Preventing Application Layer DDOS attack using Password Attack Resistance Protocol

Shama Anjum, Dr. Chandrashekara S N,
4th Sem Mtech, HOD Dept Of CS & E, CBIT Kolar ,
Computer Science and Engineering,
C. BYREGOWDA INSTITUTE OF TECHNOLOGY, Bangalore, India.

Abstract : Distributed Denial Of Service (DDoS) attacks are major blitz that effect the computers, servers and internet from many years and many methods and theories were developed to not only detect but also to protect your system from these type of harm. nowadays application layer distributed deniel of service attack towards the major servers online application increases very fast and the users have to face great loss.These attacks make the response slow for every request of the user and make the users to wait for the long time for the demand which they make. many users make the demand for the service at the same time from different systems or sources creating a disturbance because of which the site becomes busy and request will not be fulfilled in the accurate time. To overcome these defects many schemes were developed but to build a strong defence system the synergism of mitigation and detection of strikes is required.

Index Terms **- Application layer DDoS attacks, sketch data structure, intrusion prevention system.**

I. INTRODUCTION

DISTRIBUTED denial of service (DDoS) attacks have been a severe threat to the Internet for decades and numerous defense schemes have been proposed to mitigate DDoS attacks at the network layer. Recently, application layer DDoS (app-layer DDoS) attacks against web servers grow rapidly and bring victims with great revenue losses. App-layer DDoS attacks attempt to disrupt legitimate access to application services by masquerading flash crowds with numerous benign requests. Flash crowd refers to the situation when many users simultaneously access a popular website, producing a surge in traffic to the website and causing the site to be virtually unreachable. The stealthiness of app-layer DDoS attacks makes most signature-based intrusion
prevention systems ineffective.Since most DDoS attacks are launched abruptly and severely, it is desirable to design a defense system that can detect and mitigate app-layer DDoS attacks as soon as possible to minimize the losses. Turing test schemes based on graphical puzzles have been proposed to address the above problem on the cost of additional delays.Unfortunately, since a few milliseconds extra delay may cause users to abandon a web page early applying such mechanism to all users will negatively affect the Quality of Experience (QoE). Therefore, an effective defense system should mitigate app-layer DDoS attacks as soon as possible

A series of sketch-based approaches have been proposed for anomaly detection in large-scale network traffic.
Since sketches contain no direct information about the malicious hosts, they cannot be directly used for the mitigation of attacks. To tackle this problem, several efficient reverse hashing schemes have been proposed to infer the IP addresses of malicious hosts from reversible sketches. These studies attempt to retrieve the anomalous keys either by using reverse hashing methods or by storing parts of the keys. However, these methods are either computation-intensive or storage-demanding, limiting their applications in intrusion prevention systems. The challenge of designing a sketch-based defense system lies in the coordination between the detection and mitigation of attacks. First, since network traffic is inherently dynamic in the real environment, a proper measure of network traffic is essential to accurate anomaly detection. Second, when an attack occurs, it needs to identify malicious hosts accurately without affecting the access of legitimate users. Third, since most attacks are launched abruptly, the defense system should adopt an effective light-weight process to detect and mitigate the attack as soon as possible. To address these issues, we propose a novel defense system based on sketches to defend against app-layer DDoS attacks.

II. LITERATURE SURVEY

2.1 Existing System:

Application layer Distributed Denial of Service(DDoS) attacks pose very serious problem to the servers and the websites by sending many http request it gradually slow downs service and makes the delay in providing the services to the legitimate users. As many of these offensive works is done immediately and cause damage, it is very important to find a system that can not only recognize the fault but also overcome them.

There were many programmes and strategies that were developed to overcome these attacks such as Turing test scheme based on graphical puzzles, reverse hashing schemes, storing the parts of keys, taxonomy on DDoS attacks, etc., but they had certain disadvantages which were not suitable to handle these attacks, hence, it was important to develop a system that can handle these strikes as soon as they occur.

Disadvantages:-

1. Had many delays and the quality of the service was affected by the users.
2. They required too much of computation.
3. It required large amount of storage and were limiting the applications
4. They were time consuming. Necessary steps should be taken, for any attack

5. Testing was not done completely.
6. Server was detained, and did not thoroughly avoid it.
7. Response was delayed detection was very slow.
8. Needed merging of different methods.
9. User should solve the puzzle to send the request to the website.

2.2 Proposed System

We develop an successful guarding system named SkyShield, which uses the sketch data structure to find and lighten the application layer Distributed Deniel of Service. Firstly, the system detects the hacker by checking whether the host has made any abnormality in the preceding detection cycle which helps to minimize the faulty request.

It acts as barrier that mitigates the problem without the need to recapture the exact IP address. Hence, SkyShield system defend these attacks effectively. The test results show that, SkyShield can immediately reduce the harmful requests, while having small amount of influence on normal end users.

In the proposed system users are categorized as whitelist, blacklist and suspicious:
White list: legitimate user that pass the CAPTCHA test and can login without any issues.
Blacklist: malicious request is checked by the blacklist are filtered and recorded and they can send number of request to activate the application layer DDoS
Suspicious request: first claim whether its original is in whitelist if not host will be checked. If passes test added to whitelist else added to blacklist

Advantages:
1. Attacks are identified as soon as they occur.
2. Strong authentication is provided and prevent the hacker from penetrating.
3. Sketch based organizing of data is done.
4. Different hash code based CAPTCHA techniques are used.
5. Unique system of email and code generation is used.
6. Other techniques such as fog computing, bloom filters based on which hash code is merged, are used to guarantee the effectiveness of SkyShield.

III. BACKGROUND

A. Sketches

A sketch is a type of data structure composed of h hash tables of size k. it is used to efficiently estimate the original signals by aggregating high dimensional data streams into fewer dimensions. A sketch is an approximation tool to efficiently estimate a signal by sacrificing tolerable accuracy. Due to the randomization of hash functions, the distribution of values in each hash table is relatively stable for normal network traffic. Therefore, sketches are capable of detecting significant changes in massive data streams, such as high-volume network traffic.
.

B. Bloom Filters

A Bloom filter is a space-efficient data structure for set membership queries A Bloom filter employs k independent hash functions and a set A Bloom filter may also lead to a negligible false positive rate. A false positive means that an element being checked is mistakenly determined by the above criteria whereas it is actually not in the set. It is caused by conflicts of keys that occasionally share common hashing results for all hash functions. The false positive rate can be decreased by carefully adjusting the number of hash functions

IV. SYSTEM OVERVIEW

Figure 4 depicts the process of SkyShield. It is deployed behind a network fire wall that will filter out malformed HTTP requests, and the process consists of two phases, namely, mitigation and detection. In the mitigation phase, Sky Shield employs two Bloom filters, including a whitelist (B1) and a blacklist (B2) to filter incoming requests. The whitelist (resp. blacklist) contains the legitimate (resp. malicious) hosts that are confirmed by the CAPTCHA techniques. Normal requests verified by the whitelist are passed to the detection phase directly whereas malicious requests verified by the blacklist are filtered and logged. The remaining requests are inspected based on the abnormal sketch S3. The rationality behind this scheme is that malicious hosts need to send numerous requests persistently to launch effective app-layer DDoS attacks. Therefore, SkyShield can identify malicious hosts without reversely calculating their IP addresses. Detailed mitigation method is described in Section IV. For a suspicious request, SkyShield first examines whether its origin is in the whitelist. If not, the host will be checked by the CAPTCHA module. If the host passes the CAPTCHA test, it will be added to the whitelist. Otherwise, it will be added to the blacklist. Since only suspicious hosts are tested by the CAPTCHA, only parts of legitimate users might be affected. Additionally, to prevent blacklisted users from being
blocked forever, both the blacklist and whitelist are emptied periodically. Initially, B1, B2 and S3 are set to be empty and no hosts are suspected and filtered. In the detection phase, SkyShield exploits the divergence between two sketches S1 and S2 as a

signal to detect anomalies that are caused by numerous requests originated from malicious hosts. SkyShield conducts the detection cyclically with a fixed time intervalT, which is an adjustable parameter. By adjusting T, the system canbalance thetrade-offbetween the attack mitigation speed and the detection accuracy. In each detection cycle, all incoming requests are aggregated into S1, with the source IP addresses as input keys. The backup sketch S2 stores the results of S1 in the last normal detection cycle. At the end of each detection cycle, the divergence d(S1, S2) between S1 and S2 is calculated. If d(S1, S2) exceeds a threshold θt, the system is supposed to suffer an attack and an alarm is raised. If an anomaly is detected, S2 will not be updated anymore. This guarantees that the current sketch is always compared with a normal pattern. Alternatively, S3 will be updated by S1 and the abnormal buckets are calculated. When the alarm is lifted, S2 will be updated by S1 again at the end of each detection cycle and S3 is emptied.

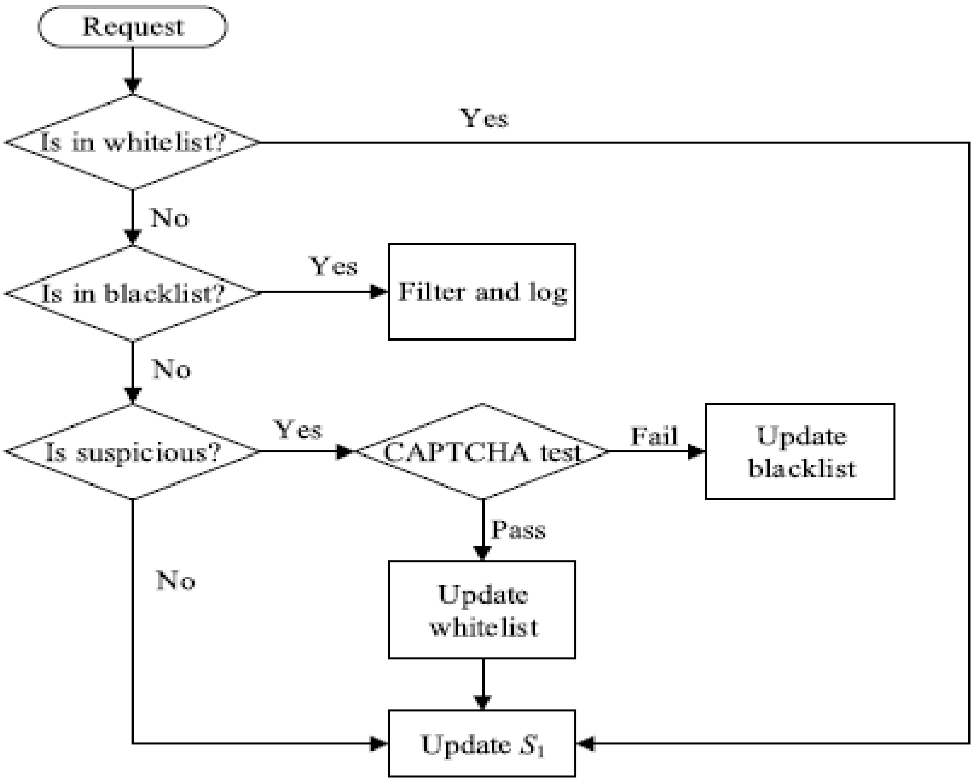

figure 4: system overview

V. SYSTEM REQUIREMENTS

System analysis helps in knowing the user needs or potential client system needs before designing the actual system. Data collected during the analysis is converted into a report which specifies the requirements. It gives small explanation of all the facilities the system provides.

5.1 Functional Requirements:

- Web enabled implementation system is required which is built by using advance technology of java along with Tomcat web server and a database server such as MYSQL
- Two actors in this system are the admin and end user.
- Admin is the important user who can login using valid user name and password in his home page without any kind of issues.
- Admin has the rights to create the users ,view the number of users and the ip address that is blocked can also be viewed by the admin.
- Appropriate user id and password is given to the end users once they register themselves successfully
- Using the authorized id and password users can login the system without any issues and delays in case the user tries to login using the wrong id or password the counter that records all the wrong attempts starts at the respective server.
- Fault attempt monitoring system will keep track on the number of all failed attempts by the user.
- The system which keeps track of the failed try made by the user has t1 and t2 threshold.
- Suspicious ip address for t1 and white listed ip address for t2
- There are different tables were the failed attempts are recorded namely table_fs and table_ft in our database.
- Then sketch based CAPTCHA test is asked once the end user exceeds the threshold
- When the IP address are recorded in the block list the system will never respond to that address again.

5.2 Non-Functional Requirements:

Usability

This system is simple for the user to understand and to work with. The home page shows the number of users present it the system, all the details of the user can be viewed by him at any point of time without any difficulty. Changes can also be made by the valid user in his profile. User can work in this new system and new environment without any complications.

Reliability

The project is well grounded and all the data stored is protected. Also the penetration testing is done to provide strong data and user authentication all the changes done by the user can be viewed by the system. Passkey is provided to the valid user whenever he tries to download the uploaded file. As a result all the data is provided with high amount of security .

Performance

This system can be used by many users at the same time. When a user tries to login from the trusted system after three attempts with wrong password or id the CAPTCHA test occurs in case of non trusted system the test occurs after 5 attempts. System is controlled by single web server and a database server in the background and hence the performance becomes an important criteria.

Scalability

The system is flexible enough as any number of users can be added at any point of time by the admin and changes can be made by the user. User is also allowed to change the password if required.

Portability

whenever the os gets crash or the web server gets stuck the entire code or the project can be ported to another system without any difficulties.

Maintainability

Maintaining this system does not require much exercise. Monitoring of the data is done and its privacy is also maintained. If too many activities are running it becomes difficult to handle the system.

VI. SYSTEM DESIGN

6.1 System Architecture

System Architecture design-identifies the overall hypermedia structure for the WebApp. Architecture design is tied to the goals establish for a WebApp, the content to be presented, the users who will visit, and the navigation philosophy that has been established. Content architecture, focuses on the manner in which content objects and structured for presentation and navigation.Fig 6 shows the WebApp architecture, addresses the manner in which the application is structure to manage user interaction, handle internal processing tasks, effect navigation, and present content. WebApp architecture is defined within the context of the development environment in which the application is to be implemented.

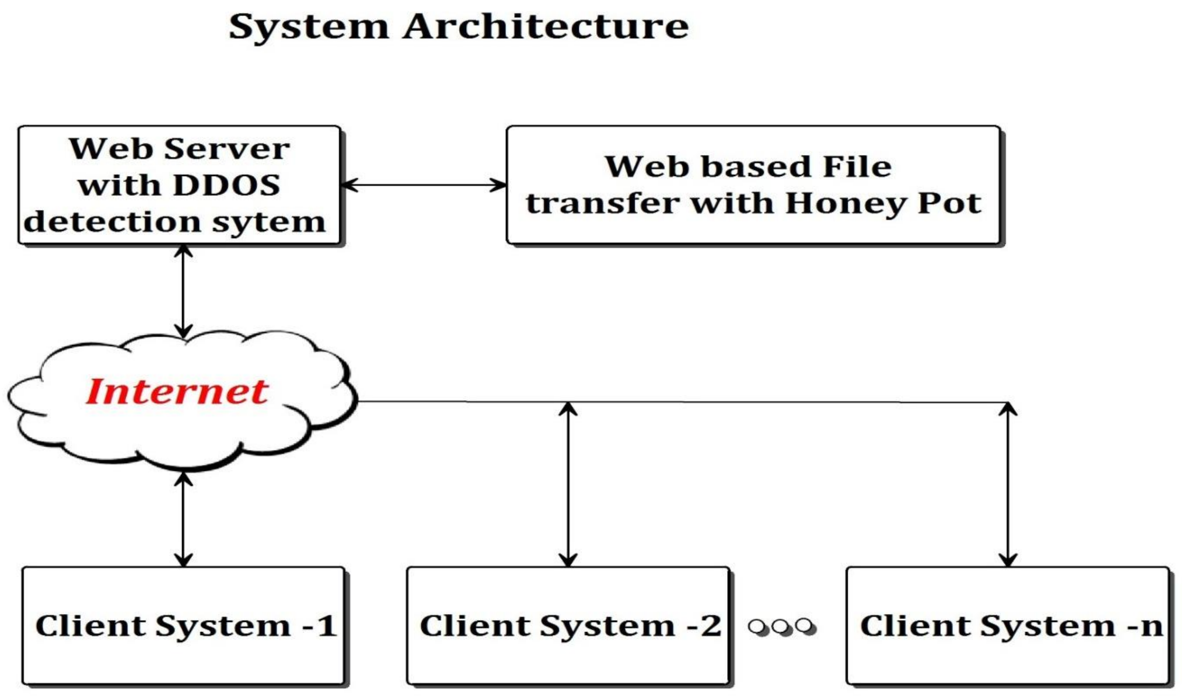

figure 6.1 system architecture

6.2 Use Case Diagrams:

A use case is a set of scenarios that describing an interaction between a source and a destination. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors. shows the use case diagram. Fig 6.2.1shows the use case diagram of the admin who can login and view all the blocked IP and users, and any number of users can also be added by the admin.

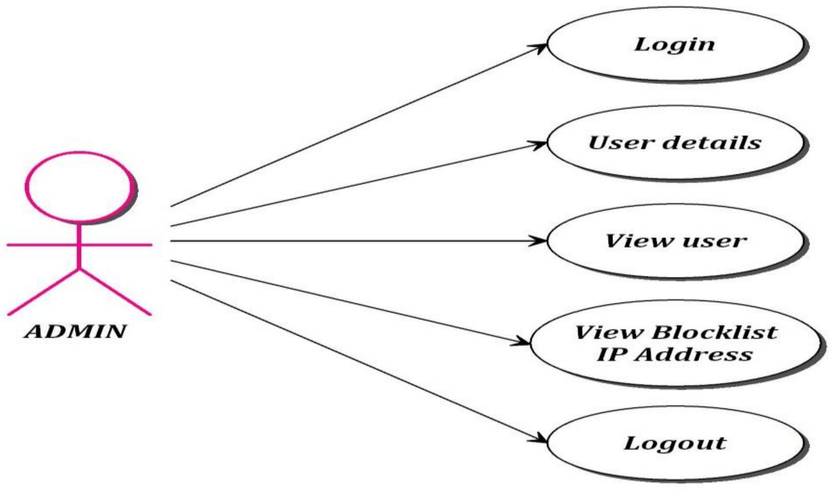

figure 6.2.1 use-case diagram admin

6.2.2 Use Case Diagram User:

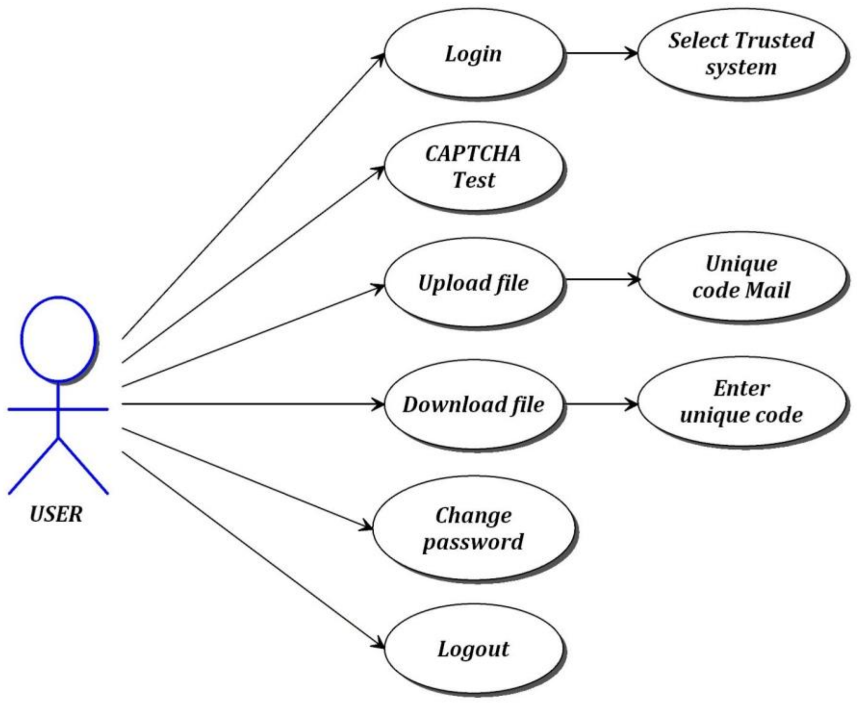

figure 6.2.2use case diagram user

Fig 6.2.2 explains the use case diagram for the user who can login with valid username and password, change password, download the uploaded file by using the unique passkey sent to his mail.

VII. IMPLEMENTATION

The implementation phase involves more than just writing code. Code also needs to be tested and debugged as well as compiled and built into a complete executable product. We usually need to utilize configuration management in order to keep track of different version of code. This is the stage of the project where the theoretical design is turned into a working system. If the implementation is not carefully planned and controlled, it can cause chaos and confusions. It is always a good idea to keep in mind that some characteristics that should be found in a good implementation like Readability- our code is written in MVC Architecture ,JAVA to achieve the objective of the project that is to introduce a novel scheme of mechanism design for balancing the resource consumptions .
Our implementation stage requires the following tasks:

- Careful planning
- Investigation of system and constraints
- Design of methods to achieve the changeover
- Evaluation of the changeover method
- Correct decisions regarding selection of the platform
- Appropriate selection of the language for application development

Java Technology Java technology is both a programming language and a platform.

7.1 The Java Programming Language

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

7.2 Modules

➢ **USER**

- Who can login with valid user name and password .
- view profile, edit profile.
- user can upload any file and also view all the uploaded files
- one can download the file only by using the password that is generated to their email.
- finally can logout whenever he needs

➢ **ADMIN**

- can login with a valid name and password
- admin can edit his profile
- add any number of new users, change
- change the password if required and logout

VIII. CONCLUSION AND FUTURE ENHANCEMENT

To defend against application layer DDoS attacks, it is essential to have a fast response system that can automatically detect and mitigate malicious requests as soon as possible. In this paper, we design and implement such a system named skyshield by taking advantages of the sketch techniques. First, to calculate the divergence between sketches in two consecutive detection cycles. Second, to identify malicious hosts efficiently, we use the abnormal sketch obtained from the last detection cycle to avoid the reverse calculation of IP addresses. Third, we leverage other techniques including Bloom filters and the CAPTCHA techniques to guarantee the effectiveness of skyshield. We have developed a prototype of skyshield and carefully evaluated its performance using real attack datasets collected from a large-scale web cluster. The experimental results demonstrate that skyshield can effectively mitigate application layer ddos attacks and pose a limited impact on normal users.

Future Enhancement

I have worked hard to develop a better and improved defense system against application layer DDoS attack which is better than the existing system. advance technologies are used to make it more secure and provide a good security for the data stored over the cloud. still i found that it could be done in a better way.

the next enhancement is that, we can enable the fingerprint based schemes for a secure authentication which includes the entire information of the user's fingerprints and personal authentication can be determined only by the fingerprint features, which provides more security for the data stored.

REFERENCES

1] M. Ali, S. U. Khan, and A. V. Vasilakos, ``Security in cloud computing: Opportunities and challenges,'' *Inf. Sci.*, vol. 305, pp. 357_383, Jun. 2015.

[2] Z. Wan, J. Liu, and R. H. Deng, ``HASBE: A hierarchical attribute-based solution for _exible and scalable access control in cloud computing,'' *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 743_754, Apr. 2012.

[3] D. X. Song, D. Wagner, and A. Perrig, ``Practical techniques for searches on encrypted data,'' in *Proc. IEEE Symp. Secur. Privacy*, May 2000, pp. 44_55.

[4] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, ``Public key encryption with keyword search,'' in *Eurocrypt*, vol. 3027. Berlin, Germany: Springer, 2004, pp. 506_522.

[5] K. Kurosawa and Y. Ohtaki, ``UC-secure searchable symmetric encryption,'' in *Financial Cryptography*, vol. 7397. Berlin, Germany: Springer, 2012, pp. 285_298.