solution and not paying ransom money. Here, the payoff of the attacker is the attack cost, and the payoff of the defender is damage caused by service downtime.

$$\text{Attacker's payoff: } -C_a \tag{8}$$

$$\text{Defender's payoff: } -V_v \tag{9}$$

This strategy does not provide advantages to either the attacker or defender; thus, we expect that this strategy will not be adopted when the attacker and defender are rational. This case corresponds to DDoS attacks that are not financially motivated, such as harassment and hacktivism activities.

4.5 DDoS attacking game conditions

On the basis of the above strategies and payoffs, here, we analyze the conditions whereby a DDoS attack is financially and rationally feasible. In Figure 3 , we have represented the different conditions as a function of the cost and benefit of the attack. In this case, attacks are only profitable in the yellow area, which does not satisfy the following conditions.

Condition 1: Profit of direct mining (Equation 2) > ransom money (Equation 4). Here, the attacker is not motivated if the income from direct mining is greater than the ransom money. In this case, mining is a better business decision than demanding ransom.

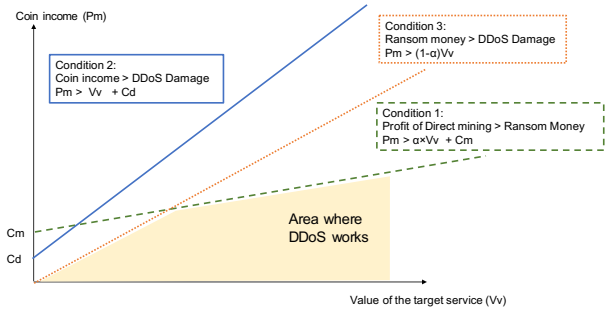

Figure 3: Conditions whereby DDoS attacks work

This condition is expressed as follows.

$$P_m > \alpha \times V_v + C_m \tag{10}$$

Condition 2: Coin income of a defender > DDoS damage (Equation 7 > 0). Here, the compensated income by the mined cryptocurrency is greater than the damage caused by the DDoS attack. Thus, the defender is not motivated to pay the ransom because there is no financial damage. This condition is expressed as follows.

$$P_m > V_v + C_d \tag{11}$$

Condition 3: Ransom money (Equation 5) > DDoS damage (Equation 7). Here, if the ransom money is greater than the cost of the DDoS damage, the defender is not motivated to pay the ransom because accepting the attack is economically rational. This condition is expressed as follows.

$$P_m > (1-\alpha)V_v \tag{12}$$

By combining the above conditions, a DDoS attack that is motivated by financial gain only makes sense in the yellow area in Figure 3. In Section 6.3.1, we examine the conditions using the parameters of actual DDoS attack cases.

4.6 Is the proposed solution beneficial for the defender?

It is expected that defenders will deploy the proposed solution if the damage caused by the DDoS attack with the solution is less than the damage without the solution. Specifically, the condition is Equation 9 > Equation 7.

$$P_m > C_d \tag{13}$$

In Section 6.3.1, we demonstrate that the proposed solution satisfies this condition using parameters used in our prototype performance and actual DDoS attack cases.

4.7 Attacker strategy to maximize profit

From the above analysis, the attacker strategy to maximize profit is described as follows.

- Targeting large companies that could pay high-value ransom money. Indiscriminate attacks against small/medium companies would not be an effective strategy because such companies can compensate opportunity loss using income from mined coins.

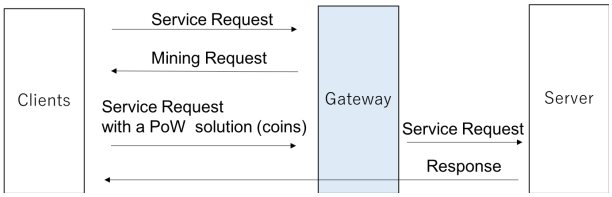

Figure 4: Architecture and communication protocol

- Targeting companies whose services are essential for citizens (e.g., banks and government-related services). These companies would accept a high α value; however, such companies and organizations would not pay ransom money because not giving in to threats is a fundamental crime-prevention policy.

5 ARCHITECTURE

5.1 Requirements

Prior to designing the architecture for the proposed model, we defined a set of requirements for a practical solution. Specifically, the solution should have the following features to minimize deployment cost.

- No modification of existing servers. There are currently millions of servers across the Internet; thus, to protect as many servers as possible, the solution should not require server modification or replacement.
- No modification of existing communication protocols. It is impractical to modify and extend existing communication protocols; thus, the solution should function effectively using existing communication protocols.

5.2 Overview

The proposed architecture comprises a server providing a service, clients accessing the server, and a gateway deployed between the server and the client (Figure 4). The server provides common services, e.g., HTTP service and SMTP services. Here, we assume some clients are bots controlled by DDoS attackers.

The gateway is a core component of the proposed architecture that mitigates DDoS attacks by requesting cryptocurrency coins when the server is subject to a DDoS attack. The gateway works as follows. Under normal circumstances, i.e., the load of a web server is under a certain threshold, the gateway passes requests to the web server. If the server's load is greater than the given threshold, the gateway hooks all requests from a client that could be legitimate or malicious. Then, the gateway returns a mining request to the client. If the gateway receives a request with a valid PoW solution, it passes the request to the web server. If the gateway receives a request with an invalid PoW solution, it drops the request.

5.3 Message format

Here, we describe the message format of the mining request issued by the gateway. The gateway issues mining requests that include the following information.

Mining request issued by the gateway

- The client ID (IP address) is encoded into the message and used to bind a mining request to a client.
- A mining request expiration time is implemented. The client should submit the solution to the gateway before the request expires. If the request expires, the client retrieves the mining request again from the gateway and mines using a new nonce search space specified by the reissued mining request.
- The mining parameters.
  - A nonce search space that specifies the upper bound and lower bound of the nonce.
  - The mining difficulty. The client must find a nonce value that meets Equation 1.
  - The hash value of the previous block, which is used to construct a blockchain by associating a new block with the previous block.
- The Message Authentication Code (MAC) is computed over the above fields using a gateway key.

Clients receive the mining request and extract the mining parameters. Then, the clients begin mining based on the given parameters. Once a client finds a PoW solution, i.e., a valid nonce value that satisfies the specified difficulty, the client accesses the server via the gateway using the following request format.

- Access request (e.g., HTTP GET message).
- A nonce (PoW solution) that satisfies the difficulty specified by the mining request.
- The mining request issued by the server.

The gateway verifies the nonce using the parameters in the mining request. If the nonce is valid, the gateway mediates the access request and the response from the server.

5.4 Gateway workflow

Here, we explain the workflow when the gateway issues mining requests and receives an access request from the clients.

First, the gateway determines whether a PoW solution is attached to the access request. If an access request does not contain a PoW solution, the gateway sets up a new mining request. Specifically, the gateway updates the nonce search space such that each client searches a nonce in a different search space. Then, the gateway specifies the difficulty that the PoW solution must satisfy. Moreover, the gateway specifies a hash value, which is a digest of the transactions to be recorded in the blockchain. Finally, the gateway calculates a MAC computed over the mining request.

If an access request is made with a PoW solution, the gateway validates the MAC of the mining request attached to the access request. The gateway also validates the PoW solution by recalculating the nonce value using the mining parameters. Finally, the gateway mediates the access request and server response. Note that the gateway acts as an HTTP proxy in cases using HTTP.

5.5 Strategy to determine mining difficulty

We can identify strategies to define the mining difficulty, which affects the time required to find a PoW solution. For each request, we classify the strategies into naive fixed difficulty and dynamic difficulty.

5.5.1 Fixed difficulty across requests. Here, we propose a strategy to issue fixed difficulty across access requests.

Ensure server processing. In this case, the difficulty is set such that the request speed equals the request processing speed of the given server. The gateway adjusts the difficulty using feedback from the server. Specifically, if the request speed is greater than the server processing speed, the gateway increases the mining difficulty.

5.5.2 Difficulty adjusted by request load. For efficient attacks, sophisticated attackers issue crafted requests that burden both the server and backend databases with heavy loads (e.g., requests that enumerate all entries in a database). Compared with light weight requests, the rate of such heavy-load requests must be limited. To address such crafted requests, we prose two strategies to adjust the difficulty dynamically.

Dynamic difficulty for static content. For static content, such as fixed HTML files and image files, the server load is proportional to the size of the requested content. Thus, the difficulty is adjusted on the basis of the size of the content. Assuming only static content, the difficulty (i.e., the time to solve a PoW) of a request is calculated as follows.

$$\text{Time to solve a PoW} = \frac{\text{Size of requested file}}{\text{Server capacity to send files}} \times \text{Number of users} \tag{14}$$

Dynamic difficulty for dynamic content. For dynamic content, the difficulty can be determined on the basis of the processing time of the servers and backend databases. With this strategy, attackers sending crafted requests can be blocked efficiently because of the high difficulty. This strategy is also fair for legitimate users because a user requesting a heavy-load process must wait longer than users requesting light-load processes. Here, a challenge is estimating the processing time of a request. For the same requests, processing time can be estimated by measuring the processing time of the previous request. For complex requests, processing time estimation is difficult, and this remains an open issue.

5.6 Deployment models

To mitigate a DDoS attack against the gateway, the computational power of the gateway must be scalable by deploying multiple gateway instances. Here, we discuss two gateway deployment scenarios.

CDN deployment. The gateway works in a stateless manner; thus, distributed deployment is feasible because gateways do not need to share states with each other. Ideally, DDoS requests should be blocked at locations near malicious clients in order to avoid the consumption of network resources for a DDoS attack. Thus, gateways are deployed in each autonomous system (Figure 5). When a DDoS attack is detected on the server side, the server identifies the attack to the gateway deployed in the autonomous systems. The gateway then performs DDoS mitigation.

Existing CDN-based DDoS mitigation comprises a set of various mitigation techniques, such as IP blacklisting, stateful inspection, behavior analysis, and mitigations against bandwidth exhaustion attacks. Thus, it is effective to simultaneously deploy our mechanism and existing CDN-based mitigation techniques. Our mechanism reduces the number of CDN cache servers due to the PoW, which reduces the attack rate.

On-premise deployment. Similar to existing server load balancing techniques, multiple gateways are deployed before the server (Figure 6). The gateway server works in a stateless manner. Specifically, the gateway server does not have any databases. Due to this statelessness, the number of gateway instances can be increased by simply adding new instances, and this feature suits the resource elasticity of cloud computing. Moreover, the function of the gateway can be implemented as a server component, e.g., as a module of the Apache web server.

In addition to our mechanism against L7 DDoS attacks, solutions against bandwidth exhaustion attacks are also required to counteract DDoS attacks in an integrated manner. For example, an organization can use both an ISP's solution against bandwidth exhaustion attacks and our mechanism.

6 IMPLEMENTATION AND EVALUATION

In this section, we evaluate the performance of the proposed architecture. We specifically address the following research questions.

- The primary concern is gateway performance because it is directly related to DDoS mitigation capability. Thus, the first is, *does the gateway have sufficient performance to mitigate a DDoS attack?* We answer this question by measuring the performance of a gateway prototype (Section 6.2).
- The second question asks: *how scalable is the gateway?* We answer this question by varying the number of CPUs and measuring performance.
- The final question asks: *how does the proposed solution limit cases where a DDoS attack makes economic sense in terms of the conditions discussed in Section 4.5?* To answer this question, we estimate the DDoS parameters in Section 6.3.1 and evaluate the conditions in Section 6.3.2.

6.1 Implementation

We implement core components to answer the above questions. Specifically, we implement a function for a mining request issue, a function for the validation of PoW solutions, and a proxy function to mediate a server and clients. For the mining request and validation, we do not employ an actual cryptocurrency. Instead, we evaluate the performance of the gateway using dummy hash calculations based on Bitcoin specifications. We implement these functions in the Go language and the `net/http/httputil` package, which provides a reverse proxy function (`http.ReverseProxy`). The gateway issues JavaScript as a mining request, which includes a SHA256 calculation library and the parameters (hash value of a previous block, hash value of a new block, etc.) for mining. The client searches for a PoW solution (i.e., mines the coin) using JavaScript and re-accesses the server using the solution. Finally, the gateway mediates the HTTP access after validating the PoW solution.

6.2 Performance measurement

To measure performance, we deploy the proposed architecture on Amazon EC2 instances. We deploy a client instance to run a web benchmark (`t2.micro` instance with one virtual CPU), a gateway instance (`c4.xlarge` instance with four virtual CPUs), and a server instance (`t2.micro` instance with one virtual CPU). With these

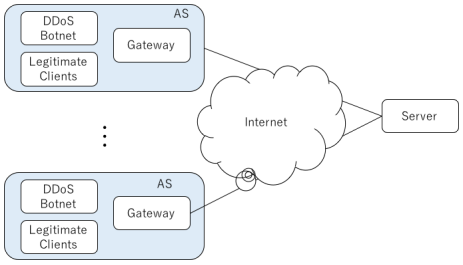

Figure 5: CDN deployment

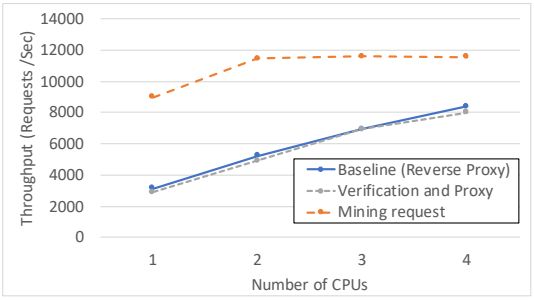

Figure 7: Gateway performance evaluation

instances, we measure performance using `wrt` [1] as an HTTP benchmark tool and `nginx` [2] as an HTTP server. Here, we use the default nginx welcome page (612 bytes). Then, we vary the number of CPUs used by the gateway by specifying the `GOMAXPROCS` parameter.

Figure 7 shows the throughput of the gateway while issuing the mining request and verifying the coin. With only one CPU, the gateway can issue 9,000 mining requests per second. The gateway can handle 3,000 verifications and proxy HTTP requests, and its performance is the same as the reverse proxy without the verification function. Thus, we conclude that our gateway is lightweight and demonstrates sufficient performance. Moreover, the processing throughput is proportional to the number of CPUs; thus, we conclude that our gateway is scalable. The cost of an Amazon EC2 `c4.xlarge` instance is only $0.249/h in the California region. Thus, we can mitigate a heavy DDoS attack by increasing the number of instances at low cost.

We also measure the latency of the gateway in three cases (Table 2), i.e., the benchmark tool directly connecting the server, a Go reverse proxy without our gateway functions (baseline), and a proxy with the PoW verification function. Compared with the baseline, our gateway increased latency by only a few milliseconds, which can be negligible compared with the required mining time.

6.3 Evaluation of the DDoS economic conditions

Here, we evaluate the DDoS conditions described in Section 4.5 based on actual DDoS cases. We first estimate the DDoS parameters, and then evaluate the conditions.

[1] wrt, https://github.com/wg/wrk
[2] nginx, https://nginx.org/

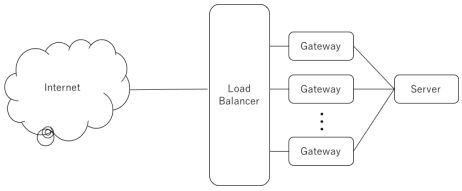

Figure 6: On-premise deployment

Table 2: Gateway communication latencies

Item	Latency (msec)	StdDev (msec)
Direct connection	18	7
Baseline (simple reverse proxy)	49	25
PoW verification and proxy	53	42

6.3.1 Estimation of parameters. To examine the conditions of the DDoS game, we estimate the profit generated by mined coins (P_m), the cost of deploying the proposed defense mechanism (C_d), and the ransom money parameter α.

Estimation of mining profit P_m of a gateway instance. Assuming that two CPUs issue mining requests and two CPUs verify the PoW solutions, an Amazon `c4.xlarge` instance can issue 4,000 mining requests and 4,000 PoW solutions verifications each second.

If a client can find a PoW solution in 1 min on average, a single gateway instance can have $4,000 \times 60$ s computation time each second. In other words, the gateway can handle $4,000 \times 60$ attack hosts. Assuming that 1 h (3,600 s) computation time mines $0.01, the income of the gateway instance (max P_m per instance) is $4,000 \times 60 \times \frac{0.01}{3,600} = 0.67$ ($/second).

Estimation of P_m based on actual DDoS attack. We estimate profit based on the botnet sizes of actual DDoS attacks. According to articles about DDoS [9][21], botnets (unique IP addresses) used for L7 DDoS comprised 89,158 and 128,833 bots. In both cases, only a single instance is required to deal with the DDoS attacks. The former case's coin income (P_m) is $\frac{89,158*0.01}{3600} = 0.25$($/second). For the latter case, P_m is $\frac{128,833*0.01}{3600} = 0.36$ ($/second). Note that we assume one IP address is dedicated to a single attack host. However, the number of actual hosts would be large due to NAT, which allows multiple hosts to share a single IP address.

Next, we discuss a case of an IoT botnet. A recent study on the Mirai [5] botnet noted that can perform an HTTP flood attack. This same study also mentioned that Mirai infected 600,000 devices. Note that the computational power of such devices is poor compared with that of computers. Assuming the devices have 1% of a computer's computational power[3], coin income (P_m) becomes 0.017 ($/second). This profit is less than the above cases; however, such devices requires more computation time due to the limited CPU resources. As a result, the PoW-based mitigation is effective.

[3] There are many types of devices; thus, we conservatively assume low-end routers in reference to an article (https://romanrm.net/router-cpu-performance).

We consider that the above estimations are reasonable because recent attackers have mined coins using their botnet to obtain profit. Our mechanism can earn a similar amount of profit by making the botnet used by DDoS attacks mining the coins.

Estimation of defense cost C_d. We estimate the cost of the defense mechanism (C_d) using the cost of Amazon EC2 instances (on which the gateway processes run). The cost of an Amazon instance of `c4.xlarge` is \$0.249/h = \$0.000069/sec, and the cost C_d can be negligible compared with coin income (P_m). Thus, P_m and C_d satisfy the economic condition (Equation 7), and we conclude that deployment of the gateway instance is beneficial to DDoS victims.

Assumption of α. The target of the DDoS attack is a web service. The attacker demands 10% of the website's value ($\alpha = 0.1$). In most cases, ransom money is only a few BTC; thus, α would be less than 10%, ; however, here, we use a conservative value.

6.3.2 Evaluation of the conditions. Here, we analyze the conditions under which a DDoS attack is not viable, as discussed in Section 4.5, using the above parameters.

Condition1: Assuming $C_m = 0$ (i.e., an attacker has their own botnet), $P_m = 0.36$ \$/second and $\alpha = 0.1$, the condition becomes $3.6 > V_v$ \$/second. This means that small and medium business are not targeted by rational DDoS attackers.

Condition2: Here, we can ignore defense cost C_d as Section 6.3.1, and the condition becomes $P_m > V_v$. In the best case (where $P_m = 0.36$ \$/second), a small business would satisfy this condition and would not need to address DDoS attacks.

Condition3: Assuming $\alpha = 0.1$ (\$/h), a business satisfying $P_m > 0.9 \times V_v$ which is nearly the same as the condition 2, would not need to address DDoS attacks.

From the above analysis results, we conclude that the proposed solution limits the economic motivations of DDoS attackers, especially for small and medium business.

7 SECURITY ANALYSIS

Here, we conduct a security analysis of the proposed architecture.

7.1 DDoS attack against gateway

The gateway performs two tasks, i.e., issuing mining requests and verifying requests from clients, and an attacker may perform a DDoS against these two tasks.

Massive access requests. Attackers may send a large number of access requests to the gateway such that the gateway consumes its computer resources for issuing the mining requests. To avoid the gateway becoming a bottleneck, the issue process of the mining request should be lightweight. Our gateway does not manage its state; thus, the process of the mining request issue is lightweight. Moreover, due to the stateless feature, the number of gateway instances can be increased easily and load balancing can be performed. Thus, we conclude that DDoS attacks via massive access requests against the gateway are difficult.

Invalid PoW solutions. The gateway must verify the nonce attached to requests from the clients. To perform a DDoS attack against the gateway, the attackers can send fake PoW solutions such that the gateway is over-burdened with verification tasks.

However, the risk of this attack is small because the PoW solution verification task is lightweight. For example, with Bitcoin, the verification process only requires a single hash calculation. Moreover, hash calculations can be accelerated using hardware, such as a GPU. Thus, the gateway can reject fake requests at minimal computational cost.

Furthermore, the gateway can deploy a penalty mechanism for clients who submit invalid coins. Specifically, the IP addresses of clients submitting invalid PoW solutions are recorded in an IP blacklist for a certain period. Then, L3 filtering drops packets from blacklisted clients. This L3 filtering is lightweight compared with the PoW verification process in L7; thus, the gateway can resist a DDoS attack.

7.2 Sabotage

A sabotage attack [30] is an attack against cryptocurrency mining pools. Here, a mining pool distributes mining tasks to miners in the pool. Malicious miners in the pool can adapt the following strategy. (1) When malicious clients find the nonce that satisfies the difficulty of the mining pool but does not satisfy the difficulty of the cryptocurrency, the clients submit the nonce to the mining pool for reward (this nonce is referred to as a partial solution). (2) When the clients find a nonce that meets both difficulties (called a full solution), the clients do not submit the nonce. With this strategy, malicious clients obtain a reward from the mining pool, but the malicious client does not share its outcome with the mining pool. Note that malicious clients do not take ownership of the coin because the mined coin is associated with the mining pool's wallet. Thus, this attack has no benefit to attackers; however, it does reduce the mining pool's profit.

In cases where honest miners are dominant, this attack is not serious; however, assuming a large botnet, this attack would be critical for our architecture. We can identify some mitigations, but each has disadvantages that allow attackers to counter the mitigations.

- Verifying the ratio of full and partial solutions. Malicious clients performing sabotage do not submit full solutions; thus, the ratio of full and partial solutions is small. One disadvantage of this is the time required to collect solutions because the mining pool must collect several solutions to avoid false positives caused by statistical fluctuations.
- Introducing cross validation of solutions from multiple clients. The mining pool may specify the same nonce search space for two or more clients. Then, the mining pool compares the solutions from different clients. If the number of legitimate clients and number of malicious clients are of the same order of magnitude, this measure functions effectively. However, if malicious clients are dominant, its effectiveness is limited because malicious clients collude to drop the full solutions. In addition, this measure reduces both the mining power and the victim's profit.

As described above, there is no effective solution to completely prevent an attack. However, even if the attacker performs a sabotage, the attacker must still perform calculations for the partial solutions; thus, the DDoS rate is reduced due to the PoW.

7.3 Pool of PoW solutions

To generate a large number of requests in short time, attackers may compute valid nonces and pool nonces prior to executing DDoS attacks. The attacks still require a PoW solution, and the victims obtain profits from a PoW. Thus, an economic measure is effective; however, the attacks can still force the victims' services offline. To prevent these attacks, we can limit mining requests, set an expiration time for mining requests, and overlap the nonce search space.

Limiting request issue. Opportunities for DDoS attackers to perform mining can be limited if the gateway does not issue mining requests under normal operating conditions, i.e., not under DDoS attack. Thus, attackers cannot begin mining prior to the DDoS. Note that attackers require the nonce search spaces specified by the mining requests; thus, attackers cannot compute the nonce in advance.

Setting an expiration time for the mining request. To prevent pooling PoW solutions, the mining request has an expiration time. The gateway embeds an expiration time into the mining request, and the verifies the time at which a clients' PoW solution is received. However, an expiration time is not a perfect solution. A short expiration time efficiently prevents a pool attack; however, clients must retrieve a new mining request when valid nonces are not found before the allotted time expires.

Overlapping the nonce search space. Relative to expiration times, attackers may pool valid nonces and submit them just before they expire. To reduce the motivation to pool nonces, the gateway forces clients to compete with each other by assigning the same nonce search space to multiple clients. Therefore, keeping a found nonce is not a good strategy because the nonce would be identified by other legitimate users, and users may submit the valid nonce before the attackers submit the nonce.

8 DISCUSSION

Ethics. Malware and malicious websites make users mine coins, and the proposed mechanism employs a similar scheme. To address ethical issues, the mechanism is activated only when a service is under DDoS attack, and mining is not required under normal operating conditions. In addition, the service provider must notify users that mining-based DDoS defenses are deployed, and user must agree to the use of such mining-based defenses. Note that the coins mind by users are refunded after a DDoS attack ends. However, there is no consensus to use this type of mechanism for DDoS defense, and this remains future work.

Do legitimate users need to mine coins? Legitimate users also mine coins to access servers, which means that the customers of the service pay additional costs. The service owner identifies legitimate users who mine coins, and the owner should return the profit derived from mining coins to the legitimate users to increase only the attackers' cost. In addition, to ensure that the income from mining is transparent, the service provider should disclose relevant information, such as duration and extent of the DDoS attack, the income derived from mining, and the cash returned to the users.

Proposed architecture vs. DDoSCoin. DDoSCoin [36] has been proposed to prove that PoW and cryptocurrency can be used for malicious purposes. To find a PoW solution, the client must access an HTTPS server. Multiple TLS connections are established to find a solution to a cryptographic puzzle that uses TLS key exchange parameters and the parameters of a signature issued by the server.

Using the DDoSCoin scheme, DDoS attackers can obtain coins by executing a DDoS attack against an HTTPS server. However, considering the proposed architecture, to attack the victim's server, DDoS attackers must engage in cryptocurrency mining. If the profit obtained from DDoSCoin is less than the profit obtained from legitimate cryptocurrency mining, the attackers are not motivated to participate in DDoSCoin.

Which cryptocurrency should be used? Various types of cryptocurrencies exist, and it is also possible to create our own cryptocurrency. To date, we have not determined the best cryptocurrency; however, we have identified a strategy to select it. Some cryptocurrencies are mined using hardware, such as ASICs, dedicated to coin mining. However, because it is unfair, some cryptocurrencies [11][29] implement a countermeasure against hardware-assisted mining. Thus, we could adopt these types of cryptocurrencies. Otherwise, legitimate users wait time increases because, typically, their computers do not have hardware accelerators.

What about attackers who do not care about attack cost? The proposed approach is also effective against attackers who do not care about attack cost because they need to mine coins to reach the target web server. Therefore, attack speed decreases due to the mining process.

Disadvantages of the proposed architecture. In terms of latency, the proposed architecture also requires mining for legitimate users; thus, latency relative to accessing a service increases. However, we believe this additional latency is better than service unavailability.

In terms of fairness, requests should be processed in a first-in-first-out manner. With the proposed architecture, the processing order changes due to the time required for mining. For example, the wait time of a powerful computer with a GPU hash accelerator is less than that of a smartphone. Thus, not all devices can benefit from the proposed architecture equally. Moreover, mobile devices, such as smartphones and tablets that require a mobile battery, cannot take advantage of the proposed system. However, recently, mobile devices have become more powerful, which will make using proposed architecture more feasible.

9 RELATED WORK

9.1 Case study: ransom-driven DDoS attacks

Criminal groups that commit ransom-driven DDoS (RDoS) attacks include DDoS for Bitcoin (DD4BC), the Armada Collective, and the Phantom Squad. Arbor Networks and Akamai have reported that DD4BC has targeted more than 100 organizations [1, 2]. DD4BC primarily attacks financial services companies; however, they have also targeted online games, media and entertainment platforms, retail companies, and hotel and travel services. Typically, DD4BC sends extortion e-mails that demand 1-100 BTC. They performed L7 attacks, i.e., GET Flood attacks, in addition to bandwidth exhaustion attacks. Unfortunately, the size of the botnet used for RDoS attacks has not been reported. The Armada Collective, which targets banks, retail companies, and hosting services, also demands payment in

Bitcoin [3, 4]. To threaten target organizations, the Armada Collective performed sample attacks over a short period (e.g., 15-30 min). Phantom squad [12] uses the same threatening technique to obtain a ransom.

9.2 PoW-based DDoS mitigation techniques

To reduce the speed of DDoS requests, PoW techniques that require solutions to computational puzzles have been proposed. With client puzzles [16], proposed by Juels and Brainard, clients must solve a computational puzzle before they are allowed to access a server. The proposed model is an extension of this technique. Compared with the client puzzle technique, our contributions are as follows. PoW solutions provide cryptocurrency coins to mitigate damage as well as an economic model and analysis of the attacker's motivation. Hashcash [6] employs three cost functions that output a token based on the PoW, i.e., a fixed function that requires fixed CPU resources, a publicly auditable function that can be verified by a third party, and a probabilistic cost function that requires probabilistic puzzles.

Lazy Susan [10], a technique that does not require solving a computational puzzle, provides a latency-based PoW that sends a delay message to clients. Lazy Susan issues a TCP delay-cookie, which is an extension of a SYN cookie. To prevent tampering, a delay time value and a MAC are encoded in the TCP delay-cookie in a SYN-ACK message sent from a server to a client. The clients must wait for the delay prior to sending the ACK message to the server. Note that Lazy Susan does not require CPU-bound and memory-bound PoW; thus, Lazy Susan does not consume the clients' computer resources. Raincheck [19] provides a virtually infinite queue, i.e., it issues tickets specifying when a server provides a service. The ticket contains a MAC; thus, attackers cannot issue fake tickets. This technique simply uses tickets to extend the server's queue, which means it is also effective for flash crowd attacks. However, Raincheck is insufficient for powerful attackers, and the wait queue can be excessively long for clients.

Captcha is a Turing test used to distinguishes humans and machines. A typical captcha presents a picture that includes words and requires recognition of the words. Other captchas require audio recognition or a solution to a simple puzzle. These techniques identify requests issued by programs that mimic humans. Unfortunately, some techniques can bypass captcha [8][23][33]; thus, captchas are not effective. In addition, complex captchas impose a burden on legitimate users.

9.3 Novel DDoS mitigation techniques

Novel mitigation techniques, such as collaborative defense and moving target defense (MTD), have been proposed. The proposed system and DDoS mitigation techniques are complementary and can be deployed simultaneously.

Defcom [26] is a DDoS mitigation mechanism that uses a collaborative defense deployed via ISPs. Defcom nodes share attack information, such the DDoS source IP address, and perform rate limiting. SIBRA [7] provides a bandwidth reservation protocol for the Internet that guarantees resource allocation based on contracts between autonomous systems; thus, minimum bandwidth allocation can be ensured.

MTD [14][15] is a concept developed to dynamically change system configurations, such as IP address assignments and network topologies, to mitigate attacks. Typically, to maximize damage and the likelihood of success, attackers investigate targets prior to executing an attack. MTD interferes with such reconnaissance by periodically changing system configurations. Therefore, attack costs increase because attackers must perform reconnaissance again after an MTD is executed. Unfortunately, with MTD, system management becomes difficult because the configuration changes dynamically. In addition, MTD would be ineffective if the attackers can identify how the configuration has changed. For example, even if the IP address of a targeted server changes, attackers only need to detect and attack the new IP address.

9.4 DDoS attack costs

MIDAS [35] is an impact scale based on the economic damage to ISPs caused by DDoS attacks. Traditional DDoS scales, such as packets and bits per second, are insufficient to assess economic damage because the extent of the damage depends on the operator's infrastructure capability. To measure economic impact, a MIDAS scale is calculated on the basis of service level agreement violation penalty and the risk of customers leaving, which reduces the ISP's revenue. Akamai also provides a tool [4] to calculate the cost of a DDoS attack. The total loss associated with a DDoS attack is calculated on the basis of lost revenue, brand damage, and operational costs, e.g., the cost of IT and help desk personnel reacting to react the DDoS attack. These economic models and tools can be integrated with our economic model to estimate the precise profit of the DDoS attackers.

Segura and Lahuerta proposed an economic incentives model for DDoS attackers [31]. In their model, the extortion is calculated on the basis of the victims' revenue and the percentage of victims who give in to blackmail. They also have estimated the cost of engaging a DDoS attack service. In addition, they have modeled the DDoS cost considering the attack bandwidth and duration.

Karami and McCoy have studied DDoS-as-a-Service [17]. They analyzed TwBooter based on the leaked operational TwBooter database. The analysis demonstrated that TwBooter obtained more than $7,500 a month and that it executed more than 48,000 DDoS attacks against 11,000 victims.

9.5 Malicious use of cryptocurrencies

Cryptojacking [13] is the unauthorized use of a computer to mine cryptocurrencies. Crypto-mining code can be surreptitiously embedded in a user's computer via an infected web server thereby hijacking legitimate users' CPU resources. Coinhive [5] is a cryptojacking service that issues a script to mine Monero. The design of our proposed solution is similar; however, our solution targets DDoS attackers to reduce attacker motivation.

Some types of malware have functions to mine cryptocurrencies [18]. It is estimated that such malware botnets could generate

[4] https://www.akamai.com/us/en/products/cloud-security/calculate-the-cost-of-ddos-attacks.jsp
[5] https://coinhive.com/

up to $30,000 a month. As discussed in Section 4.4, financially motivated attackers are expected to mine coins if the profit from mining is greater than the returns from RDoS attacks.

10 CONCLUSION

We have propose a unique technique to disincentivize attackers from launching a DDoS attack by increasing its cost. To execute a DDoS, attackers must mine cryptocurrency coins as a PoW, and victims obtain coins by solving the PoW. Thus, attackers have fewer opportunities to profit from DDoS attacks. We evaluate attacker strategies in a game theory manner and demonstrate that the proposed solution gives a negative economic impact to attackers. In future studies, we intend to address a solution to sabotage attacks and evaluate the proposed technique in a real-world network environment. We believe that the proposed technique provides an effective new paradigm relative to the mitigation of DDoS attacks.

REFERENCES

[1] 2015. *ASERT Threat Intelligence Report DD4BC DDoS Extortion Threat Activity.* Technical Report. Arbor Networks. http://pages.arbornetworks.com/rs/082-KNA-087/images/ATIB2015-04DD4BC.pdf.

[2] 2015. *Case study: summary of operation DD4BC.* Technical Report. Akamai. https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/dd4bc-operation-profile-bitcoin-extortion-ransom-case-study.pdf.

[3] 2016. Armada Collective is back, extorting Financial Institutions in SW. https://www.govcert.admin.ch/blog/19/armada-collective-is-back-extorting-financial-institutions-in-switzerland. (2016).

[4] 2017. *ERT Threat Alert. The Bottom Line: RDoS On The Rise.* Technical Report. Radware. https://security.radware.com/ddos-threats-attacks/threat-advisories-attack-reports/bottom-line-rdos-rises/.

[5] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. 2017. Understanding the Mirai Botnet. In *26th USENIX Security Symposium (USENIX Security 17).*

[6] Adam Back et al. 2002. Hashcash-a denial of service counter-measure. (2002).

[7] Cristina Basescu, Raphael M. Reischuk, Pawel Szalachowski, Adrian Perrig, Yao Zhang, Hsu-Chun Hsiao, Ayumu Kubota, and Jumpei Urakawa. 2016. SIBRA: Scalable Internet Bandwidth Reservation Architecture. In *Proceedings of Symposium on Network and Distributed System Security (NDSS).*

[8] Elie Bursztein and Steven Bethard. 2009. Decaptcha: Breaking 75% of eBay Audio CAPTCHAs. In *Proceedings of the 3rd USENIX Conference on Offensive Technologies (WOOT'09).*

[9] Daniel Cid. 2015. Analyzing Popular L7 Application DDoS Attacks. https://blog.sucuri.net/2015/09/analyzing-popular-layer-7-application-ddos-attacks.html. (2015).

[10] Jon Crowcroft, Tim Deegan, Christian Kreibich, Richard Mortier, and Nicholas Weaver. 2007. *Lazy Susan: dumb waiting as proof of work.* Technical Report UCAM-CL-TR-703. University of Cambridge, Computer Laboratory.

[11] dEBRYUNE, dnaleor and the Monero project. 2018. PoW change and key reuse. https://ww.getmonero.org/2018/02/11/PoW-change-and-key-reuse.html. (2018).

[12] Brad Duncan. 2017. Emails threatening DDoS allegedly from Phantom Squad. https://isc.sans.org/forums/diary/Emails+threatening+DDoS+allegedly+from+Phantom+Squad/22856/1. (2017).

[13] S. Eskandari, A. Leoutsarakos, T. Mursch, and J. Clark. 2018. A first look at browser-based Cryptojacking. *ArXiv e-prints* (2018). arXiv:1803.02887

[14] Jafar Haadi Jafarian, Ehab Al-Shaer, and Qi Duan. 2012. Openflow Random Host Mutation: Transparent Moving Target Defense Using Software Defined Networking. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks (HotSDN '12).*

[15] Q. Jia, K. Sun, and A. Stavrou. 2013. MOTAG: Moving Target Defense against Internet Denial of Service Attacks. In *2013 22nd International Conference on Computer Communication and Networks (ICCCN).*

[16] Ari Juels and John G Brainard. 1999. Client puzzles: A Cryptographic countermeasure against connection depletion attacks.. In *NDSS*, Vol. 99. 151–165.

[17] Mohammad Karami and Damon McCoy. 2013. Understanding the Emerging Threat of DDoS-as-a-Service. In *Presented as part of the 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats.* USENIX.

[18] Kaspersky Lab. 2017. Got any hidden miners? I wouldn't be so sure... https://www.kaspersky.com/blog/hidden-miners-botnet-threat/18488/. (2017).

[19] Y. H. Kung, T. Lee, P. N. Tseng, H. C. Hsiao, T. H. J. Kim, S. B. Lee, Y. H. Lin, and A. Perrig. 2015. A Practical System for Guaranteed Access in the Presence of DDoS Attacks and Flash Crowds. In *2015 IEEE 23rd ICNP.*

[20] Denis Makrushin. 2017. The cost of launching a DDoS attack. https://securelist.com/the-cost-of-launching-a-ddos-attack/77784/. (2017).

[21] Michael Mimoso. 2016. IoT Botnet uses Http traffic to DDoS targets. https://threatpost.com/iot-botnet-uses-http-traffic-to-ddos-targets/121199/. (2016).

[22] Carlos Morales. 2018. NETSCOUT Arbor Confirms 1.7 Tbps DDoS Attack. https://asert.arbornetworks.com/netscout-arbor-confirms-1-7-tbps-ddos-attack-terabit-attack-era-upon-us/. (2018).

[23] Marti Motoyama, Kirill Levchenko, Chris Kanich, Damon McCoy, Geoffrey M. Voelker, and Stefan Savage. [n. d.]. Re: CAPTCHAs: Understanding CAPTCHA-solving Services in an Economic Context. In *Proceedings of the 19th USENIX Conference on Security (USENIX Security'10).*

[24] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[25] Harikrishna Narasimhan, Venkatanathan Varadarajan, and C. Pandu Rangan. 2010. Game Theoretic Resistance to Denial of Service Attacks Using Hidden Difficulty Puzzles. In *Information Security, Practice and Experience.* 359–376.

[26] G. Oikonomou, J. Mirkovic, P. Reiher, and M. Robinson. [n. d.]. A Framework for a Collaborative DDoS Defense. In *2006 22nd Annual Computer Security Applications Conference (ACSAC'06).*

[27] Radware. 2017. HTTP Flood Tools: Inside the Hackers Arsenal. https://security.radware.com/ddos-threats-attacks/threat-advisories-attack-reports/hackers-arsenal-http-flood-tools/. (2017). Accessed: 2017-08-17.

[28] Redwolf. 2017. What is the maximum bandwidth / throughput of the DDoS test in Gbps? https://www.redwolfsecurity.com/resources/maximum-bandwidth-throughput-ddos-test-gbps/. (2017).

[29] Reuben Yap. 2018. ASIC Resistance Is Still Worth the Fight for Egalitarian Mining, This Time With Merkle Tree Proofs (MTP). https://cryptoslate.com/asic-resistance-is-still-worth-the-fight-for-egalitarian-mining-this-time-with-merkle-tree-proofs-mtp/. (2018).

[30] Meni Rosenfeld. 2011. Analysis of Bitcoin Pooled Mining Reward Systems. *CoRR* abs/1112.4980 (2011). http://arxiv.org/abs/1112.4980

[31] Vicente Segura and Javier Lahuerta. 2010. Modeling the economic incentives of ddos attacks: Femtocell case study. In *Economics of information security and privacy.* Springer, 107–119.

[32] Alireza Shameli-Sendi, Makan Pourzandi, Mohamed Fekih-Ahmed, and Mohamed Cheriet. 2015. Taxonomy of Distributed Denial of Service Mitigation Approaches for Cloud Computing. *J. Netw. Comput. Appl.* 58, C (Dec. 2015), 165–179.

[33] Suphannee Sivakorn, Jason Polakis, and Angelos Keromytis. 2016. I'm not a human: Breaking the Google reCAPTCHA *(Black Hat Asia).*

[34] T. Spyridopoulos, G. Karanikas, T. Tryfonas, and G. Oikonomou. 2013. A game theoretic defence framework against DoS/DDoS cyber attacks. *Computers & Security* 38 (2013), 39 – 50. https://doi.org/10.1016/j.cose.2013.03.014 Cybercrime in the Digital Economy.

[35] R. Vasudevan, Z. M. Mao, O. Spatscheck, and J. Van der Merwe. 2007. MIDAS: An Impact Scale for DDoS attacks. In *2007 15th IEEE Workshop on Local Metropolitan Area Networks.* 200–205.

[36] Eric Wustrow and Benjamin VanderSloot. 2016. DDoSCoin: Cryptocurrency with a Malicious Proof-of-Work. In *10th USENIX Workshop on Offensive Technologies (WOOT 16).*