International Journal of Security and Its Applications
Vol. 11, No. 4 (2017), pp.85-96
http://dx.doi.org/10.14257/ijsia.2017.11.4.07

# Application Layer DDOS Attack Detection Using Hybrid Machine Learning Approach

Rizwan ur Rahman, Deepak Singh Tomar and Jijin A.V.

*Maulana Azad National Institute of Technology, India*
*rizwan.rahman12@gmail.com,deepaktomar@manit.ac.in*

***Abstract***

*Application Layer Distributed Denial of Service (App-DDoS) attack has become a major threat to web security. Attack detection is difficult as they mimic genuine user request. This paper proposes a clustering based correlation approach for detecting application layer DDoS attack on HTTP protocol. Proposed approach has two main modules ----Flow monitoring module and User behavior monitoring module. Flow monitor is responsible to analyze data flow information. User behavior monitor analyses end user behavior. Proposed approach is capable to detect three main attacks on HTTP protocol, i.e. HTTP-GET attack, HTTP-POST attack and Slow Read attack. It is also possible to detect hybrid type of DDoS attacks which uses a mixture network and application layer DDoS techniques. Comparative analysis of clustering algorithms on generated dataset is also done to demonstrate the effectiveness of detection approach.*

***Keywords****: Dos, DDoS, App-DDoS, Botnet, web service*

## 1. Introduction

Web services have now become an integral part of daily life. Keeping the service available is of supreme importance for every organization. Unavailability of these services may create a drastic effect on users and organizations using them. Web services are now facing security threats from both Local and Distributed Denial of Service (DDoS) attacks. Traditional DDoS attacks targets Network (Layer-3) and Transport Layer (Layer-4). The main intent of these attacks is to make service unavailable by consuming bandwidth of the network. As more DDoS defense strategies have emerged around Layer-3 and Layer-4, attackers shifted their target to Application Layer (Layer-7).

In Application Layer DDoS attack network is not flooded with unbounded traffic as in Network or Transport Layer attacks but it exploits flaws in application protocols [7]. Application Layer attacks are also difficult to detect as attack traffic resembles genuine client traffic [5]. Tracing of application layer attack is a painful task as attackers uses a chain of proxies during attack. One of the main advantages of layer-7 attack over traditional DDoS is that it doesn't need huge botnet army for attack. Even a single machine can cause devastating effect on the target with the help of highly efficient attack scripts.

Only a few methods exist for Application Layer DDoS attack detection. This paper proposes a hybrid approach to capture behaviour and flow details of users for Layer-7 DDoS attack detection. Average Request Rate, Average Request Duration and Average Bytes transferred are analysed to capture Flow behaviour of user. User Access Matrix is defined to identify the user's access behaviour on a particular webpage. Clustering based correlation approach is applied on the flow and behaviour information to identify the attacker's. Proposed approach is validated by attack traffic generated with the help of open source software's like Slowloris, Pyloris and Slowhttptest.

The remainder of the paper is organized as follows. Section-2discusses phases of DoS attack, Layers targeted by attack and types of application layer attacks. Section-3

ISSN: 1738-9976 IJSIA

describes prior and ongoing Application Layer attack detection and defense methods. Section-4 presents the proposed approach for Application layer attack detection. Section-5 describes the environmental setup for attack generation. Section-6 present outcome of the experiment and Section-7 concludes the paper.

## 2. Denial of Service Attack

Denial of Service (DoS) attack is an attempt by which attackers try to prevent services of a resource to legitimate users. Services are disrupted either temporarily or indefinitely to intended users. A more intense form of DoS attack is Distributed DoS attack. A typical Denial of service (DoS) attack can be divided into three phases

- Target Acquisition phase
- Background work
- Attack phase

In target acquisition phase attacker identifies vulnerable host in the network and collects information about them. Information like current services running in the host, open ports, load balancing servers available etc are collected during this phase. In background work, attacker uses the information collected during the first phase to identify the weakness in the services running on the victim machine. Finally, in the attack phase, attacker uses his knowledge which he has gained about the target in first two phases to perform the attack. Real attack is performed during this phase.

Earlier DoS and DDoS attack mainly focused on Network Layer (layer-3) and Transport Layer (layer-4) of OSI model. Weaknesses in the network and transport layer protocols are exploited to perform the attack. As most of the DoS defence strategies are built around layer 3 and 4, attackers started shifting their target to layer 7.

### 2.1. Network and Transport Layer Attacks

Network layer (Layer 3) and Transport Layer (Layer 4) DDoS attacks are usually volumetric attacks which floods network infrastructure with UDP/ICMP /IGMP/TCP packets. These types of attacks are difficult to defend even with additional hardware if an attacker sends more traffic than a network can handle. Most of these attacks originates from large number of distributed machines working together to overwhelm a target system.

### 2.2. Application Layer Attacks

Application layer DDoS attacks occurs mainly because of poor coding or flaws in application layer protocols. Protocols like DNS, HTTP, FTP etc are prone to DDoS attacks. HTTP-GET and HTTP-POST and Slow Read Attack are the three important application layer attacks on HTTP.

1. HTTP-GET Attack*:* During HTTP-GET attack complete header field is not send at once. Header is divided into many small packets and is send one at a time. HTTP specifies a threshold time limit up to which it can maintain connection before it receives next header field. So the attacker sends next header field before timeout is reached to maintain the connection making victim server waiting for remaining header fields. As it gets packets in regular intervals connections won't get terminated. Like this multiple connection are established during attack, there by exhausting all victim resources. Victim's service becomes slow or unavailable after the attack.
2. HTTP-POST Attack*:* HTTP-POST is another type of HTTP attack in which the attacker initially establishes a complete TCP connection. Attacker builds header

 

section and send in full to the target machine. Once header part is send, the body of the message is sending in sequence (1 byte per 110seconds). HTTP header contains a field called Content-Length which describes the length of the HTTP message body. Victim machine waits for the complete message whose length is specified in Content-Length field. If the size of message body is large, more time is required to transfer the message there by maintaining the connection indefinitely. Multiple connection of this type exhausts all the resources and victim services become unavailable.

3. Slow Read Attack: It targets the same resources as HTTP-POST and HTTP-GET attacks. In Slow Read Attack attacker exploits the fact that modern web servers are not restricting connection duration even if there is a zero or minimal data flow [4]. Here, attacker controls the incoming data rate by manipulating receive window size to a small value there by making victim server to maintain the connection.

## 3. Related Work

In this section literature survey of various detection mechanisms for Application Layer DDoS attack is analyzed. It should be note that researchers try to differentiate genuine user request from attack traffic in their work for attack detection. S. Ranjan, *et al.* [7] proposed a counter mechanism called DDoS shield to defend against application layer DoS attack. It has two parts suspicious assignment mechanism and DDoS-resilient scheduler. Session history is used by suspicious mechanism to assign a suspicion measure to each client session. DDoS-resilient scheduler decides the requests to be forwarded according to service rate and scheduling policy. One of the main disadvantages of DDoS shield is that it cannot distinguish flash crowd from attack traffic.

J. Yu, *et al.* [5] divides application layer attacks are divided into three classes: session flooding attacks, request flooding attacks and asymmetric attacks. Defence and Offense Wall (DOW) is proposed for attack prevention. Anomaly detection mechanism is used to filter asymmetric and request flooding attack. Encouragement model is proposed to defend against session flooding attack. Detection model helps to drop suspicious sessions, while the currency model encourages legitimate sessions. DOW creates delay in servicing client request as encouragement module put forward puzzles to the clients for authentication.

Sheng Wen, *et al.* [6] proposed an architecture extension called CLAD to protect web server against attacks that mimics flash crowd. It uses a front end sensor to monitor incoming traffic. In case of intense pulse in traffic, front end sensor sends a signal to back end detection mechanism. Mess extend of IP address is used to identify flash crowd from attack traffic in CLAD. CLAD also helps to separate security mechanisms away from the web server.

Y. S. Choi, *et al.* [10], proposed a model called Timeslot Monitoring Model (TMM) is used to generate normal and attack user profile. It extracts IP address of attacker from legitimate traffic. TMM uses SVM for classification but it fails to identify low rate DDoS attacks

K. Zheng, *et al.* [3], applies clustering method to analyse application layer DDoS attack. Features like average size of objects requested, request rate, average popularity of all objects, average transition probability is used for clustering. Model uses hierarchical clustering to discover clusters.

Sujatha Sivabalan, *et al.* [9] proposed CAPTCHA and AYAH to detect and prevent DDoS attack. CAPTCHA uses a puzzle authentication mechanism to determine suspicious users. AYAH is also a mechanism similar to CAPTCHA which can dynamically determine signature of non-human users. Disadvantage of using this model is that occasionally it causes some delay for user authentication.

S. Renuka Devi, *et al.* [8] describes a method to detect application layer DDoS and to schedule flash crowd during attack. To detect suspicious users from normal users and flash crowd access matrix is used. It uses 3 step methods *i.e.*, collecting the data, data abstraction and attack detection for defending against DDoS attack.

From the literature survey it is observed that none of the detection mechanisms considers HTTP slow read attack which is an emerging form of Application Layer DDoS attack. It is also observed that detection of HTTP-POST attack has given little importance during attack detection. So there is a need for an attack detection approach which is capable of detecting DDoS attacks on HTTP protocol in its infancy.

## 4. Proposed Solution

Users Flow and Behaviour details are equally important for detection of Application Layer DDoS attack. To detect HTTP GET, HTTP POST, and HTTP slow read attack a hybrid clustering based correlation attack detection approach is proposed. It mainly consists of two parts a "User Behaviour Monitor" system and "Flow Monitor" system.

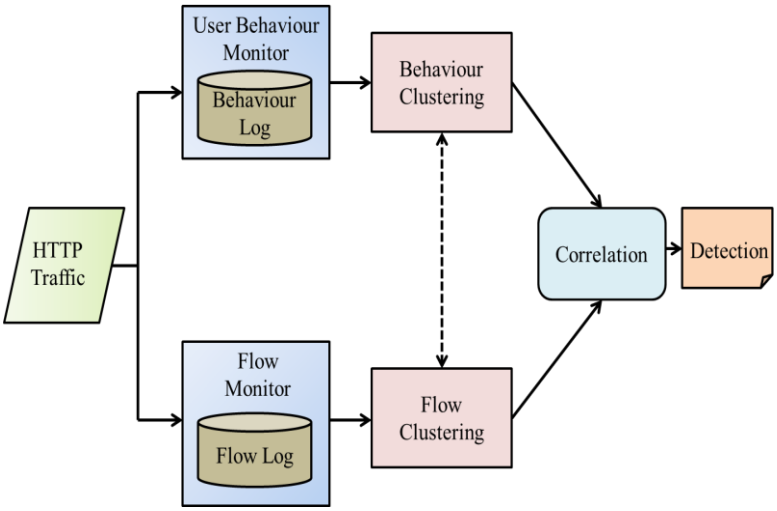


**Figure 1. HTTP Ddos Detection Approach Developed**

User Behaviour monitoring system stores information about activity of a particular host on a website. Flow Monitoring system takes care of data flow related information like IP address, port number *etc*. Clustering algorithm is applied to both behaviour and flow logs to separate attack traffic from normal traffic. Finally flow and behaviour clusters formed are correlated to detect attacking hosts. Proposed approach for attack detection is shown in Figure 1.

### 4.1. User Behaviour Monitoring Module

User Behaviour Monitor is a program written in PHP which logs the entire request going to a particular webpage. It generates user "Behaviour Log" file containing information about user's activities. It logs the activity of the user.

It monitors and logs details like pages accessed, request type, content type, content size, user agent and response code. These details are used to analyse user browsing behaviour. It is important to note that user behaviour monitor alone is not sufficient to detect application layer DDoS attack on HTTP protocol. Flow information is also required for proper attack detection.

 

### 4.2. Flow Monitoring Module

The Flow Monitor logs flow information about communicating entities. It is able to detect data flow information like IP address of remote host communicating, request time and date, remote host port number, server port accessed and duration of each request. Flow monitor logs flow information in "Flow Log" file. Similar to Behaviour Monitor, Flow Monitor is also a PHP program that gets executed for every request to a webpage.

### 4.3. Behaviour Clustering

Page access matrix for each unique user is constructed to analyse user behaviour. IP along with Date is used as unique user. Each unique user is represented as a transaction, where each transaction denotes number of request to a particular page during a visit. Consider a web page with n pages and m unique users visited the website. Page access matrix is shown in Figure 2.

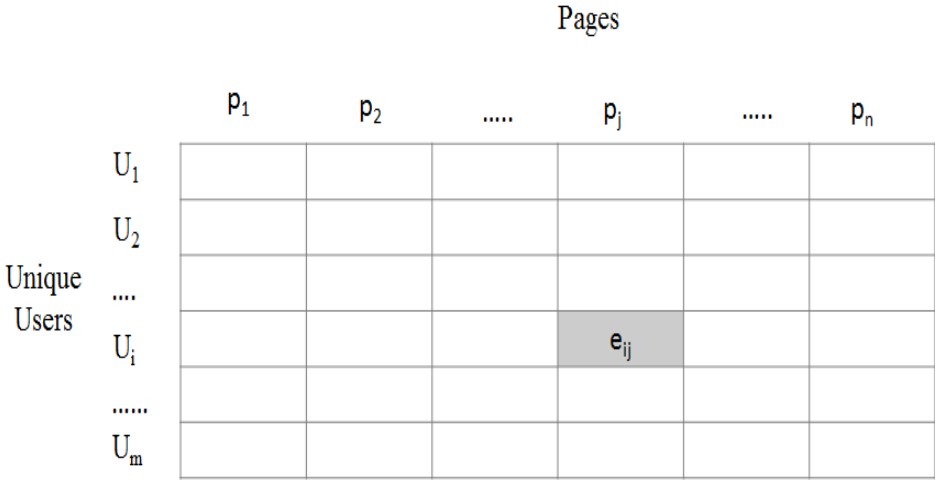


**Figure 2. Page Access Matrix Showing Users Access to Different Pages**

The cell value $e_{ij}$ represents a weight associated with contribution of a particular page for a unique user. Number of request of each unique user on a particular webpage is taken as weight. Normalization is performed in page view space to avoid the influence caused by relative amount difference in request number for a unique user.

Once the page access matrix is created clustering algorithm is used to divide unique users to two different clusters- Attack and Normal user. Within each cluster users may share same interest. Similarity based measure is used to determine how close two unique users are in a particular cluster. Cosine similarity is applied to identify similarity between two unique users. Suppose U1 and U2 are two unique users then cosine similarity between them is denoted by the equation.

$$\text{Sim (U1, U2)} = \frac{U1.U2}{||U1||||U2||}$$

Clustering algorithm is applied to identify clusters during behaviour clustering. Users with similar behaviour are grouped into same cluster. K-Means, K-Medoid and Agglomerative clustering are applied on the user behaviour tuples. Even though agglomerative clustering gives higher detection rate, K-Means clustering is given priority in this paper as it is suitable for large datasets. Time complexity of agglomerative clustering algorithm for large dataset is much higher compared to that of K-Means clustering. Pseudo-code for K-Means clustering algorithm is as follows.

*Algorithm:* K- means clustering
*Input:* User access matrix
*Output:* Two clusters C1 and C2
*Method:*
Let C1 and C2 be two clusters and Cid1 and Cid2 be centroids respectively.
begin

i. Choose first two unique host U1 and U2 as two clusters C1, C2 and the centroid of respective cluster, *i.e.*, C1 = {U1}, Cid1 =U1 and C2 = {U2}, Cid2 =U2

ii. For each unique host Ui calculate the similarity between Ui and centroid of existing two clusters, *i.e.*, Sim (Ui, Cidj). Where j can be 1 or 2.

iii. If Sim (Ui, Cid1) > Sim (Ui, Cid2), allocate Ui to Cid1 else assign Ui to Cid2. If Ui is reallocated to a cluster Cj, recalculate centroid of the cluster Cj as Cidj $=1/|Cj| \sum_{i \in Cj} Ui$

iv. Repeat step (ii) and (iii) until all unique hosts are processed and all centroids do not update any more.

end begin

### 4.4. Flow Clustering

Main objective of flow clustering is to divide users into groups with similar flow behaviour. In flow clustering unique user is identified using IP and date. In order to apply flow clustering flow information is to be translated to suitable vector representation. Statistical features which are extracted from flow details and translated into vector representation are

- *Average request duration (ard)*, is calculated by summing time duration of requests divided by total number of request.
- *Request Rate (rr)*, is calculated by finding number of request per day.
- *Average bytes transferred (abt)*, is calculated as total number of byte transferred in average request duration.

Each unique user Ui can be represented in vector form as Ui = {ard, rr, abt}. After constructing the vector representation of flow data, clustering algorithms is applied to group users into different clusters. Within each cluster a user share same flow features. Vectors which are close to each other in vector space represent users with same communication pattern. In order to identify the users that share similar properties K-Means clustering algorithm is applied. K-Medoid and Agglomerative clustering algorithms are also applied to check their effectiveness.

### 4.5. Cross Cluster Correlation

Four different clusters are formed after applying clustering algorithm, two each from flow and behaviour logs. Two clusters need to be find which contains the attacking host IP's. One each from flow and behaviour clusters formed. In other words, attack clusters from both flow and behaviour clusters need to be correlated.

In order to identify attacking user's clusters, an attack score is assigned to each flow and behaviour features. Scores are having the range 1 to 10. Higher the score implies more chance of attack. Min-Max normalization is used to convert each feature values to the attack score range 1 to 10. Let X is the value of feature Fi which is to be normalized to the range [1, 10]. Formula for calculating attack score is given below.

 

$$\text{Score } (F_i) = \frac{(X - Xmin)}{(Xmax - Xmin)} * (10-1) + 1$$

Here Xmin represents minimum value of feature Fi and Xmax represents maximum value of feature Fi. Individual score of each cluster is to be calculated initially to compare the scores. Flow clustering divides the flow log into two clusters, one containing attack user's details and the other with genuine users. Three features are used for flow clustering, *i.e*., avg request rate (F1), avg byte transferred (F2) and avg request duration (F3). In order to calculate the combined score of all the three features, initially calculate individual score of each feature Fi and combine the score to get the final score.

Let score (F1), score (F2) and score (F3) be the scores of three features F1, F2 and F3. Final score of a user Ui is given by the formula

$$\text{FinalScore } (Ui) = 0.4*(\text{score } (F1)) + 0.3* (\text{score } (F2)) + 0.3*(\text{score } (F3))$$

Here score (F1) denotes the score of user Ui for the feature avg request rate. Average request rate is given priority (multiplies by 0.4) over other two features because all the HTTP attacks have it as a common factor. After computing individual score of each user Ui in the two flow clusters, next is to find the combined score of each cluster. Let CScore (Ci) represent combined score of cluster Ci. If U1, U2, U3… Un is the users in cluster Ci, then

$$\text{CScore } (C_i) = \frac{\sum_{i=1}^{n} FinalScore\ (Ui)}{n}$$

Similarly scores of two clusters formed by the behaviour clustering is also computed. In behaviour clustering number of request form a unique user to a particular page in the website is considered for clustering. Pages P1, P2, P3, P4, P5 and P6 of the website is considered as features B1, B2, B3, B4, B5 and B6 for behaviour clustering. Min-Max normalization is applied as in previous case to find the score of each feature Bi. Let score (Bi) denotes score of feature Bi for a user Hi. Final score of a unique user is calculated by taking the average of score of each feature Bi, where i = 1 to 6. Ie

$$\text{FinalScore } (H_i) = \frac{\sum_{i=1}^{6} score(Bi)}{6}$$

Combined cluster score of both the clusters formed after behaviour cluster are calculated by summing scores of each user in the cluster divided by the number of unique users in each cluster. Let CScore (Ck) represents combined behaviour cluster score of Cluster K.

$$\text{CScore } (C_k) = \frac{\sum_{i=1}^{n} FinalScore\ (Hi)}{n}$$

Finally, to identify the two attack clusters among the four clusters formed, two clusters having maximum Combined Cluster Score (CScore) one each from the flow and behaviour clusters are taken. In other words, cluster having maximum attack score in flow and behaviour clusters represent attack users.

## 5. Environment Setup

Environmental setup for application layer attack detection includes a website hosted on Apache server and systems infected with 4 different HTTP DoS attacking software's. DoS website developed for experiment contains six different pages say P1, P2, P3, P4, P5 and P6.

The following software's are explored to launch application layer DDoS attack.

- Slowhttptest open source software, which is explored to perform application layer DDoS attack on http protocol in test bed environment
- Slowloris, a Perl script to launch HTTP-GET attacks
- Pyloris, a Python script similar to slowloris in functionality
- Bot, a .NET program for application layer DDoS attack

Systems infected with slowhttptest software can perform -- HTTP-GET, HTTP Slow read and HTTP-POST attacks. Slowloris and Pyloris are two other tools which are explored to launch HTTP-GET attack. A bot program written in .NET is also developed for performing HTTP-GET attack. .NET bot program launching HTTP-GET attack is shown in Figure 3.


**Figure 3. Dot NET Bot Program Sending Multiple HTTP Request**

## 6. Result

Flow clustering helps to detect majority of the attackers. It divides unique users to two clusters in which one cluster mainly contains attackers and the other cluster with both attacking and non-attacking hosts. On the other hand, Behaviour clustering discovers attacking IP which are not identified during Flow clustering. Similar to Flow Clustering Behavior clustering divides users to two clusters in which one cluster contains only attacking IP's. After correlating the four different clusters using attack scoring techniques, it is possible to identify most of the application layer DDoS attack performing IP's with minimal false positive rate.

### 6.1. Result Evaluation

DATASET generated for evaluating the proposed detection approach is shown in Table 1. Details like Number of instances and attributes of Normal and Attack DATASET is shown in Table 1.

 

**Table 1. Normal and Attack DATASET**

| *DATASET* | *Number of instances* | *Number of attributes* |
|---|---|---|
| **Normal user flow dataset** | 486 | 3 |
| **Normal user behaviour dataset** | 486 | 6 |
| **Attacker flow dataset** | 575 | 3 |
| **Attacker behaviour dataset** | 575 | 6 |

Number of Attack instances detected after applying K-Medoid, Agglomerative and K – Means clustering algorithm is shown in the Table 2.

**Table 2. Attack Instances Detected after Applying Clustering Algorithms**

| Attack Type | Number of Attack Instances Detected | | | Number of Attack Instances Undetected | | |
|---|---|---|---|---|---|---|
| | K-Medoid | Agglomerative | K-Means | K-Medoid | Agglomerative | K-Means |
| **HTTP-GET** | 454 | 490 | 466 | 46 | 10 | 34 |
| **HTTP-** | 340 | 340 | 330 | 0 | 0 | 10 |
| **HTTP-Slow** | 272 | 302 | 284 | 38 | 8 | 26 |

It is observed detection of slow read attack is lesser compared to HTTP-GET and HTTP-POST attacks for all clustering algorithms applied. It is also observed that agglomerative clustering shows high attack detection compared to K-Means and K-Medoid. Percentage of Attack instances detected during detection process is shown in Figure 4.

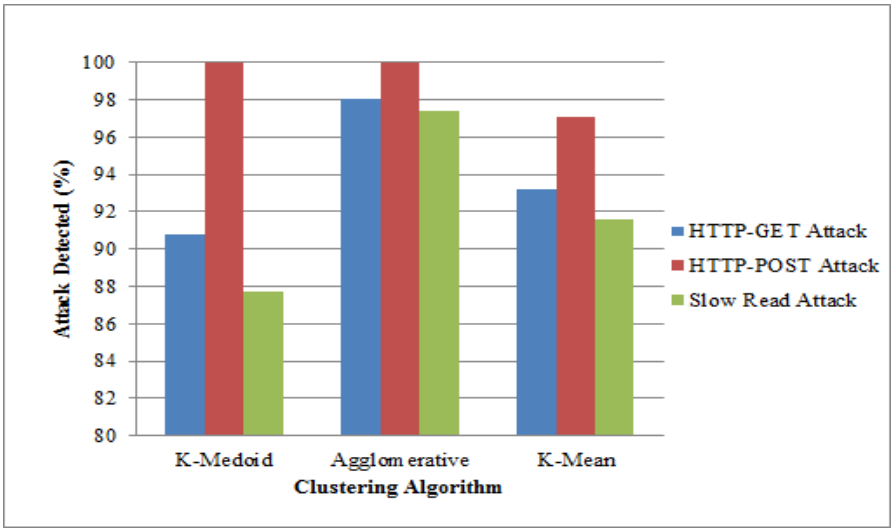


**Figure 4. Attack Instances Detected Percentage**

### 6.2. Comparative Analysis

Detection Rate (DR) and Accuracy (A) are the two parameters used for evaluating the proposed DDoS detection approach.

1. Detection Rate (DR): Detection rate is defined as number of attacking users identified among total attacking users.
2. Accuracy (A): Accuracy shows how much precise or correct the result is.

To compare the effectiveness of developed approach clustering algorithms are applied on the flow and user behaviour tuples and their results are analysed. Results from applied clustering algorithms are shown in Table 3. It is observed that performance of K-Mean and K-Medoid algorithms are almost same. Both these algorithms have similar Detection Rate and attack detection Accuracy.

**Table 3. Results From Clustering Algorithms**

| Clustering Algorithm | True Positive (TP) | False Negative (FN) | True Negative (TN) | False Positive (FP) |
|---|---|---|---|---|
| **K-Medoid** | 1066 | 84 | 820 | 152 |
| **Agglomerative** | 1132 | 18 | 920 | 52 |
| **K-Means** | 1080 | 70 | 828 | 144 |

Hierarchical clustering method (Agglomerative clustering) is found to have highest attack detection rate among all the clustering methods analysed. It is also observed that separate groups of three HTTP attacks are identified during agglomerative clustering. Other clustering algorithms like DBSCAN and Expectation Maximization clustering are also applied which gives clusters of poor quality. Attack Detection Rate and Accuracy of applied clustering algorithm are shown in Figure 5.

Even though agglomerative clustering algorithm gives high detection rate for the generated dataset, K-Means algorithm is better for larger dataset. Running time of K-Means clustering algorithm is linear *i.e.*, O(n). Time complexity of agglomerative algorithm is quadratic *i.e.*, O ($n^2\log(n)$) which increases exponentially for large datasets.

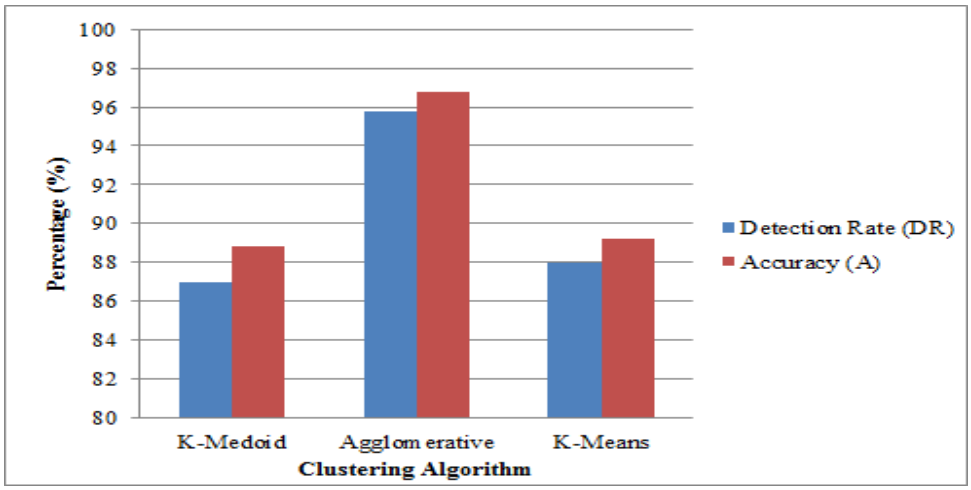


**Figure 5. Detection Rate and Accuracy of Clustering Algorithms**

## 7. Conclusion

In this paper an approach to detect Application Layer DDoS attacks like HTTP-GET, HTTP-POST and Slow Read attack is developed. These attacks are difficult to detect as

they resemble genuine user request. As they resemble genuine request, along with flow information user behaviour is also analysed for attack detection.

For better understanding two distinct log files are maintained during attack detection process. Clustering algorithm is applied to group users having similar characteristics. Correlation is then applied on the groups formed to detect attackers.

Developed approach is capable to detect HTTP-GET, HTTP POST and also slow read attack which was left behind by many other DDoS detection mechanisms. The proposed approach is also capable of detecting emerging DDoS attacks that uses mixture of Layer-3 and Layer-7 DDoS attacking techniques.

Further work can be focused on wider aspects of experiment like detection of Flash crowd. Flash crowd detection can be done by using proposed approach with little modification in parameters used in behaviour clustering. Attack categorization can also be done in enhanced detection approach to precisely identify type of attack that causes DDoS.

## References

[1] A. Kumar and P. S. Tilagam, "A Novel Approach for Evaluating and Detecting Low Rate SIP Flooding Attack", International Journal of Computer Applications, vol. 26, no. 1, **(2011)**, pp. 31.

[2] B. Prabadevi and N. Jeyanthi, "Distributed Denial of service attacks and its effects on Cloud environment- a survey", International Symposium on Networks, Computers and Communications, Hammamet, **(2014)**.

[3] C. Ye, K. Zheng and C. She, "Application layer DDoS detection using clustering analysis", 2nd International Conference on Computer Science and Network Technology (ICCSNT), Changchun, **(2012)**.

[4] J. Park, K. Iwai, H. Tanaka and T. Kurokawa, "Analysis of Slow Read DoS attack", International Symposium on Information Theory and its Applications (ISITA), Melbourne, VIC, **(2014)**.

[5] J. Yu, Z. Li, H. Chen and X. Chen, "A Detection and Offense Mechanism to Defend Against Application Layer DDoS Attacks", Third International Conference on Networking and Services, Athens, **(2007)**.

[6] S. Wen, W. Zhou and W. Zhou, "CALD: Surviving Various Application-Layer DDoS Attacks That Mimic Flash Crowd", 4th International Conference on Network and System Security (NSS), Melbourne, VIC, **(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, vol. 17, no. 1, **(2008)**, pp. 26-39.

[8] S. Renuka Devi and P. Yogesh, "An effective approach to counter application layer DDoS attacks", Third International Conference on Computing Communication & Networking Technologies (ICCCNT), Coimbatore, **(2012)**.

[9] S. Sivabalan and P. J. Radcliffe, "A novel framework to detect and block DDoS attack at the application layer", IEEE TENCON Spring Conference, Sydney, NSW, **(2013)**.

[10] Y. S. Choi, J. T. Oh, J. S. Jang and I. K. Kim, "Timeslot Monitoring Model for application layer DDoS attack detection", 6th International Conference on Computer Sciences and Convergence Information(ICCIT), Seogwipo, **(2011)**.

 