ISSN (Print) : 2319-8613
ISSN (Online) : 0975-4024

Agile method for detecting DDoS attacks in the application layer based on user's dynamism

Silvia Bravo [#1], David Mauricio [*2]

[#] Faculty of Engineering and Applied Sciences,
Technical University of Cotopaxi Latacunga, Ecuador
[1] silvia.bravom@utc.edu.ec
[*] Faculty in Systems Engineering and Computer Science,
National University of San Marcos Lima, Peru
[2] dmauricios@unmsm.edu.ec

***Abstract*— DDoS attacks are one of the most damaging computer attacks of recent times. Attackers send large number of requests to saturate a victim machine and it stops providing its services to legitimate users. In general attacks are directed to the network layer and the application layer, the latter has been increasing due mainly to its easy execution and difficult detection. The present work proposes a low cost detection approach that consists of two steps: first, user characteristics are extracted in real time while browsing the web application; second, each extracted feature is used by an order sorter O(1) to differentiate a real user from a DDoS attack. A real user is identified by making requests using peripherals for navigation (user dynamism), while DDoS attacks are requests sent by robots and do not require the use of peripherals to make requests, therefore the characteristics of the user's dynamism are used for the detection of a DDoS attack. The results on the attack tests using the attack tools LOIC, OWASP and GoldenEye, show that the proposed method has a detection efficiency of 100%, and that the characteristics of the web user allow to differentiate between a real user and a robot.**

Keyword - Application layer, DDoS, user's dynamism, detection attacks, use of peripherals

I. INTRODUCTION

DDoS attacks have become one of the threats with the greatest impact on the security of computer systems. These attacks are aimed at consuming bandwidth or server resources, preventing legitimate users from accessing the services. These attacks can be in the network layer (protocols, hubs, switch) and in the application layer (system, CPU, resources), the latter has increased in recent years due to its easy execution and difficult detection, thus, the efforts in the mechanisms of detection are focusing to this type of attack. The attacks directed to the application layer are considered sophisticated because they mimic the requests of real users, so it is more difficult to detect them. The methods consider information of user requests, and some logic that allows to relate these with an attack or a user. The logic is given by techniques such as neural networks, genetic algorithms, support vector machine and statistical models that in general consume considerable resources. The excessive consumption of resources means that the detection process is slower and even more so with large amounts of information. The slowness of the process impacts the system causing saturation of the bandwidth and consumption of server resources. In addition, the mentioned techniques have a waiting time before detection, to know if it is an attack, which affects the productivity of the services. The most difficult task that detection methods have is to differentiate a request to identify it as a real user or attack. In [1] they introduces the characteristics of the user's dynamism, indicating that they come from the interaction between the user and the system. In [2] they specify that the characteristics of the user's dynamism allow differentiating a robot from a real user.

The attack detection mechanisms do not contemplate any of the characteristics of the user's dynamism. Therefore, in this work we propose a simple and low cost method based on the characteristics of the user's dynamism for the detection process of DDoS attacks in the application layer. To do this, keystrokes, mouse dynamics and interaction with the graphical user interface (GUI) [3] for the identification of real users are evaluated. The proposed method has been validated in a case study to evaluate its efficiency. For this, an algorithm has been implemented that detects the interaction of the user and the system. It was tested on a web system in real time. The system has a three-layer architecture to implement the user interface and the detection algorithm. The attacks were generated using the LOIC, OWASP and GoldeEye tools to provoke flood attacks. This work is organized as follows. In section 2, a review of the literature of DDoS attack detection methods at the application layer level is made. Section 3 proposes the agile method of detecting DDoS attacks by using the dynamism of the user. In section 4 the numerical experiments are carried out and the results are shown and, finally, the conclusions are presented in session 5.

II. RELATED WORK

The review of the literature regarding the detection methods of DDoS attacks in the application layer records nine proposed methods. Hidden semi-Markov Model is a method that analyses the statistics of the user's search process and access to web objects [4] [5]. However, it has been proven that robots are able to emulate search patterns and access statistics recorded in a session [6]. A mechanism that counts the requests made by a user in a session called Counter Mechanism was implemented to detect attacks [7]. However, robots can simulate statistics by mimicking requests from real users [8]. A Fuzzy Estimator implemented in an attack detection mechanism allows analysing the number of requests, number of users and access patterns in order to establish statistics to identify anomalies in the system [9]. Attackers have developed robots that are capable of generating requests by imitating the number of requests and users, as well as patterns of access to the system [5]. The correlation analysis has also been used in the detection of attacks, indicating the statistical probability of sending requests from the same group of IP addresses [10]. When the attackers have a group of computers under their control, they can make requests from different places avoiding correlation of the points where the request arises. [11] Support vector machine is used to analyse the statistics of the sessions of each client to later identify the anomalies [12]. For this, in this method the characteristics are used: strings of client, paths of client, all clients of domain, connections of client, response times, request type, payload of all clients. However, these characteristics correspond to statistics of user sessions that have to be processed by SVM, which implies a high computational cost and consumption of server resources. For the detection of attacks, prototype systems were also used, such as the mechanism called intrusion detection system (IDS) [13]. In this mechanism statistics of incoming requests were used as: duration of the conversation, number of packets, number of bytes, average packet size, size of TCP window, average time, percentage of packets, and percentage of encrypted packets. Despite being an innovative proposal, being built in Python, becoming an application aimed at detecting anomalies, resource costs turn out to be high. The Hellinger metric has also been used in the detection of computer attacks [14]. Two techniques have been used for attack detection, Neural Networks and Genetic algorithm. These techniques use the characteristics of incoming requests, analysing the entropy and variance of the captured characteristics. It should be noted that these mechanisms employ features that can be easily simulated by attackers (web page requested, request count) by employing robots that issue requests from low-speed users.

TABLE I. Detection Method and features for the detection of DDoS attacks in the application layer

Method	**Features**	**References**
Hidden semi-Markov Model (HsMM)	Users' browsing process Access to web objects	[4] [5]
Counter mechanism	Session's requests	[7]
Fuzzy estimator	Number of request Number of users Access pattern	[9]
Correlation analysis	IP address	[10]
Support vector machine (SVM)	Average length of query strings of client Number of different resource paths of client Sum of incoming payload of all clients of domain Fraction of connections of client that request the most frequent resource path Sum of response times of all clients of domain Sum of response times of client Fraction of connections for domain that accepts any version of English Entropy of request type (GET/POST/OTHER) Sum of outgoing payload of all clients	[12]
Intrusion Detection System Prototype	Duration of the conversation Number of packets sent in 1 second Number of bytes sent in 1 second Maximal, minimal and average packet size Maximal, minimal and average size of TCP window maximal, minimal and Average time to live (TTL) percentage of packets with different TCP flags Percentage of encrypted Packets with different	[13]

	properties	
Hellinger Distance Metric	Flow similarity Client legitimacy Web page requested	[14]
Neural Networks Genetic algorithm	HTTP GET request count Entropy of the requests Variance of the entropy	[15]

Table I shows the methods and features used by the DDoS attack detection mechanisms in the application layer. In total nine methods and thirty characteristics are observed. It is also observed that SVM uses the greatest number of features for the detection of attacks, which implies high computational costs. The detection mechanisms [13] [15] are the ones with the highest degree of detection, 98.5% and 98.32% respectively. This is mainly due to the fact that in the first case, a system is implemented for the exclusive detection of anomalies, it is implemented in Python. While in the second case, two techniques for data analysis are merged. However, in none of the two cases are the characteristics of user dynamism considered.

III. PROPOSED METHOD

In this work we present a low cost detection method that allows detecting DDoS attacks oriented to the application layer. For this, it uses characteristics of the dynamism of the user extracted in real time. These characteristics show the user's interaction with the system.

A. Architecture of the Detection Method

Figure 1 shows the architecture used for the implementation of the DDoS attack detection method. In the same it is observed the entrance of the requests coming from the Internet to the interface of the web application. The requests made generate a data bank where the established connections and the processes performed are recorded. The data bank generated in the application layer is analysed by an interaction detector. At the application level, the processes that the user generates are recorded (links, resources, forms, etc.). The detector records the activity between the user and the mouse and keyboard peripherals. The characteristics of Table I are extracted in real time by programming in PHP and Javascript. These characteristics are stored until the user executes the next request. Both the request and the characteristics of the user are sent to the detection algorithm of Figure 1 for evaluation. As indicated in the previous section, this algorithm is responsible for determining the existence of requests and interactions with the system, taking a decision between real user or computer attack.

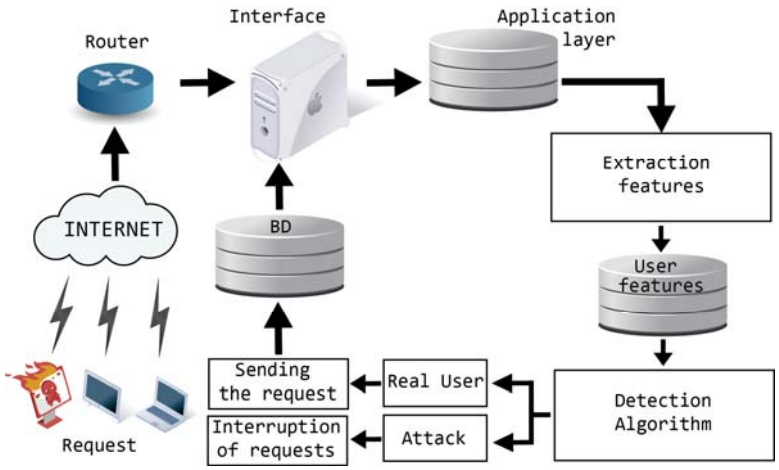

Fig. 1. Architecture of the detection method

B. User Dynamism

The present work considers the characteristics of the dynamism of the user in the computer system. The dynamism of the user arises when the user interacts with the system. In [16], mentions that the user's dynamics are the interests of the users and their preferences. The model of user requests and server responses provide limited knowledge about user behaviour. For better compression it is better to move to the client side. To do this, collect information such as mouse move, click, blur, or resize. In [17] they mentions that an alternative to predict the next web page to be opened by a user, comes from the dynamism of the movement of the mouse with the direction it takes in the graphical interface. In [18] they proposes a technique to identify users by grouping keystroke dynamics. In [19] they used the pulse dynamics, which uses the rhythm and the way in which an individual writes characters on the keyboard, it is used as behavioural biometrics. The keystroke rhythms of a

user, in terms of time, are measured to develop a unique biometric template of the user's typing pattern for future authentication. In [20] they evaluated the characteristics of the mouse to identify real users of DDOS attacks. Checking that the dynamism of the mouse provides unique characteristics to identify this type of attack.

C. Characteristics of the User's Dynamism

The characteristics of user behaviour are extracted from the processes between the peripherals used and the interaction with the system. In this work, the dynamism of the user is observed through the transactions that are made with the mouse and keyboard peripherals. Table II shows the user characteristics that are extracted and used in the proposed DDoS attack detection method. It is worth mentioning that these features are extracted using PHP and Javascript functions in real time.

TABLE III. Features of the user's dynamism

Id	Features	Description
f1	Mouse move	The mouse is moved to a location on the screen to perform an action.
f2	Mouse click	When a user presses and releases a mouse button and there are five types of click events that are recorded: left click, right click, and double left click.
f3	Mouse highlight	This action begins with a left mouse click/hold to begin the highlighting and ends with the mouse release.
f4	Mouse drag	When an object is dragged and dropped. This action begins with a left mouse click/hold and ends with the mouse release
f5	Mouse drop	
f6	Mouse scroll	The Mouse Wheel or Scroll is an event when the movement of the wheel or scroll has a net up or down effect. The resultant effect is based on the consecutive wheel or scroll movements.
f7	Mouse wheel	
f8	Key press	This happens when a user presses a key and slides the touch device (finger or stylus)

D. Architecture of the detection method

Figure 2 shows the algorithm used to capture the characteristics of the user's dynamism. The algorithm works every time the user performs an operation with the mouse or keyboard and its interaction with the graphical interface. When a user interacts with the mouse and keyboard it is registered by means of a Javascript function. The captured characteristics are stored in a register to be sent in the following to be checked by the detection algorithm. When a user requests a service, the user is forced to use a peripheral to make the request. The capture of the user's characteristics consists of taking the pulsations that the user is making with the peripherals. When a user interacts with a peripheral it is registered by a Javascript function in a data bank or registry.

```
1. begin catch
2. when interaction then
3. id = operation
4. if request then
5. submit id
6. end catch
```

Fig. 2. Algorithm for capture features (ACF).

E. Detection and mitigation algorithm

The main idea of the algorithm is to verify if the request made to the system presents any of the characteristics of the web user to differentiate a real user from a computer attack in real time.

1. Input request
2. begin verification
3. for i equal 1 to 8
4. if fi stores true then id stored true
5. end verification
6. begin send each request
7. when id stores true then execute query request
8. when id stores false then execute message
9. end send

Fig. 3. Algorithm for detection of DDoS attacks (ADDA).

In Figure 3 the proposed attack detection algorithm is presented, it uses the characteristics of the dynamism of the web user that are shown in Table I and verifies if they are active or not. For this purpose, 1) a request is made to the system, 2) the verification of the captured characteristics begins, 3) a repetitive loop is used that goes from one to the total of characteristics used in this work, 4) if the analysed characteristic they have been activated, the activations performed will be stored in another variable, 5) and the verification of the characteristics of the user's dynamism is completed. 6) When a request is made, the characteristics of the user's dynamism must be verified. 7) When the variable that stores the verification is active, the request is made. 8) When the variable that stores the verification is inactive, a message is sent that must be answered by the user otherwise the request will not be given. This last step is in mitigating the algorithm that requests from attackers are sent to the server.

IV. Numerical Experiments

A. *Experimental Design*

To validate the proposed algorithm, the web services of a hotel in the city of Detroit in the United States were considered. It receives 3100 passengers annually. The hotel has a restaurant service and the information of it is in a dedicated server, it uses Linux CentOS, 4-core processor, 8 GB memory, 1TB disk space. Figure 4 shows the built-in validation environment that allows the incorporation of three levels for the detection of DDoS attacks. The first level involves the user interface, where the user interacts with the system making requests for links, videos, graphics, etc. In the second level are located the functions that load of information to all the applications of the system. In this level are the functions that perform the call to the ACF algorithm. Finally, on the third level is the ADDA algorithm. The response of the algorithm has two outputs, execute the request made by the user or send a verification message.

Fig. 4. Validation environment for detection of DDoS attacks.

The server used in this work has been subjected to a series of simulated attacks to verify the efficiency of the proposed method. The results obtained have been extracted using the same attack tools for later analysis.

B. Simulation of Attacks

To generate DDoS attacks LOIC software (Low Orbit Ion Canon) [21], OWASP DOS HTTP POST [22] and GoldenEye HTTP [23] were used. It is worth mentioning that these tools were selected because they are the most used for the generation of this type of attacks, due to their simplicity and effectiveness [23]. To do this, several attacks were made with each tool towards the hotel server (victim), in order to evaluate the attack rate needed to overload the server. In each attack, the overload values were obtained, which would then be evaluated using the proposed detection algorithm. Table III shows the tools that were used to simulate the attack on the web system, the amount of solitudes generated and the time it took the system to overload.

TABLE IIIII. Evaluation Results without detection method

Tool attack	**Number of request**	**System overload time (min)**
LOIC	4800	2.15
OWASP DOS HTTP POST	4000	1.30
GoldenEye HTTP Denial Of Service Tool	5300	3.20

The results of Table III show that the computer attacks generated by the LOIC, OWASP and GoldenEye software use about two minutes to overload the system, causing inaccessibility to resources and services for real users. It is also observed that the number of requests used to overcharge the system varies between 4000 and 5300.

C. Results

Table IV shows the results obtained using the proposed detection method. It shows 100% of attacks generated by the tools have been detected effectively. The time used in the detection was on average 60 milliseconds and the same amounts of simulation requests were used to generate the attack. It is worth mentioning that there are no dataset related to DDoS attacks for tests. In addition, the works with the highest detection rate in the application layer [13] and [14] do not show the tools that were used to evaluate the proposed methods.

TABLE IVV. Evaluation results with the detection method

Tool attack	**Number of request**	**Detection time (mil)**	**Detection rate %**
LOIC	4800	60	100
OWASP DOS HTTP POST	4000	58	100
GoldenEye HTTP Denial Of Service Tool	5300	63	100

Table IV shows that the detection mechanism developed through the use of web user dynamism features is effective with a 100% detection rate for the three attack generation tools. In addition, the time spent is around 60 milliseconds. These results show the effectiveness of the detection method through user interaction with the system through the peripherals used. It should be mentioned that with the improvement of the detection mechanisms, the attackers also improve their attack strategies, so the possibility that the input values of the user characteristics evaluated in this work can be supplanted is not ruled out.

V. CONCLUSION

This paper presents an agile and effective detection mechanism based on the characteristics of the web user's dynamism for the detection of DDoS attacks in the application layer. This mechanism employs eight new characteristics of user behavior that have not been used in any other similar work. The method of detecting DDoS attacks using the characteristics of user behavior has a 100% effectiveness in detection. This result shows the influence of the characteristics that identify a user when interacting with the system. The tests in a real-time platform and the application of the attack tools LOIC, OWASP and GoldenEye allow to evaluate the algorithm under a simulated attack environment. These simulations allowed to verify that the algorithm reaches an optimal result when processing large quantities of requests.

REFERENCES

[1] I. Brosso, A. La Neve, G. Bressan, and W. V. Ruggiero, “A continuous authentication system based on user behavior analysis”. Availability, Reliability, and Security, 2010. ARES'10 International Conference on IEEE, 2010.
[2] G. Oikonomou, and J. Mirkovic. "Modeling human behavior for defense against flash-crowd attacks." Communications, 2009. ICC'09. IEEE International Conference on IEEE, 2009.
[3] M. Abramson, & D. W. Aha, User Authentication from Web Browsing Behavior. FLAIRS conference, 2013.
[4] Y. Xie and S. Z. Yu, Monitoring the application-layer DDoS attacks for popular websites. IEEE/ACM Transactions on Networking (TON), vol. 17, no 1, p. 15-25, 2009.
[5] C. Huang, J. Wang, G. Wu, and J. Chen, Mining Web User Behaviors to Detect Application Layer DDoS Attacks. JSW, 9(4), 985-990, 2014.
[6] F. Yu, Y. Xie, and Q. Ke, Sbotminer: large scale search bot detection. Proceedings of the third ACM international conference on Web search and data mining, pp. 421-430, 2010.

[7] S. Ranjan, R. Swaminathan, M. Uysal, A. Nucci and E. Knightly, DDoS-shield: DDoS-resilient scheduling to counter application layer attacks. IEEE/ACM Transactions on Networking (TON), vol. 17, no 1, p. 26-39, 2009.
[8] C. Ye, and K. Zheng, Detection of application layer distributed denial of service. Computer science and network technology (ICCSNT) 2011 International Conference, Vol. 1, pp. 310-314, 2011.
[9] L. C. Giralte, C. Conde, I. M. De Diego, E. Cabello, Detecting denial of service by modelling web-server behaviour. Computers \& Electrical Engineering, vol. 39, no 7, p. 2252-2262, 2009.
[10] W. Zhou, W. Jia, S. Wen, Y. Xiang, W. Zhou, Detection and defense of application-layer DDoS attacks in backbone web traffic. Future Generation Computer Systems, 38, 36-46, 2014.
[11] N. Hoque, H. Kashyap, and D. K. Bhattacharyya, Real-time DDoS attack detection using FPGA. Computer Communications, 110, 48-58, 2017.
[12] U. Dick, T. Scheffer, Learning to control a structured-prediction decoder for detection of HTTP-layer DDoS attackers. Machine Learning, 104(2-3), 385-410, 2016.
[13] M. Zolotukhin, T. Kokkonen, T. Hämäläinen and J. Siltanen, On Application-Layer DDoS Attack Detection in High-Speed Encrypted Networks. International Journal of Digital Content Technology and its Applications, 2016.
[14] R. Saravanan, S. Shanmuganathan and Y. Palanichamy, Behavior-based detection of application layer distributed denial of service attacks during flash events. Turkish Journal of Electrical Engineering \& Computer Sciences, 24(2), 510-523, 2016.
[15] K. Johnson Singh, K. Thongam and T. De, Entropy-Based Application Layer DDoS Attack Detection Using Artificial Neural Networks. Entropy, 18(10), 350, 2016.
[16] L. A. Leiva, Mining the browsing context: Discovering interaction profiles via behavioral clustering. User Modeling, Adaptation and Personalization (UMAP), 19, 2011.
[17] A. Kundu, Dynamic web prediction using asynchronous mouse activity. Computational Social Networks Springer, London, pp. 257-280, 2012.
[18] S. Sznur, Advances in Keystroke Dynamics Techniques to Group Users Sessions. International Journal of Information Security Science, 4(2), 26-38, 2015.
[19] G. Kulkarni, R. Chandorkar and N. Chavan, A Security By Biometric Authentication. International Journal of Computer Science and Engineering Research and Development (IJCSERD), 2(1), 7-14, 2012.
[20] S. Bravo, D. Mauricio and Á. H. Moreno, Mouse Features for DDoS Attacks Detection in the Application Layer. Proceedings of the 9th International Conference on Information Management and Engineering, pp. 177-181, 2017.
[21] L. Y. Zhang, Q. I. A. N. Ming and Y. B. Chi, DDoS Attack Detection Using Sliding Window Method. DEStech Transactions on Computer Science and Engineering, 2017.
[22] M. Y. Arafat, M. M. Alam and M. F. Alam, A Practical Approach and Mitigation Techniques on Application Layer DDoS Attack in Web Server. International Journal of Computer Applications, 131(1), 2015.
[23] H. H. Jazi, H. Gonzalez, N. Stakhanova and A. A. Ghorbani, Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling. Computer Networks, 121, 25-36, 2017.

AUTHOR PROFILE

Silvia Bravo was born in Latacunga, Ecuador. She graduated from the Technical University of Cotopaxi in 2007, where she received the title of “Computer Science”. She is currently pursuit a Ph.D. from National University of San Marcos within the Doctoral Program of “Computer and System”. She is currently working as a professor and researcher at the Faculty of Engineering Science, in the Technical University of Cotopaxi. Her research activity is mainly focused on the software development and informatics security.

David Mauricio was born in Lima, Peru. He graduate from the National San Marcos University in 1987, where he received the title of “Computer Science”. He obtained the title of “Master in Mathematics Applied” from the Federal University of Rio de Janeiro, Brazil, in 1991. In 1994, he obtained the title of “Doctor in Systems Engineering” from the Federal University of Rio de Janeiro. He is currently working as a professor at the Faculty of Systems Engineering, in the National Mayor de San Marcos University and scientific consultant in National Council for Science and Technology (CONCYTEC). His research activity is mainly focused on the combinatorial optimization, designs and analysis of algorithms, heuristics search, metaheuristics, mathematical programming, expert systems, data mining, and artificial intelligence.