Erciyes University Journal of the Institute of Science and Technology
Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi
ISSN 1012-2354
Cilt (Volume): 32, Sayı (Issue): 1, Ocak/January-2016
http://fbe.erciyes.edu.tr/

ddosdaps4web: Web'e Yönelik DDoS Tespit ve Koruma Yöntemi

Abdullah Talha KABAKUŞ [1,*], Resul KARA [2]
[1] *Abant İzzet Baysal Üniversitesi, Bilgi İşlem Daire Başkanlığı*
[2] *Düzce Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü*

Anahtar Kelimeler: Servis engelleme, dağıtık servis engelleme, güvenlik, web, HTTP, DoS, DDoS

ÖZET

Her koruma tespitle başlar. Dağıtık servis engelleme (DDoS) saldırıları, ağları veya bilgisayarlara yoğun kullanım sonucunda verdikleri servisi engellenmektedirler. Günümüzde bilgisayarlardaki yazılımsal ve donanımsal gelişmelere rağmen, kısa bir zaman dilimi DDoS ataklarının kötücül etkilerini gerçekleştirmesi için yeterli olmaktadır. Bu sebepten ötürü DDos saldırılarını engellemek için gerçek zamanlı bir tespit ve koruma sistemine ihtiyaç duyulmaktadır. Geleneksel ağ tabanlı koruma sistemleri uygulama katmanı DDoS ataklarına karşı güvenlik sağlayamamaktadır. Bu çalışmada, HTTP tabanlı DDoS ataklarını tespit etmek ve sistemi korumak için *ddosdaps4web* isimli DDoS tespit ve koruma sistemi öne sürülmüştür. *ddosdaps4web* üç servisten faydalanmaktadır: (1) Tüm HTTP isteklerinin depolanıp, istek başlıklarından detaylı analiz için bilgi çıkartımı yapılmasını sağlayan *depolama servisi*, (2) her dakika çalışan ve ön tanımlı istek limitlerine göre kötücül istekleri tespit etmeyi sağlayan *izleme servisi*, ve (3) gelen bütün istekleri keserek, oluşturulan kurallara göre kötücül olanları devre dışı bırakan *durdurucu servisi*. *ddosdaps4web* rastgele oluşturulmuş 10000 HTTP isteği üzerinden test edilerek DDoS doğru tespit oranı %94 olarak bulunmuştur.

ddosdaps4web: DDoS Detection and Protection System for Web

Key Words: Denial of Service, Distributed Denial of Service, Security, Web, HTTP, DoS, DDoS

ABSTRACT

Protection starts with detection. Distributed denial of service (DDoS) attacks flood networks or computers in order to deny their services. Due to advances in the modern computers in terms of hardware and software, a small amount of time is enough to complete their malicious actions. Therefore, a real-time detection and protection is required in order to prevent DDoS attacks. Traditional network based protection systems are not able to provide a security for application layer DDoS attacks. In this paper, we propose a DDoS detection and protection system namely *ddosdaps4web* in order to detect and protect the system from HTTP based DDoS attacks. *ddosdaps4web* uses three services: (1) *Storage service* stores all HTTP requests and extracts information from request headers for further analysis, (2) *Monitoring service* runs every minute to detect malicious requests through predefined request limits and constructs rules in order to prevent current and upcoming attacks, and (3) *Interceptor service* filters all incoming requests to eliminate malicious ones through the constructed rules. ddosdaps4web is evaluated by randomly generated 10000 HTTP requests and its accuracy is calculated as 94%.

*Sorumlu Yazar (Corresponding author) e-posta: talha.kabakus@ibu.edu.tr

1. Introduction

Security is as old as internet itself. According to the report by Web Hacking Incident Database 2011 (WHID 2011) [1], Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are one of the most common threats in the web. The only purpose of DoS attacks is to consume a computer's or network's resources in order to prevent its service. WWW Security FAQ [2] categorizes DoS attacks through their targets: (1) Bandwidth attacks consume network resources such as network bandwidth to busy the whole network traffic, and (2) Connectivity attacks consume computer resources such as central processing unit (CPU) and memory through high volume of requests to make the computer no longer process the benign requests.

Traditional (D)DoS attacks are carried out of at network layer using Transmission Control Protocol/User Datagram Protocol (TCP/UDP). Application layer (D)DoS attacks target server resources such as sockets, CPU, memory, disk/database bandwidth and I/O bandwidth [3]. At the same time, this type of attacks is typically more efficient than network based one by requiring fewer network connection to achieve their malicious purposes. They are also harder to detect due to they do not involve large amount of network traffic. Unlike the network based DDoS attacks, application layer DDoS attacks do not necessarily rely on inadequacies in the underlying protocols or operating systems; they can be easily mounted with benign requests from legitimately connected hosts [4]. Traditional network based DDoS detection and protection mechanisms contain following issues to detect application layer DDoS attacks [4]: (1) Network based DDoS detection methods are unable to collect enough signals due to they belong to different layers, (2) TCP anomaly detection mechanisms can hardly identify HTTP DDoS attacks based on successful TCP connection, and (3) anomaly detection by IP packet does not work due to attackers use legitimate IP addresses.

Hyper Text Transfer Protocol (HTTP) requests are one of the most common ways of application layer (D)DoS attacks that target web servers. Web is built on client-server architecture which lets clients to communicate with servers through internet. The way clients communicate with servers varies through applications. Websites provide their services using HTTP – a communication protocol based on TCP/IP protocol suite. Web clients send request messages *(HTTP Request)* to web servers through any platforms that support HTTP. Then web servers return response messages *(HTTP Response)* back to clients. Fig. 1 illustrates how HTTP works.

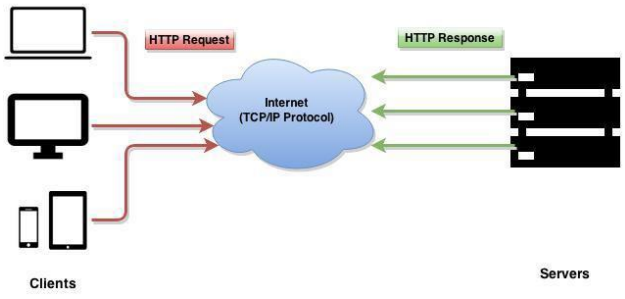

Figure 1. Illustration of how HTTP works

assigns a continuous value and the scheduler decides whether and

HTTP (D)DoS attacks are generally launched by generating huge volume of request traffic to consume web servers' resources such as CPU, memory, disk bandwidth and also the network bandwidth that these servers use to connect other servers such as database servers, mail servers, FTP servers. Generally, attackers first establish a network of compromised hosts that are commonly known as botnets and can include hundreds or even thousands compromised systems. Then, attackers install all required tools on these hosts and launch attacks [3].

Application layer DoS attacks are launched by specific clients and can be easily prevented after detecting the client by its IP address. But due to DDoS attacks use a large number of geographically widespread hosts and more dynamic methods such as IP changing, a more comprehensive approach is necessary to detect them. Therefore, the proposed system should provide the following features:

- ***Feature Extraction:*** Features from malicious requests should be extracted as many as possible in order to provide protection against them.
- ***Dynamic Rule Construction:*** Due to features of malicious requests change by time, the proposed system must be able to construct rules against DDoS attacks dynamically.
- ***Real-Time Monitoring:*** The system must monitor all incoming requests in real-time to be aware of attacks before they complete their malicious actions.
- ***Real-Time Protection:*** The attacks should be blocked in real-time.
- ***Request Database:*** All requests must be stored in a persistent system (i.e. a database management system) for further analysis.
- ***Attack Simulation:*** In order to expand the system's attack detection and protection capabilities against different types of DDoS attacks, the system must provide both attack and benign HTTP request simulation.
- ***User-friendly Interface:*** The system must provide user-friendly interface in order to monitor and manage attacks.

The rest of paper is organized as follows: Section 2 presents related works against DDoS attacks. Section 3 describes the proposed DDoS detection and protection system for web namely *ddosdaps4web*. Section 4 presents results and discussion. Finally, Section 5 concludes the paper.

2. Related Works

Yuan and Mills [5] propose an early detection of DDoS flooding attacks by monitoring network-wide effects. They benefit from cross-correlation analysis to construct traffic patterns. These patterns are used to inform where and when a DDoS attack possibly arises.

Ranjan et al. [6] presents a counter-mechanism namely *DDoS Shield* that uses statistical methods to detect characteristics of HTTP sessions. *DDoS Shield* consists of a suspicion assignment mechanism and a DDoS-resilient scheduler. The suspicion mechanism of *DDoS Shield* estimator on the mean packet inter arrival times to detect

when the session is serviced. They also calculate session inter-arrival times between consecutive sessions. *ddosdaps4web* does not use timing constraints due to attackers can also benefit from it by making requests periodically to appear as benign visitors. This constraint can protect system from flooding attacks but can also increase false alarms.

Wang et al. [7] propose a relative entropy based application layer DDoS detection method in which the click ratio of the web object is defined and the cluster method is used to extract the click ratio features. Through these extracted features, the relative entropy is calculated to detect suspicious sessions that can be used for DDoS attacks.

Oikonomou and Mirkovic [8] propose defenses against application layer DDoS attacks via behavior modeling to differentiate DDoS bots from human users. They model three aspects of human behavior with regard to (1) request dynamics, (2) request semantics, and (3) ability to process visual cues.

Kandula et al. [9] propose a system that uses Completely Automated Public Turing test to tell Computers and Humans Apart (CAPTCHA) - a graphical test to determine whether the client is under control by a human or not - to protect web servers from DDoS attacks. CAPTCHAs are powerful components to prevent DDoS attacks for websites that contain submissions through `HTTP POST/PUT` requests. But not all websites serve in this way; there are lots of websites that are not in need of any submissions and provide their services through `HTTP GET` requests. This brings a target audience limitation for this system. ddosdaps4web is intentionally developed to detect DDoS attacks for any type of websites whether it is in need of an authentication or not. Another disadvantage of using CAPTCHAs for accessing websites is preventing search engines to index the content due to CAPTCHAs deny web crawlers access to websites.

Jung et al. [10] propose DDoS attack detection method with benefiting client clustering, per-client request rate and adaptive content delivery network. The proposed method has following inefficiencies: (1) it is not always possible to determine the amount of resources each client consumes, and (2) filtering clients by IP address prefixes is not enough to detect attacks due to botnets consist of large number of geographically widespread machines. For this reason, *ddosdaps4web* does not use any geographic information to detect DDoS attacks.

Park et al. [11] propose a method based on human-computer interactions to detect bots. They use mouse or keyboard keystroke events, random requests instead of link following as patterns of human web browsing. But an attacker can implement a bot to generate mouse or keyboard keystroke events and even more s/he can disable these events due to they use JavaScript which can be easily overridden. Due to some users/systems tend to disable JavaScript for security issues, this approach may increase false positives.

Basu et al. [12] present an approach that classifies connections as malicious or not by achieving real-time detections with a confidence measure. Their system consists of several components: a network parser, a feature extractor, a classifier, and an output processor. They allow user to adjust false detection alarm trade-off through the provided output variable.

Shiaeles et al. [13] propose a method that constructs a fuzzy DDoS attacks. If the observed packet arrival time is less than the mean packet arrival time, then they regard the event as a DDoS attack. While they focus on network based DDoS attacks, *ddosdaps4web* focuses on application layer DDoS attacks with providing a flexible rule based protection. *ddosdaps4web* detects DDoS attacks through more detailed parameters such as location of the attacker, the software attacker uses.

3. *ddosdaps4web*

The developed system DDoS Detection And Protection System for Web namely *ddosdaps4web* contains three main services to detect both application layer DoS and DDoS attacks to prevent the system from their malicious actions: (1) ***A storage service*** that stores all HTTP requests with request details such as request IP address, session ID, request date, user agent, location including latitude and longitude on a relational database management system, (2) ***A monitoring service*** runs per minute to detect malicious requests that repeat more than 20 times per minute and constructs rules against them using their signatures, and (3) ***An interceptor service*** that filters all incoming requests to eliminate malicious ones through the constructed rules. The system components are illustrated in Fig. 2.

ddosdaps4web provides a user-friendly interface to monitor and manage both malicious and benign requests as it is shown in Fig. 3. *ddosdaps4web* user interface provides:

- Simulate HTTP request DDoS attack
- Simulate benign HTTP request
- Monitor and store all requests with extracted features listed in Table 1
- Filter previously analyzed requests by type (malicious/benign)
- Search requests through their IP addresses
- Manage constructed blocking rules (see Fig. 4)
- Keep history of blocks in order to track attacks and decide how long this block will be active (see Fig. 4)

ddosdaps4web provides continuous and real-time DoS & DDoS attack detection and protection based on three sequential services. Analyzing service works continuously to provide a real-time protection by analyzing all incoming HTTP requests to construct dynamic rules. These rules are used by the interceptor service to check each new HTTP request's session parameters. The workflow of the developed system is presented in Fig 5. Pseudocode of the proposed algorithm is presented in Fig. 6.

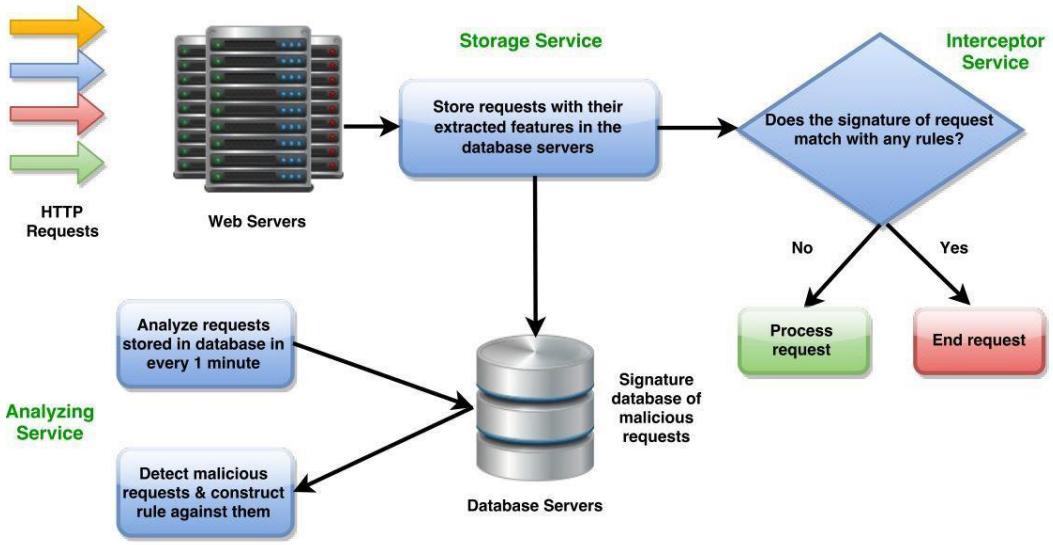

Figure 2. Overview of the proposed system's components

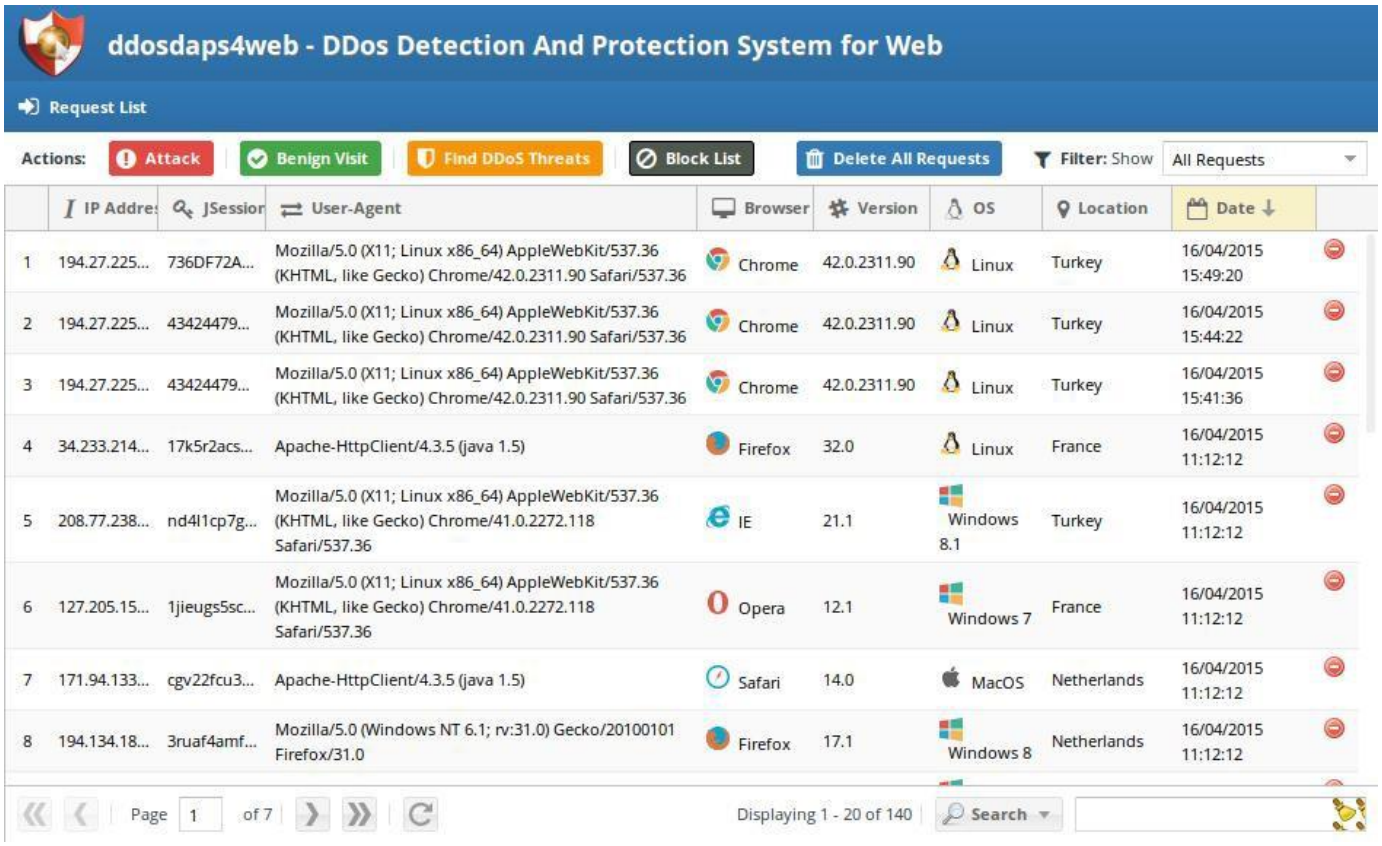

Figure 3. The main user interface of ddosdaps4web

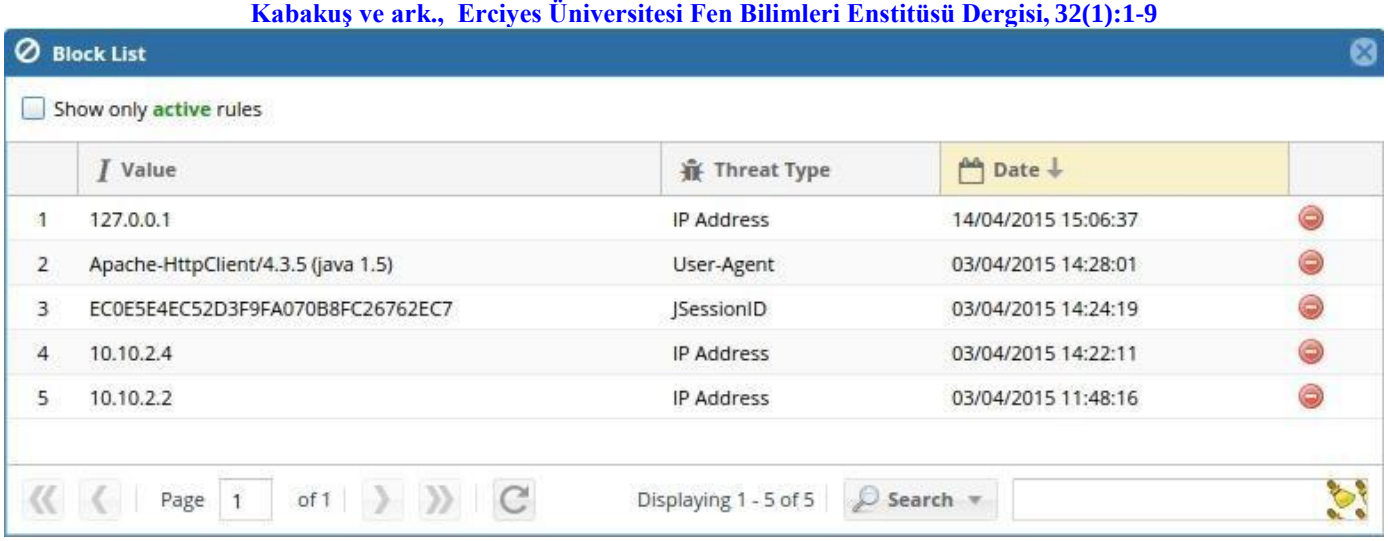

Figure 4. Rule management user interface

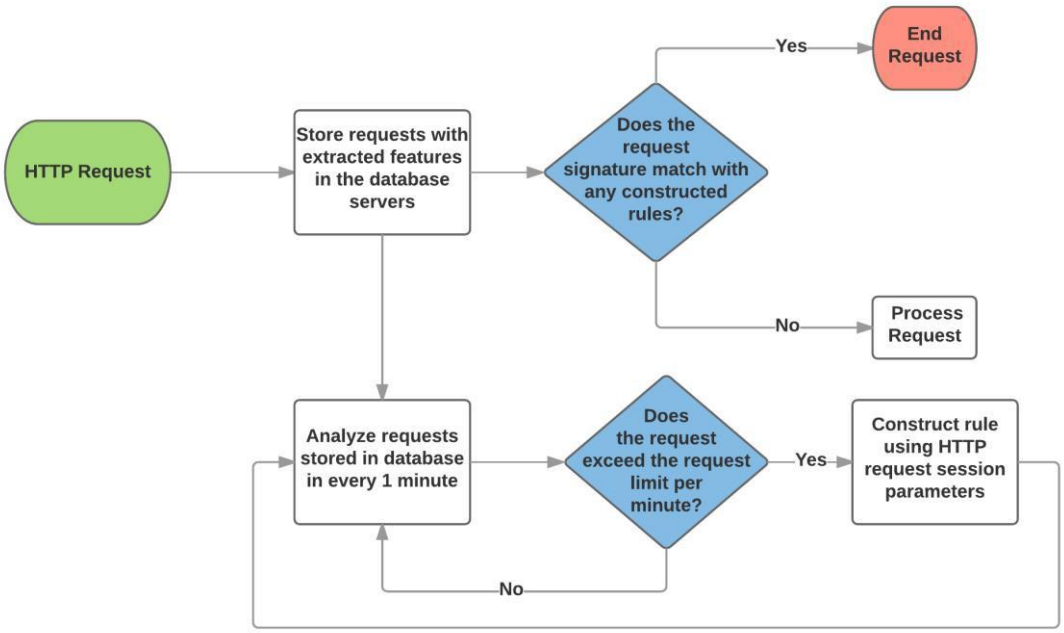

Figure 5. Workflow of *ddosdap4web*

```
For each new_request
      Store request with headers into database
      If signature of new_request matches with any existing rules
            Intercept new_request
      Else
            Process request
      End of if
End of loop

While(true)
      Analyze requests that are stored in last minute
      For each request in the list
            If the request exceeds the request limit per minute
                  Construct rule using the request's headers
            End of if
      End of loop
End of while
```

Figure 6. Pseudocode of the proposed algorithm

3.1. Storage Service

The first step to protect a system from attacks is detection. In order to detect malicious requests, it is necessary to store them to be able to start analyzing. Storage service monitors all incoming requests and store them into a relational database management system. Due to stored features are used to construct security rules against DDoS attacks, choice of features is critical. *ddosdaps4web* stores the features extracted from request header listed in Table 1.

Table 1. Stored features of requests

Feature	Description
IP Address	IP Address of client
Session ID	Session ID of client
User-Agent	Which tool/software is used to make request
Browser	Which browser is used to make request
Browser Version	Which version of browser is used to make request
Operating System	Which operating system is used to make request
Location Latitude	Latitude of the location of request
Location Longitude	Latitude of the location of request
Location Country Code	Country code of the location of request
Location Country Name	Country name of the location of request
Request Date	Request date & time

External IP addresses (over NAT) of requests are retrieved from a web service [14] that *Amazon* provides. Usage of this service is mandatory to get real IP address of the request owner instead of the one assigned inside a private network. Geographic location related to IP address is found by using *GeoIP* [15] – a product that provides libraries in various programming languages to get detailed geographic information such as latitude, longitude through IP addresses. *GeoIP* provides both IPv4 and IPv6 databases that are updated monthly. ddosdaps4web uses IPv4 database instead of IPv6 due to around 97% of global internet traffic is carried by IPv4 [16-18].

3.2. Monitoring Service

Monitoring service logs intercepts all requests and extracts the features listed in Table 1. This service works per minute and it is responsible to construct rules by inspecting requests that repeat more the pre-defined request limit per minute which is set to 20 by default after inspecting daily requests of our universities' home pages. The frequency of executing monitoring service is set to per minute after testing the system performance with several values. The frequency is not increased anymore since the service needs time to compare all incoming HTTP requests' session parameters (signatures) with the previously constructed rules and intercept the matched ones. At the same time, the frequency of analysis is not prolonged in order to prevent backlogs since the service is also responsible both detecting malicious requests and constructing rules against them which is needed to be completed in 1 minute. The developed system is specifically designed to be flexible enough to work with different scale systems in terms of target audience. So the request limit per minute is dynamic and can be set by system administrator through the target audience of system. Then, this service analyzes new (not analyzed yet) requests and marks the requests exceed the request limit as malicious. Then monitoring service constructs rules through the signatures of these malicious requests' in order to stop ongoing and upcoming attacks. Signatures that are used to construct rules to prevent both DoS and DDoS attacks are listed in Table 2.

Table 2. Signatures that are used to construct rules to prevent both DoS and DDoS attacks

Signature	Useful for	Description
IP Address	DoS	All requests from IP addresses marked as malicious are blocked.
Session ID	DoS	Each session has a unique Session ID. When the session ID is marked as malicious, other requests from same session are blocked.
User-Agent	DoS & DDoS	Since DDoS attackers tend to change their IP addresses and Session IDs dynamically, the only option to block this type of attacks is determining their agent. Also, this criteria lets the system to eliminate malicious attacks that use other clients rather than web browsers.

3.3. Interceptor Service

Interceptor service is responsible to block DDoS attacks through the constructed rules by monitoring service. This service intercepts all requests in order to block the requests that match with constructed rules. If none of rules matches, then request is processed and response is returned to the client. Otherwise, the request is immediately intercepted and ended. The work flow of the interceptor service is illustrated in Fig. 7.

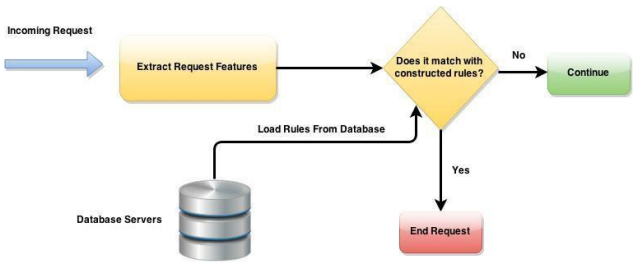

Figure 7. Work flow of *ddosdaps4web*'s

4. Results and Discussion

Protection starts with true detection. In order to test and improve true attack detection ratio, the system is in need of attack simulations. We have searched for a public HTTP DDoS attack logs to evaluate the developed system but for the best of our knowledge, there are some publicly available HTTP DDoS attack datasets such as CSIC 2010 but they do not contain IP address and request time information which are used by *ddosdaps4web* to detect DoS & DDoS attacks. For this reason, we have implemented services to simulate both DoS and DDoS attacks.

4.1. DoS Attack Simulation

There are some both commercial and non-commercial softwares to perform load and stress testing on web applications such as Apache JMeter, HP LoadRunner, The Grinder but due to these tools are designed for benign intentions, they do not let users to simulate DoS or DDoS attacks. Therefore, we have implemented our own DoS attack simulator that lets us to set flexible HTTP headers such as user-agent, referrer and host to simulate real DoS attacks. We are able to set number and frequency of HTTP requests that are going to send the system.

4.2. DDoS Attack Simulation

Due to we do not have a botnet that contains a large number of compromised hosts to simulate DDoS attacks, we have implemented our own services to insert randomly generated HTTP requests in the format listed in Table 2 into the database. This service provides setting HTTP headers listed in Table 3 in order to simulate a real DDoS attack. A list of predefined IP addresses and user-agents is used to simulate malicious requests that target to deny the service. Also, since attackers tend to hide information about the way they attack as much as possible, the developed HTTP DDoS attack simulation service randomly hides some information from requests such as `browser`, `browser version`, `operating system`. The required HTTP headers of requests are `IP address`, `location` and `user-agent` information which cannot be hidden. When simulating DDoS attacks IP addresses and user-agents are restricted to imitate real attacks.

Table 3. List of HTTP headers with header values that are used to simulate HTTP DDoS attacks

HTTP Header	Header Value	Nullable
`IP Address`	A random IPv4 address	false
`Session ID`	A random session ID	true
`User-Agent`	A list of clients such as Apache HTTPClient, GoogleBot, and web browser engines with variants	false
`Browser`	• Chrome • Firefox • Internet Explorer • Safari • Opera • No information	true
`Browser Version`	A random version number in the format X.Y	true
`Operating System`	• Linux • MacOS X • Windows 7, Windows 8, Windows 8.1, Windows 10 • No information	true
`Location`	A list of predefined real locations with their latitude and longitude information retrieved from *GeoIP* database	false

Since we use our own services to simulate DDoS attacks and generate attacks logs, a flag is set into the generated HTTP requests in order to differentiate benign and malicious requests. *ddosdaps4web* is evaluated by using 7000 *malicious* and 3000 *benign* HTTP requests that are generated by the developed simulation services described in sections 4.1 and 4.2. *ddosdaps4web* has successfully detected 94% of these simulated 10000 HTTP requests. Receiver operating characteristics (ROC) analysis is a technique used to visualize, organize and select classifiers based on their performance [19]. Table 4 presents ROC confusion matrix of test results on total 400 unique HTTP requests. ROC curve of results is presented in Fig. 8. Since *ddosdaps4web*'s aim is to detect malicious requests, *'positive'* refers to *malicious* and *'negative'* refers to *benign* requests. Reasons of false detections can be listed as: (1) Requests that have similar signatures with the opposite ones, and (2) both detection and protection is completed in a limited time (1 minute) which may cause to

miss some requests.

Table 4. ROC confusion matrix of *ddosdaps4web*'s test results

	Positive (P)	Negative (N)	Total
True (T)	6474	2932	9406
False (F)	526	78	594
Total	7000	3000	10000

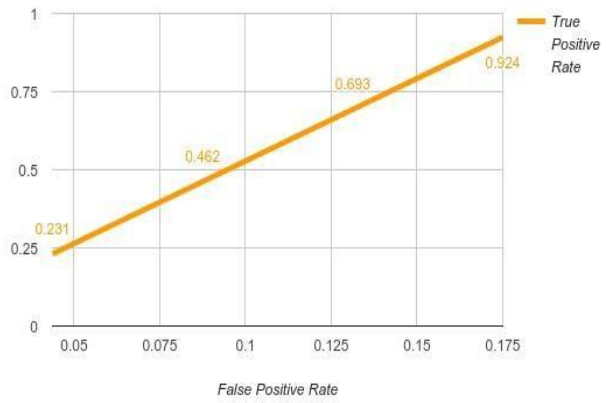

Figure 8. ROC curve of *ddosdaps4web*'s test results

There are many techniques in the literature to detect DDoS attacks but none of them promises a certain solution against HTTP DDoS attacks since attackers' abilities to behave like benign clients. A comparison of related DDoS detection and protection techniques is presented in Table 5.

Table 5. Comparison of related DDoS detection and protection systems

Feature	Park et al. [11]	Ranjan et al. [6]	Shiaeles et al. [13]	*ddosdaps4web*
Usage of patterns/rules	Yes	Yes	Yes	Yes
Usage of statistical methods	No	Yes	Yes	Yes
Real-time protection	Yes	Yes	Yes	Yes
Usage of human-computer interaction techniques	Yes	No	No	No
Usage of HTTP session parameters	No	Yes	No	Yes

4.3. Known Limitations

Due to we do not have botnets, HTTP DDoS attack simulation is applied by generating random requests and storing them with their extracted features into database. Therefore, we could not monitor system resources usage such as control processing unit (CPU), random access memory (RAM). Similarly, we could not simulate our system under a real DDoS attack to reveal how fast it detects attacks and creates security rules against them.

ddosdaps4web detects HTTP DDoS attacks using `IP addresses`, `session IDs` and `user-agent` features. `User-agent` is optional due to advanced attackers may benefit from it. If they set their `user-agent` as a valid browser, *benign* visitors would also be blocked and this will increase false negative (FN) detections. Therefore, this detection feature is optional and disabled by default.

5. Conclusion

Depending on both the attacker and victim resources, DDoS attacks can complete their malicious actions in a few seconds or minutes. Therefore, DDoS attacks bring with two major challenges: First, detection of DDoS attacks is hard due to botnets consist of large number of geographically widespread machines that aim to behave like benign visitors. Second, the detection of DDoS attacks must be completed in nearly real-time in order to protect systems before the victim services suffer from exhaustion of resources. In order to address these challenges, we developed a DDoS Detection and Protection System for Web namely *ddosdaps4web* that uses features extracted from HTTP headers to detect DDoS attacks. As soon as DDoS attacks are detected, ddosdaps4web constructs rules against these attacks using their HTTP headers to protect system from ongoing and also possible upcoming attacks.

The developed system's accuracy can be improved by using machine learning techniques in order to provide continuous and more adaptive system. With this addition, the system will be able to set request limits automatically. Also, *ddosdaps4web*'s ability to deal with nonlinearities and uncertainties can be improved by using soft programming techniques. *ddosdaps4web* already extracts request locations through IP addresses with latitude and longitude details. Another improvement built on this information can be using location based analysis for DDoS attack detection due to some websites only serve locally.

References

[1] "What is Cross Site Scripting and How Can You Fix it?," 2011. [Online]. Available: http://www.acunetix.com/websitesecurity/cross-site-scripting/. [Accessed: 14-Nov-2015].
[2] L. Stein and J. Stewart, "WWW Security FAQ: Securing Against Denial of Service Attacks," W3C, 2015. [Online]. Available: http://www.w3.org/Security/Faq/wwwsf6.html. [Accessed: 14-Nov-2015].
[3] S. R. Devi and P. Yogesh, "Detection Of Application Layer DDOS Attacks Using Information Theory Based Metrics," Comput. Sci. Inf. Technol., pp. 217–223, 2012.
[4] Y. Xie and S. Z. Yu, "Monitoring the application-layer DDoS sttacks for popular websites," IEEE/ACM Trans. Netw., vol. 17, no. 1, pp. 15–25, 2009.
[5] J. Yuan and K. Mills, "Monitoring the macroscopic effect of DDoS flooding attacks," IEEE Trans. Dependable Secur. Comput., vol. 2, no. 4, pp. 324–335, 2005.

[6] S. Ranjan, R. Swaminathan, M. Uysal, A. Nucci, and E. Knightly, "DDoS-shield: DDoS-resilient scheduling to counter application layer attacks," IEEE/ACM Trans. Netw., vol. 17, no. 1, pp. 26–39, 2009.
[7] J. Wang, X. Yang, and K. Long, "Web DDoS detection schemes based on measuring user's access behavior with large deviation," in GLOBECOM - IEEE Global Telecommunications Conference, 2011.
[8] G. Oikonomou and J. Mirkovic, "Modeling human behavior for defense against flash-crowd attacks," in IEEE International Conference on Communications, 2009.
[9] S. Kandula, D. Katabi, M. Jacob, and A. Berger, "Botz-4-sale: Surviving organized DDoS attacks that mimic flash crowds," Proc. 2nd …, pp. 287–300, 2005.
[10] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites," in Proceedings of the 11th international conference on World Wide Web (WWW '02), 2002, pp. 293–304.
[11] K. Park, V. S. Pai, K.-W. Lee, and S. Calo, "Securing web service by automatic robot detection," in Proceeding ATEC '06 Proceedings of the annual conference on USENIX '06 Annual Technical Conference, 2006, p. 23.
[12] R. Basu, R. K. Cunningham, S. Member, S. E. Webster, and R. P. Lippmann, "Detecting Low-Profile Probes and Novel Denial-of-Service Attacks," in Proceedings of the 2001 IEEE Workshop Information Assurance and Security, 2001.
[13] S. N. Shiaeles, V. Katos, A. S. Karakos, and B. K. Papadopoulos, "Real time DDoS detection using fuzzy estimators," Comput. Secur., vol. 31, no. 6, pp. 782–790, 2012.
[14] "Amazon IP Check Web Service," Amazon, 2015. [Online]. Available: http://checkip.amazonaws.com/.
[15] "GeoIP Products - Maxmind Developer Site," maxmind, 2015. [Online]. Available: http://dev.maxmind.com/geoip/. [Accessed: 14-Nov-2015].
[16] "IPv6 Deployment Hits 2%, Keeps Growing | Internet Society." [Online]. Available: http://www.internetsociety.org/blog/2013/09/ipv6-deployment-hits-2-keeps-growing. [Accessed: 14-Nov-2015].
[17] "Google's IPv6 Stats Pass 3% Less Than 5 Months After Passing 2%! | Deploy360 Programme." [Online]. Available: http://www.internetsociety.org/deploy360/blog/2014/02/googles-ipv6-stats-pass-3-less-than-5-months-after-passing-2/. [Accessed: 14-Nov-2015].
[18] "APNIC at the Global IPv6 Summit 2014 | APNIC." [Online]. Available: https://www.apnic.net/publications/news/2014/apnic-at-the-global-ipv6-summit-2014. [Accessed: 14-Nov-2015].
[19] T. Fawcett, "An introduction to ROC analysis," Pattern Recognit. Lett., vol. 27, pp. 861–874, 2006.