PROCEEDINGS OF SCIENCE

# Detection for Application-layer Denial of Service Attack Based on Cluster Analysis

**Xiaowei Jing**[1]
*Tsinghua University, Beijing, 100084, China*
*E-mail:* jingxw@tsinghua.edu.cn

**Dongming Bai**
*Research Institute of Petroleum Exploration & Development, Beijing, 100083, China*
*E-mail:* baidm@petrocihna.com.cn

**Mei Feng**
*Research Institute of Petroleum Exploration & Development,Beijing, 100083, China*
*E-mail:* fm@petrocihna.com.cn

**Liang Chen**
*Research Institute of Petroleum Exploration & Development,Beijing, 100083, China*
*E-mail:* chen_l@petrocihna.com.cn

**Youjian Zhao**
*Tsinghua University,Beijing, 100084, China*
*E-mail:* zhaoyoujian@tsinghua.edu.cn

Denial of service attack is evolving from the network layer to the application layer in recent years. Their characteristics show that attack in the application layer is more secret with less data traffic compared with that in the network layer, and it's more difficult to defend this kind of attack, so we propose the model of application-oriented detection for denial of service attack which involves an aggregation algorithm for application partition, monitoring running status of application system, detecting abnormal information, and assessing threat level of malicious access source. Through modelling and verification of actual problem, we expound upon the effectiveness and practicality of this model.

*ISCC 2015*
*18-19, December, 2015*
*Guangzhou, China*

[1]Speaker & Corresponding Author





PoS(ISCC2015)032

## 1.Introduction

Distributed Denial of Service (DDoS) attack has always been a ubiquitous security threat in the network. Attackers use zombie host, network flow, system bugs, resource costs, etc., to make the target host respond slowly, reduce its service quality, and even make it stop providing services, or unable to handle normal requests of legitimate users. From the data of the DDoS attacks in recent years, we can see that the application-layer DDoS attack grows more popular, which is dominated by the mode of attack with a low rate, high resource consumption by imitating the normal user behavior. Such attacks will not produce a large number of abnormal flows and are also highly secret, which pose great challenges to the traditional DDoS protection.

This paper firstly expounds upon the characteristics of the new attack type, and the shortcoming of the traditional defense. Through the analysis of and reference to the idea of the detection method of the attack mainstream and to make up for its inadequacy, this paper summarizes the clustering analysis algorithm that uses response delay to detect application-layer DDoS attacks and distinguishes the applications, and compares it with the effect of the overall application detection.

## 2.Threat Brought by the Evolution of DDoS

### 2.1 Attack Evolution Trend

DDoS attacks are mainly divided into two categories: network layer attacks and application layer attacks. The network layer DDoS attacks mainly act on the network layer, and such attacks often use the protocol loophole and working characteristics of the network layer or the transport layer to attack the target host. Typical network layer DDoS attack iuses the attack node of the forged IP address to send a large number of attack groups to the target host. Compared with network layer DDoS attacks, application-layer DDoS attacks adopt real IP address for attack and establish TCP connection with the host. In addition, the application layer protocol is more complicated, such as http protocol, in which a request statement may cost a large amount of data of the server, such as interception and replay of normal access data packets of users, access to the target site in a recursive way to realize the effect similar to the crawler download in the search engine, and access to the query function or downloading large quantity of files with elaborate design requests to make the attacked host consume resources. These attack message formats are exactly the same as the data packet format of normal users, without abnormal packet or abnormal field values, which makes it difficult for fthe irewall and other security devices to detect attacks (Fig. 1).

Since 2011, application layer attacks have shown a trend of rise in the proportion of DDoS attacks. Compared with the network layer attack, the application-layer DDoS attack is more advantageous for attackers (Table 1).

| | Network layer DDoS attacks | Application-layer DDoS attacks |
|---|---|---|
| Attack traffic | Relatively large | Relatively small |
| Secret | Low | High |
| Main phenomena | Network bandwidth exhausts<br>Network or security equipment overloads<br>Performance of server reduces | Network bandwidth exhausts<br>Performance of server reduces |
| Depth of interaction | Mainly act on the low layer networking protocol | Mainly act on the high layer application protocol |
| Effect degree | Gradually reduce from the network layer to the application layer | Gradually rise from the network layer to the application layer |

**Table 1:** Comparison of DDoS Attacks

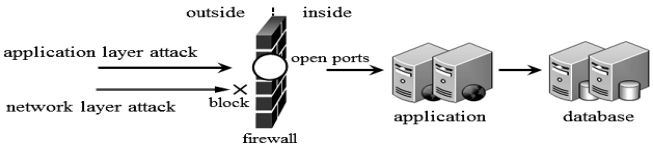



**Figure 1:** Application-layer DDoS Attack Flow Penetrates the Firewall

### 2.2 Characteristics of Application Layer Attacks

Nowadays, twebsite pages are abundant in embedded resources, such as pictures, flash, and dynamic content, which are often provided by multiple background servers. Looking from analysis results of access capture package of Taobao's home page, we can see that there are as many as 27 servers responding to a request of the user, among which, 1 server has the ratio of request to response of more than 1:44. Application layer attacks are great in load distribution for servers (as shown in Fig. 2). In addition, the application-layer DDoS attack can use lightweight access message to pass through the network protocol and port that must be open for the application system to provide services, so as to attack the target system, reduce the service quality of application system, and even make the server stop.

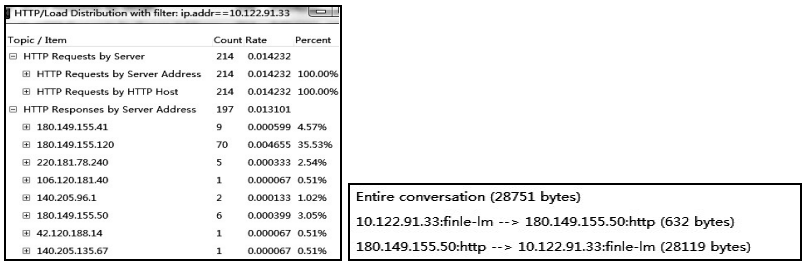



**Figure 2:** Load Distribution of Server of the Tabao's Home Page

### 2.3 Deficiency of Traditional Defense

At present, there are three major measures to defend against DDoS attacks:

(1) Access control: access control equipment detects authenticity of the source address of the network request, discards the network requests with fake source addresses, identifies attacks and blocks them based on network behavior characteristics; (2) Traffic cleaning: analyzes network data flow and leads the network traffic with attack characteristics to the cleanup center so as to filter the attack traffic, keep the normal flow and restore it to the original network; (3) Intrusion prevention: compares the network data flow with defined characteristics, and cuts off the attack traffic in line with characteristics of DDoS.

Based on the above techniques, use relatively mature security products of this industry, deploy them in key nodes in the network for monitoring network traffic and its corresponding characteristics, and associate the server IP address with the application system according to the deployment strategy to alarm the abnormal situation. However in practice, following problems exist:

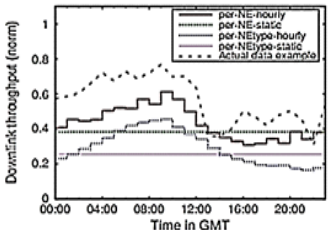
(1) security equipment price and operation maintenance cost are higher; (2) device configuration is not flexible, and configuration workload is larger; (3) the network traffic is mainly monitored, which can provide the basis only for the application anomaly at the network level; (4) although it can be associated with the application system, we still cannot know which applications are attacked in Web services, so the alarm value is smaller.

## 3.Popular Detection Method

In terms of the anomaly detection, a lot of researches and applications are carried out in the foreign country, in which, the average threshold, Time Series Decomposition, Holt Winters and the cluster analysis model are tmost representative.

### 3.1 The Average Threshold

The average threshold method[1] is an anomaly detection algorithm that uses the historical average and standard deviation to calculate the threshold. It realizes the compression algorithm complexity in space and time through the tradeoff among a variety of methods, and tries to ensure a low false alarm rate and misreporting rate. Fig. 3 shows the example of using this method for the automatic anomaly detection of a 3G telecommunications network operation and maintenance data, but the selected anomaly detection algorithm is relatively simple. Upper and lower thresholds of a monitoring point are calculated by the $T = \mu - c \times \sigma, T = \mu + c \times \sigma$ formula where μ is the expectation for the historical data, reflecting the average level of current monitoring points; σ is the standard deviation, reflecting the normal fluctuation range of monitoring points; c is the weight which can be adjusted according to the demand. This method compresses the complexity in the time and space mainly through the granularity of the calculated threshold so that it can complete the application in specific scenarios, but it does not have universality. For example, as for the monitoring index with a complicated and diversified nature, the result of this algorithm has low accuracy so we need to establish an automatic threshold recognition and selection algorithm.



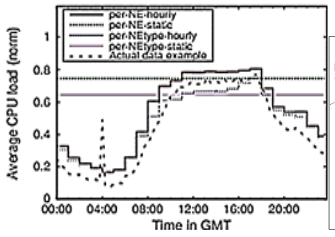




**Figure 3:** Results Comparison Under Threshold of Different Size

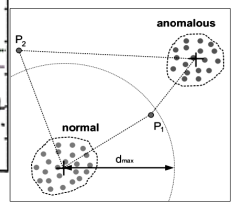



**Figure 4:** Determination of Anomalous Data

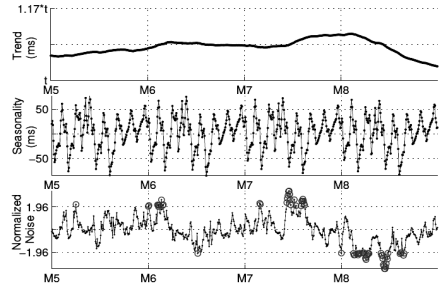



**Figure 5:** Time Series Decomposition Results and the Anomaly Detection Specific to Noise Terms

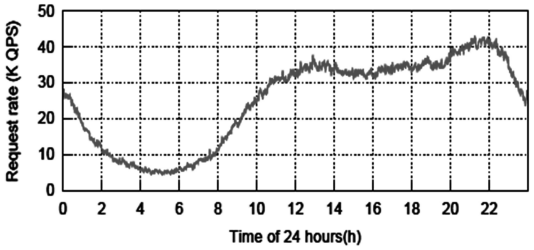



**Figure 6:** Overall Request Rate Within One Day

K-means modle[2] clusters the weighted distance formula of data points with concentrated learning data into normal and abnormal categories according to their different KPIs (Key

Performance Indicator, such as BPS, PPS, origin-destination IP address number and other performance indexes):

$$d(x,y)=\sqrt{\sum_{i=1}^{m}\left(\frac{x_i-y_i}{s_i}\right)^2}$$

(3.1)

Specific anomaly detection method is shown in Fig. 4, in which, points away from the normal group and exceeding the threshold value dmax or close to the abnormal group are anomalous data.

### 3.3 Time Series Decomposition

Time Series Decomposition[3] decomposes each time series data point into the trend term, periodic term and noise term, and uses different statistical methods for the anomaly detection specific to different components. Each data point in the response latency timing data is decomposed into the trend term (Lt), period term (St), noise term (Nt) through Time Series Decomposition. Decomposition results are shown in Fig. 5.

$$L_t=\frac{1}{T+1}\sum_{i=-T/2}^{T/2}SRT_{t+i},S_t=\frac{1}{M+1}\sum_{i=0}^{M}Y_{t-iT},M=\lfloor t/T\rfloor,N_t=SRT_t-L_t-S_t$$

(3.2)

In view of the noise term (Nt) that establishes a Gaussian distribution model per hour in a certain period (seven days), we consider that points with confidentiality of over 95% are anomalous data, i.e., $\frac{|N_t-\mu_t|}{\sigma_i}>1.96$.

### 3.4 Holt Winters

Holt Winters[2,4,5] predicts the future value based on the historical RTT (Round Trip Time) time-series data through the exponential moving average and takes into account the periodicity and trendy, in which, data with a large difference between the actual value and the predicted value are detected as anomaly. Anomaly filtering mechanism is designed to ensure that abnormal data will not have an impact on the prediction. Its basic principle is that when the difference between the actual and predicted value is too large, the predictive value of the last window should be used as the predicted value.

The above anomaly detection models carry out the detection specific to certain characteristics of tdata, such as timeliness, periodicity and volatility, which cannot achieve universal detection effect for the vast diversity of monitoring data of large Internet content provider. Table 2 summarizes the anomaly detection algorithm.

| | Implementation approach |
|---|---|
| The mean - threshold | The rational range of the current value (mean and variance) is obtained according to historical records, and the one beyond the range is determined to be the anomalous. |
| Time series decomposition | Decomposition of the composition of the current values (trend, period and noise) is conducted, and then anomaly detection is performed specific to different components, such as noise. |
| Cluster analysis | Cluster the data according to the distance between the data points to determine the normal and abnormal data groups, and take this as the basis for anomaly detection. |
| Holt Winters | Predicts the next value, and carries out the anomaly detection according to the actual and the predicted value. |

**Table 2:** Comparison of Detection Methods

In addition, there are some other detection methods: use the moving average method to carry out the anomaly detection on P2P behavior[6]; carry out the high-speed KPI sampling through high-speed Web service[7], and then find the system bottleneck points by drawing the request rate and system throughput graph, so as to detect the instantaneous anomalies of performance; use the corresponding indexes in TCP protocol stack such as number of lost packets, fast retransmit and so on to identify the TCP performance degradation, and locate the anomaly position through the classification of abnormal indexes and the correlation analysis of abnormal TCP streams[8]. Literature 9 make a summary of major anomaly detection methods[9].

The above anomaly detection methods are applied in a particular scenario. In the DDoS attack detection of HTTP service, the main measure of system service situation is the response latency, since service latency will rise sharply after the attack. In detecting the amount of service requests, it will inevitably produce false positives, the speed limit and filtering policy caused by which will block part of the user's requests. Therefore, to avoid the bad impact on normal users to the minimum extent, monitoring of the response latency, a service quality evaluation index in this paper, will not trigger anomaly alarm in the case that the response performance meets the performance requirements of the service, i.e., it will tolerate the attack of small data traffic that does not cause the system overload. After the response latency exceeds the threshold requirement, the attack anomaly alarm will be triggered and a statistical analysis of request data traffic during the anomaly will be made to analyze the source of attack.

## 4. DDoS Attack Detection Model Based on Cluster

To understand the real Web access and solve model design problems, we collect and analyze the Web access data from the enterprise data center. The data center hosts more than 200 applications, and the peak of the overall request rate (Fig. 6) can reach 42,000 queries per second (QPS). This paper focuses on the analysis of applications with the traffic ranking top 64.

By measuring the maximum latency of overall granularity in the data set, the average latency, and the 90-percentile latency (latency per unit of time ranking 90th from the small to large), we find that both the 90-percentile latency and average latency are within 0.25s (Fig. 7), which indicates that the service quality is in good condition. As the maximum latency jitters greatly with many interference factors, it is not used as a service quality evaluation index. However, users complain that a few applications are particularly high in latency at some period of time, but such abnormality cannot be expressed in the monitoring of overall granularity.

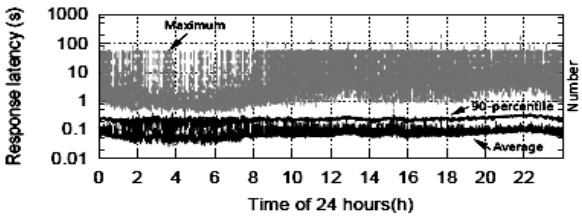



**Figure 7:** Overall Response Latency Within One Day

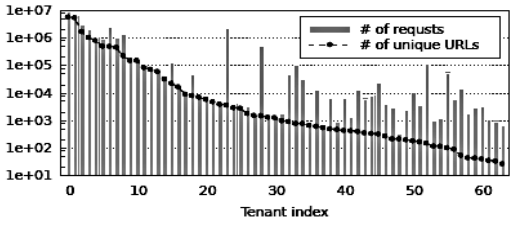



**Figure 8:** Number of Requests and Unique URLs in Requests of 64 Web Services

This further proves that the overall granularity detection is not very accurate, especially for the application of low flow, whose abnormity is almost impossible to detect.

### 4.1 Data Analysis

(1) Most Web services have a large number of unique URLs.

The number of requests of 64 Web services ranges from 4 million to 20 million, and in about a quarter of Web service requests, 90% has different unique URLs (as shown in Fig. 8) in the data set. It is mainly because the URL encoding mode has the parameter field (parameters here are not confined to ones behind the "?" at the end of the URL, and the field of URL itself may also contain parameters, such as /news/0001), which makes each requested URL slightly different.

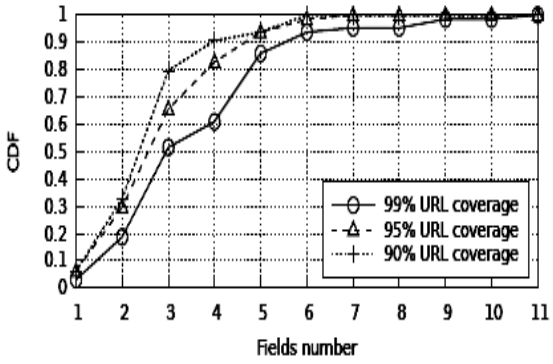



**Figure 9:** Cumulative Distribution of URL Coverage in the Same Common Fields

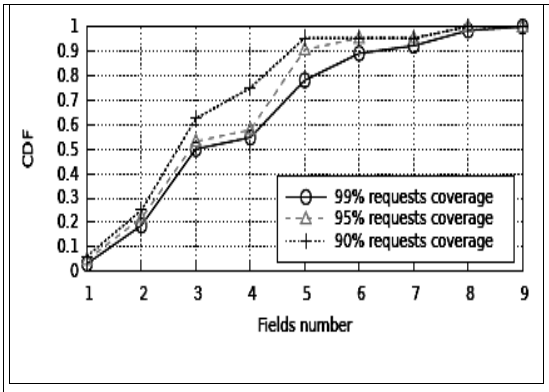



**Figure 10:** Cumulative Distribution of URL Coverage in the Same Common Fields

(2) URL fields that can be enumerated

For 64 Web services, 99% of the URLs have 11 URL fields at most (i.e., character string separated by "/" in the URL path), of which, 61 Web services contain 99% of their URLs with only 7 fields. Those fields with a very small number of URLs are simple index pages. In addition, requests are more inclined to the URLs with field number less than 10. For all 64 Web services, URLs of less than or equal to 9 fields receive 99% of the requests (as shown in Fig. 9)

For the rest of 57 Web services, only six fields can cover their 99% requests (Fig. 10). The result shows that regardless of the wide URL values in each field and the large number of overall URLs, only a small number of the same common fields of URL can be enumerated.

(3) Different application system latency time

Latency time is a very important indicator in detecting the running status of the application system. Through monitoring, we find that each application system is different in peak access hour and response latency time. The access latency of business application systems is generally higher than that of web portal systems (as shown in Fig. 11 and Fig. 12). Therefore, it is very important to set up different latency points for different application systems to assess anomalies.

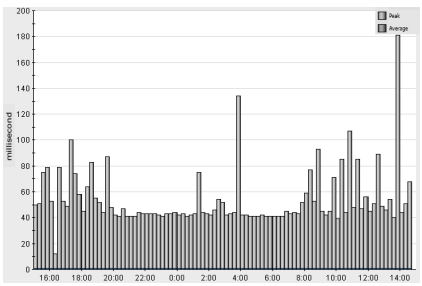



**Figure 11:** Latency of Portal System

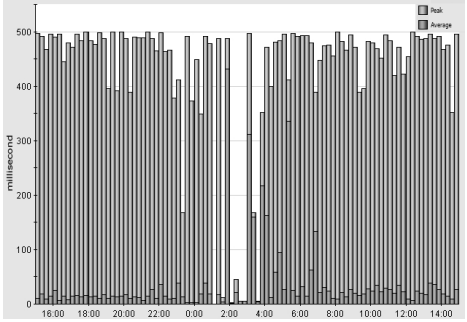



**Figure 12:** Latency of a Certain Business Application

By tracking, we find that larger values in system response latency are generally caused by the executing of complex logic, I/O operations, normal operations of data backup and the batch updates by some application systems, but when the system normally runs, these values are relatively smaller in 90-percentile latency per unit time; when the system is attacked, they will become significantly larger, which represents the overall service quality.

### 4.2 Clustering Algorithm

Traditional differentiation of data traffic is carried out mainly based on the characteristics of the network layer and transport layer, such as IP and port. This method does not have visibility to upper layer applications, which is unable to subdivide the different applications of the same port. Especially in today when HTTP protocol is popular, a large number of systems are built based on the HTTP protocol, and traditional methods cannot make a service differentiation specific to the application layer. Therefore, the fine-grained application-layer detection is established here to discover its response anomalies.

According to the data analysis, there are many unique URLs in most Web applications because URL contains a parameter field. Moreover, the number of field in URL can be enumerated. In this case, it's necessary to cluster the URL of the application system, to eliminate the influence of the parameter field and distinguish between different applications. At the same time, to meet the different needs for the detection of different application systems, changeable parameters should be considered as configurable parts and withdrawn as the input of the model. For example, the latency time for anomaly detection varies according to different application systems, and is not necessary 90-percentile latency.

Basic concept of the clustering algorithm is as follows:

(1) X is an element in the data set, represented by a vector such as {x1,x2,…,xn};

(2) D (x, y) represents the distance between elements x and y in the set;

(3) LC (X, Y) represents the distance between clusters X and Y (such as the minimum or maximum distance between X and Y, etc.).

In this way, all the elements are individually aggregated into different clusters according to the clustering strategy (such as the minimum distance).

As to the URL clustering problem, each URL u has a set of fields. If the ending field has a "?", then the parameter content behind the question mark will be filtered out. In this way, u can be expressed by a vector F (U) = {f1, f2, ..., fi}, where the subscript of each field is its position in the u. For example: "/forum/userInfo.html? 16" is represented as {f1 = forum, f2 = userInfo.html?}.

The distance between URLs can be expressed by the number of public fields, so that the element distance function D can be expressed as:

$$D(u_1,u_2)=\begin{Bmatrix} \infty & ,if\ F(u_1)\cap F(u_2)=\varnothing \\ \dfrac{1}{|F(u_1)\cap F(u_2)|} & ,otherwise \end{Bmatrix} \tag{4.1}$$

where the distance function D represents that the more public fields in two URLs are, the closer the distance between two URL's is. Level of Cluster refers to the URL hierarchy where the cluster nodes lie in, denoted by L(V). The distance between any two members of L(V) is less than or equal to L/L(V). The clustering algorithm uses the similarity and hierarchy of the URL structure to cluster while excluding noise parameters, i.e., uses the similarity to cluster the public fields of URL, and uses the hierarchy to identify the position of the cluster. Cluster nodes

are divided into three categories: (1) Virtual Cluster, intermediate result in the process of clustering; (2) final result after the completion of clustering; (3) the root node that does not deposit F(U), and just acts as the father node of all clusters under the condition that L(v) = 1. As a result, the messy URL, after the URL data being subject to clustering, will eventually cluster into a tree structure composed of three kinds of aggregation node.

When a URL arrives, it creates a Virtual Cluster in each level and uses F (URL) for its initialization. Virtual Cluster is represented by the public field of its members, referred to as F(V), where $|F(V)| \geq L(V)$. The maximum value L of the level is a parameter, which limits the maximum distance between clusters. According to the enumerability of URL fields, it can be known that L is a very small value. At the same time, only Virtual Clusters that have the same ancestor node can be merged, where the cluster distance function LC and merger conditions are as follows:

$$LC(V_1, V_2) = D(V_1, V_2)$$
$$D(V_1, V_2) \leq \frac{1}{l}, where\ l(V_1) = l(V_2) = l \tag{4.2}$$

The concrete merging example is shown in Fig. 13, in which, /a/b/c and /x/y/2 do not cluster together, and when /a/e arrive, it only merges with /a/b/c at level 1.

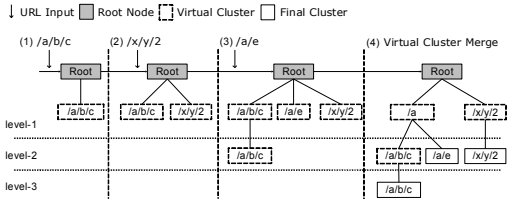



**Figure 13:** URL Clustering Process

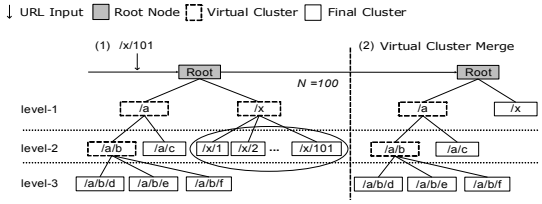



**Figure 14:** URL Clustering Process

We have got all the Virtual Clusters, recorded all the clustering processes, and now need a stop condition to determine which clusters are the Final Clusters. The generation of the Final Cluster does not need to wait for the completion of generation of all the Virtual Clusters. The generation processes of both are parallel. The cluster on the leaf node is Final Cluster.

The number of subclasses produced by different URL patterns is different, among which, the number of subclasses with parameter field is larger. To filter out the parameters interference, a threshold value N is set in the algorithm, which is used to decide which Final Clusters need to be further merged. When the number of subclasses of a Cluster is more than N, the Cluster will be marked as the Final Cluster, and its child elements will be eliminated (as shown in Fig. 14).

```
Algorithm uCluster(URL u, L, N)
current = root
for d = 1 to min(L,|F(u)|) do
    merge_flag = false
    if current.type <> FINAL_CLUSTER then
        for each CLUSTER c in current DESC by his
        do
            if |F(c) ∩ F(u)|  d then
                current = c
                merge_flag = true
                F(c) = F(c) ∩ F(u), break
            end if
        end for
        if merge_flag == FALSE then
            if |F(current)| == N then
                current.type = FINAL_CLUSTER, brea
            else
                VIRTUAL_CLUSTER v = F(u)
                current.add_virtual_cluster(v)
                current = v
                if d == min(L,|F(u)|) then
                    current.type = FINAL_CLUSTER
                end if
            end if
        end if
    end if
end for
```

### 4.3 Anomaly Detection Method

A detection of timing service capability is established for the application system to obtain statistics in the anomalies period and the application information, so as to provide a reference for determining whether the application system is under attack.

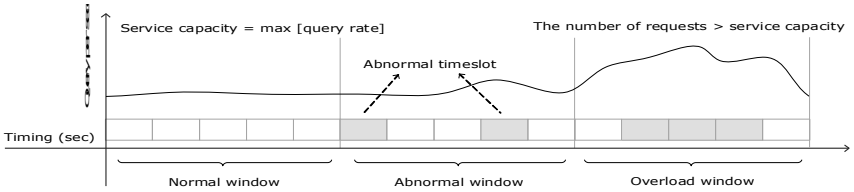



**Figure 15:** Anomaly Detection Method

The detection method mainly includes the following concepts:

(1) Query per second (QPS): the frequency of each traffic source's access to the application can be obtained by monitoring the historical data;

(2) Detection timeslot: default of 1 second;

(3) Abnormal timeslot: 90-percentile latency in the detection timeslot exceeds the threshold (default of 1.5 seconds or other value configured according to the application);

(4) Abnormal window: the window has at least N abnormal timeslots;

(5) Normal window: there is no abnormal timeslot in the window;

(6) Capacity lower bound (CLB): the maximum frequency of requests within the normal window is selected as the CLB. As the monitoring time continues to increase, CLB will be constantly updated;

(7) Overload window: the average number of requests in the abnormal window exceeds its service capacity;

(8) Overload ratio (OR): it refers to the proportion of the overload window relative to the total number of windows within an abnormal period.

The anomaly detection is mainly conducted to analyze the abnormal timeslot and overload window. In the monitoring process, the abnormal timeslot and overload window do not necessarily occur simultaneously. When an abnormal timeslot occurs within the detection window, there are two cases:

① QPS <CLB: it indicates that the current request rate is still within the system service capacity, so you can eliminate the possibility of system service anomaly caused by too high QPS, and we can find such problems through the practical application and tracking; in the case that you have excluded potential reasons when the system is handling a complex logic (such as generating reports or processing large files) or the system is carrying out the normal operation of data backup and batch updates, itcan be concluded that the decrease of the system service capacity is caused by other problems, such as server hardware and software error, internal network fault, configuration errors, that other application resources are too much overhead, etc.;

② QPS> CLB: it refers to that the system running is overloaded. For anomalies that have been detected to be caused by the system overload, we should evaluate the threat level through the traffic source threat rating process (Fig. 16), to provide the basis for banning the malicious traffic source.

The trust rating of the traffic source can be assessed according to the following indexes of the traffic source IP in the abnormal period:

Index 1: it's the proportion of the request number of the abnormal IP access application irelative to the request number of all abnormal applications, reflecting the influence of the traffic source on the application system anomaly; a large value of index 1 represents a high-frequency abnormal traffic source, which plays a key role in the abnormal events, thus banning such traffic source can greatly reduce the load of the current overloaded application. Therefore, in the face of abnormal overload, the frequency of the abnormal traffic source access application should become the primary consideration. If index 1 is not too large, the threat level is 0; if index 1 is too large, then IP already has a threat, and a further assessment should be made.

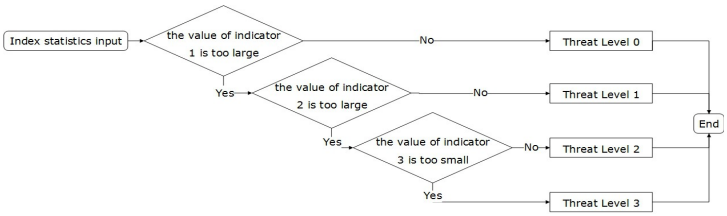



**Figure 16:** Traffic Source Threat Rating Process

Index 2: it's the proportion of the request number of the abnormal IP access application relative to the request number of all applications, which reflects the concentration of the abnormal access of the traffic source. If the value of index 2 is too large, the traffic source mainly access the abnormal application in the abnormal period, which has a higher level of threat than the traffic source with a more decentralized access.

Index 3: it's the number of the IP access application, reflecting the concentration of the application access. The smaller the value is, the greater the likelihood of the procedural attack is. Hence we need to use this index to increase the threat level of the traffic source.

### 4.4 The Overall Model

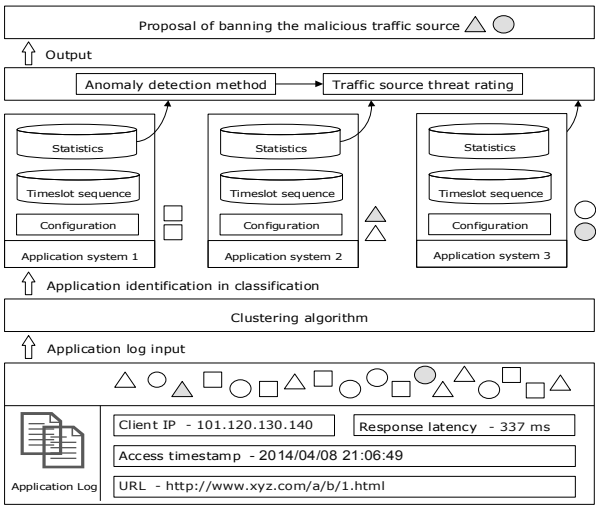



**Figure 17:** Overall Functional Architecture

The model takes the application log as the input. The log contains four fields, i.e., the client IP, the access URL, the response latency time and access time stamps. It achieves application identification in classification by URL clustering. It also establishes timeslot sequence within each application, to give dynamic statistics of the service status of the application and the application request information. It will analyze and record the abnormal latency and system overload information as the input of the traffic source threat rating model, with the use of anomaly detection methods according to the latency time and other configuration

information set by the user for application (as shown in Fig. 17). The model will give the statistics of the malicious traffic source information after running for a certain time.

## 5.Application Analysis

### 5.1 Shortcomings of Overall Granularity Anomaly Detection

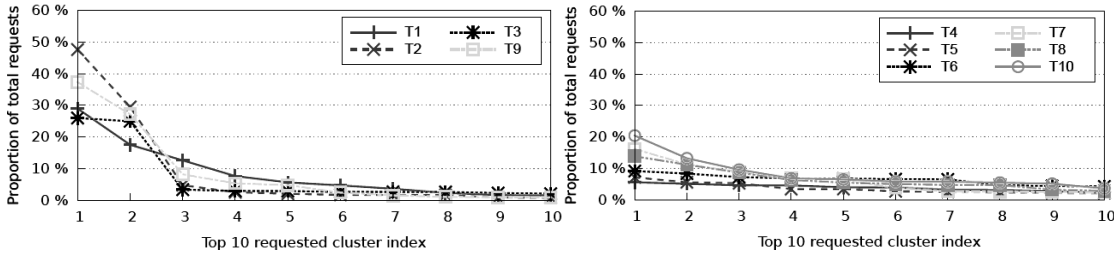



**Figure 18:** the Clustering Applications of the Top 10 Web in Traffic Rank and Their Proportions

In the clustering applications of the top 10 Web services in traffic rank, only four Web services have clustering applications with total traffic of more than 20% (Fig. 18), and most clustering applications have a traffic proportion of less than 5% of the total traffic.

Due to the above characteristics of the Web service, during the overall granularity detection, when the overall request latency or 90-percentile latency is used, it is difficult to detect the anomaly of a single application, because the traffic of a single application accounts for a very small proportion in the total traffic and is not enough to trigger volatility of the average or 90-percentile values.

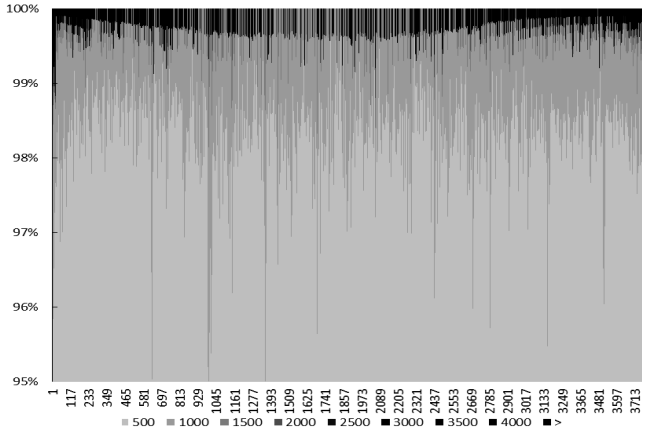
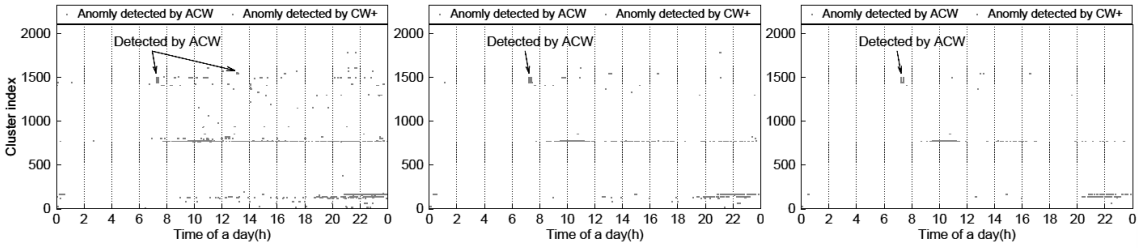



**Figure 19:** Percentile Distribution of an Application System Latency

Fig. 19 shows the latency distribution of an application system in a certain time period, in which, the vertical axis represents the latency percentile and the horizontal axis represents the number of seconds in an entire day, while the color depth represents the latency level. The deeper the color is, the more the latency will be. It can be seen that 99% of quantile latencies are in good condition in the vast majority of cases. If an application system accounts for less than 1% of the total traffic, even if all of its latencies are abnormal, it is difficult to be reflected on the overall granularity. In contrast, choosing the percentile will be too radical, and it will lead to constant triggering of latency anomalies due to a reasonable long tail effect of the system. This proves the necessity of a fine-grained anomaly detection, which means to detect different applications separately.

### 5.2 Results Comparison of Cluster Analysis and Overall Granularity Detection

The results comparison of the anomaly detection on 90-percentile latency is carried out between the anomaly detection method based on clustering applications (CW + in Fig. 20) and the overall granularity detection method (ACW in Fig. 20). The determination criterion for anomaly is whether the number of windows with the 90-percentile latency exceeding the threshold value within W windows is more than n.




(a) $n/W = 3/10$. (b) $n/W = 4/10$. (c) $n/W = 5/10$.

**Figure 20:** Results Comparison Under Threshold of Different Granularity (n/w = 4/10)

It can be seen that few anomalies are found in the detection results of the overall granularity (the point pointed by the arrow), which verifies the previous inference, and in the anomaly detection method based on clustering applications, the anomaly detection is conducted on each application separately and the latency fluctuation and anomaly of the application can easily be perceived, so this detection method is more sensitive and can better detect the hidden anomaly. In Fig. 20, there is an application with long-time anomaly in the middle of each figure, and an application with relatively concentrated anomaly is detected in the lower right portion. Meanwhile, with the increase of parameter n/W, sensitivity of the anomaly detection gradually reduces. Selection of parameters in this group depends on the needs of the actual system for sensitivity.

### 5.3 Selection of The Number of sub-Applications

In the process of application clustering, selection of the threshold value of high frequency mode is more critical, namely, when the number of sub-applications exceeds the threshold, sub-applications will be aggregated into an application, as an application identifier. Through the simulation experiment we can find the relationship between the number of sub-applications and their corresponding hits of two Web services with the most hits. The horizontal axis represents the serial numbers of applications, the red bar of the vertical axis represents the number of sub-applications, and the blue line represents the average hits of sub-applications. The left side of Fig. 21 displays that fewer than 20 applications have a large number of sub-applications (100-100000), and the average hits of their sub-applications are low, mostly less than 10 hits; the right side of Fig. 21 shows that the number of sub-applications of all the applications of this Web service is small, and the average hits of their sub-applications are relatively high. Thus, we can see that the number of sub-applications is inversely proportional to the average hits. So it needs to select appropriate parameters according to the specific application situation.

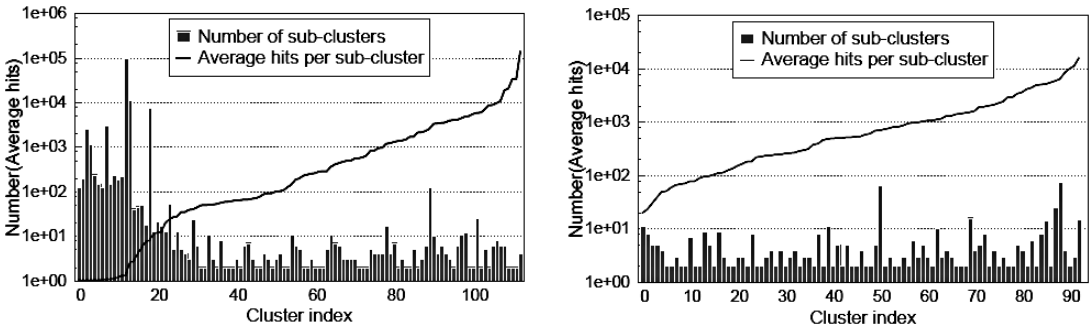



**Figure 21:** Relationship Between the Number of sub-Applications and Hits

## 5.4 Low-frequency Filtering

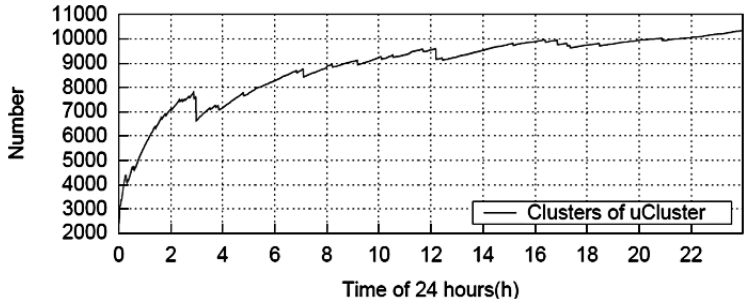



**Figure 22 :**the Number of Clustering Applications During the Clustering

In the process of clustering the complete data (more than 200 Web services), the low-frequency application filtering method is used to filter clustering results, namely, filtering low-frequency applications in clustering results. The low-frequency clustering filtering threshold is 0.01 times per second, and the sub-application threshold is set at 100. The change of its clustering application number is shown in Fig. 22. The horizontal axis represents the time of simulation data (24 hours a day), while the vertical axis represents the number of clustering applications. 10,000 applications are clustered from more than 200 Web services, averagely one Web service for 50 applications, which shows that the number of applications is fewer after clustering, thus greatly reduces the amount of detection data. For the curve vibration in Fig. 22, the rise represents that the model is constantly recognized for new applications as the URL is being processed constantly, and the fall shows that some applications are filtered out due to a lower frequency, or, application polymerization produces due to too many sub-applications.

## 5.5 Detection Results

This paper uses the detection model to get the application system list (shown in Fig. 23 on the left), selects an application, obtains the subfield set of this application system (shown in Fig. 23 on the right), and at the same time, gets the request and response distribution of this application system.

| Hostname | Requests per second | Total abnormal time (seconds) |
|---|---|---|
| zotwq.wqorx.egd | 104.23 | 6900 |
| vtfax.wqorx.egd | 70.92 | 6100 |
| hoe.wqorx.egd | 36.36 | 3100 |
| solz.odqut.wqorx. | 1.29 | 300 |
| hsqn.wqorx.egd | 5.91 | 200 |
| cortfg.wqorx.egd | 28.23 | 100 |
| wqoat.wqorx.egd | 23.8 | 0 |

| Application cluster and statistical information (host name: zotwq.wqorx.egd) | |
|---|---|
| Application clusters (total: 140) | Requests per second |
| Total | 104.23 |
| /miowg/ziktqr? | 30.19 |
| /y | 13.12 |
| /h | 7.96 |
| /lout/sqprdgfzi | 5.88 |
| /yqdt-iqss | 3.78 |

**Figure 23:** URL Clustering Results

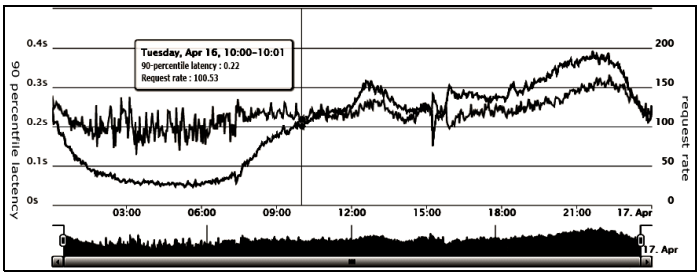



**Figure 24:** Application Request and Response Distribution (24 hours)

Narrow the time range to query the contrast of latency anomaly and request rate (shown in Fig. 24), and at the same time get the average latency time, the proportion of overload, duration of frequent IP and other key information of specific URL in the application after the clustering (shown in Fig. 25).

| 90th-latency average | anomaly proportion | average requests per second | maximum requests per second | overload ratio | frequent IP | proportion | application number |
|---|---|---|---|---|---|---|---|
| 4.13 | 70.00% | 1.60 | 2.00 | 100.00% | 58.153.19.83<br>183.221.29.118<br>61.158.152.146 | 3.12%<br>1.88%<br>1.88% | 1<br>1<br>4 |
| 2.79 | 50.00% | 3.95 | 5.30 | 100.00% | 221.11.51.170<br>111.36.137.219<br>120.90.98.192 | 22.03%<br>5.06%<br>4.81% | 3<br>10<br>4 |
| 5.94 | 50.00% | 0.16 | 0.60 | 60.00% | 210.21.225.242<br>219.219.124.215<br>120.209.215.223 | 68.75%<br>12.50%<br>6.25% | 3<br>1<br>2 |
| 3.07 | 80.00% | 0.79 | 1.30 | 100.00% | 110.115.194.155<br>111.36.137.219<br>120.209.206.179 | 6.33%<br>5.06%<br>3.80% | 8<br>12<br>5 |

**Figure 25: Application Request and Response Distribution and Abnormal Information**

## 6.Conclusion

Application-layer DDoS can exert deeper impact on the application system. Such attacks are characterized by low resource utilization, strong concealment and high attack yields. Traditional defense methods cannot distinguish details of the application against such attacks,or recognize such attacks. The model of application-oriented detection for denial of service attack gives a time series analysis for each application through the application log collection, sampling and using the clustering algorithm for partition of applications; it realizes a dynamic calculation of running indexes and abnormal information of the application system with timeslot as the time unit in ime series, and uses the abnormal information for the threat rating of malicious IP. It can provide a strong reference for the DDoS attack response and defense.

## References

[1] Suk-Bok Lee, Dan Pei, MohammadTaghi Hajiaghayi, Ioannis Pefkianakis, Songwu Lu, He Yan, Zihui Ge, Jennifer Yates, and Mario Kosseifi. *Threshold compression for 3G scalable monitoring*[C]//INFOCOM, 2012 Proceedings IEEE. IEEE, 2012: 1350-1358.

[2] Münz G, Li S, Carle G. *Traffic anomaly detection using k-means clustering*[C]// .Electrical and Computer Engineering (CCECE), Annual IEEE Canadian Conference 2007.

[3] Chen Y, Mahajan R, Sridharan B, et al. *A provider-side view of web search response time*[C]//Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM. ACM, New York, 2013: 243-254.

[4] Kalekar P S. *Time series forecasting using Holt-Winters exponential smoothing*[J]// Kanwal Rekhi School of Information Technology, 2004, 35(9):1-13.

[5] Brutlag J D. *Aberrant Behavior Detection in Time Series for Network Monitoring*[C]//LISA '00 Proceedings of the 14th USENIX conference on System administration ,USENIX Association Berkeley, CA, USA . 2000: 139-146.

[6] Choffnes D R, Bustamante F E, Ge Z. *Crowdsourcing service-level network event monitoring*[J]. ACM SIGCOMM Computer Communication Review, 2011, 41(4): 387-398.

[7] Wang Q, Kanemasa Y, Li J, et al. *Detecting transient bottlenecks in n-tier applications through fine-grained analysis*[C]//Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on. IEEE,Philadelphia, PA , 2013: 31-40.

[8] Minlan Yu, Albert Greenberg, Dave Maltz. *Profiling network performance for multi-tier data center applications*[C]// Proceedings of the 8th USENIX conference on Networked systems design and implementation. USENIX Association, 2011:57-70 .

[9] Chandola V, Banerjee A, Kumar V. *Anomaly detection: A survey*[J],ACM Computing Surveys (CSUR) , 2009, 41(3): 15.