DROP-FAST: Defending against DDoS Attacks using Cloud Technology

Rashad Aliyev[1], **Dongwon Seo**[2], **and Heejo Lee**[2]
[1]Department of Computer Science and Engineering, Korea University, Seoul 136-713, Republic of Korea
[2]Department of Computer Science and Engineering, Korea University, Seoul 136-713, Republic of Korea

Abstract— *DDoS attacks continue to be a major threat to network security. Several new types of attacks such as Layer-7 attacks (e.g., HTTP flood, Slowloris, RUDY, etc.) have emerged. We propose a novel DDoS defense mechanism called DROP-FAST. Our mechanism provides distributed DDoS defense utilizing multiple replicas of the protected server throughout a cloud infrastructure. DROP-FAST is dynamic and can adapt by controlling the number of replicas on cloud based on attack strength. Main server is isolated from network using replica servers. Service quality features such as response time, incoming traffic load, and load sharing are improved due to distribution of attack and replication of the main server throughout the cloud. We describe our mechanism in detail and discuss improvements made over previously existing related works. We set up an experiment that shows significant improvement of the traffic load on the main server as a result of utilizing DROP-FAST mechanism.*

Keywords: DDoS Attack, Cloud Computing, Load Sharing, Network Security

1. Introduction

Cloud systems are advantageous for utilization in the field of computer security due to their distributed nature. Cloud technology enables efficient resource management since more cloud resources can be allocated on demand. Absence of a specific location is a very useful feature of cloud systems. A single physical machine can be a host for several cloud services. Single cloud service can be distributed over several physical machines. It is very hard to understand the structure of the cloud network due to absence of information about the amount of resources available.

DDoS attacks are one of the main problems that need to be addressed in the field of network security and communications. Whether a particular system is vulnerable to DDoS attacks depends on the amount of computational resources available. The more resources the system has the more complex and larger attacks it can withstand. It is therefore straightforward that the system resources should exceed the power of the botnets used for the particular attack. Not knowing how strong an attack can be makes it difficult to prepare a stable and reliable defense method. Several cloud infrastructures existing today already have an exceeding resource capacity compared to most of existing botnets [1]. Therefore cloud systems need to be utilized for DDoS defense. In such a way, system resources can be shared among a large number of clients, making it more efficient and low cost.

Existing DDoS prevention solutions are either victim-based or network-based. Victim-based solutions suffer drawbacks such as necessity of oversized server resources (e.g., bigger servers, larger bandwidth). Network-based solutions have challenging drawbacks such as link congestion issue. Newly appearing types of attacks such as Layer-7 DDoS attacks (HTTP flood, Slowloris, RUDY, etc.) contribute to the difficulties with network-based approaches as well. Signature based solutions are utilized across various types of existing methods. Attack signatures are automatically shared among service providers [2]. Signature-based solutions are not strong enough due to the time consuming process of signature collection and analysis. CDN (Content Delivery Network) based approaches also exist. Solutions such as Akamai [3], and CloudFlare [4] are well known. As any CDN-based method they utilize cache servers. This means they are vulnerable to cache pollution attacks such as locality disruption attack, or false locality attack [5], [6]. Attackers might keep sending requests for unpopular data. This will pollute the cache stored in different locations within the systems topology leading to disruption of service and additional cost.

As a result of our study, we have come up with a distributed DDoS prevention mechanism utilizing cloud technology, DROP-FAST. Our approach is based on utilizing massive computing power of the cloud infrastructures, and distributed nature of the cloud allowing protection from DDoS attacks. The core concept is switching from centralized defense strategies to a distributed scheme via cloud. This is achieved by moving the battleground from the main server to the cloud. Providing a method for handling attacks using cloud infrastructure greatly improves defense possibilities and service stability. DROP-FAST is based on replicating the main server throughout the cloud infrastructure. Content of the replicated servers is kept synchronized with the main server. All clients are serviced through cloud allowing for a strong protection of the main server while providing fast and robust service. In order to achieve our goals, we have defined several requirements and principles.

Requirements:

1) Server response time improvement.
2) Filtering improvement.
3) Main server isolation from attackers.
4) Load-sharing utilization.
5) Attack traffic drop location improvement.
6) Service stability under a strong attack.

Principles:

- *Dynamic* in its ability to adapt based on attack strength.
- *Reactive* since defensive actions will be taken immediately after an attack is detected.
- *Optimized* since amount of resource to use and network locations for replicas to reside can be chosen.
- *Proactive* since replica servers are ready to be activated at all times.

Based on the principles described above we named our mechanism *DROP-FAST*.

Several works describing cloud-based defense systems exist [1], [7]–[9]. In Section II we shall discuss pros and cons of some of them. In Section III we describe the DROP-FAST mechanism and compare it with some previous works. We continue with a description of a simple experiment in Section IV and state our ideas for future work. The paper is finalized with a conclusion in Section V.

2. Related Work

Researchers acknowledge that modern protection tools and mechanisms already have the capabilities and power to scan almost any kinds of objects on various depth levels on the network [1]. The problem that has been noticed is the delay between the time a piece of malware is discovered and the time of protection being available. Cloud computing has been suggested for storing the latest protection solutions. Clients would be checking for updates from a single location on cloud. This type of strategy allows making use of the geographical location of clients and creating cloud instances closer to large groups of users. In our research we make use of the vast computing resources cloud infrastructures provide. We also try to choose the positions for the replica servers so that it is closer to large groups of clients. In such a way, a faster response time and better load sharing is possible.

Another important issue to consider is the availability of the cloud system itself. The cloud system could be a target for a DDoS attack just as any other computing system. The cloud system that happens to be under a DDoS attack will most probably request more computing resources. The reason is that any service provider would like to keep stable service even under attack. However this leads to financial loss due to high cost. Therefore, cloud providers resolve the issue by imposing limitations for clients. Service providers in their turn impose limits and define thresholds for users. With events such as flash crowds or when a valid client requests large amounts of data, the threshold is passed but it is not an attack. These cases were analyzed by researchers and alternate strategies such as load balancing and honeypot were suggested [7]. There is a need for the service providers to decide on the maximum amount of resources that they would like to be using at pick moments. This is necessary due to financial issues of each service provider. It is necessary to realize that even in that case a very large attack or a flash crowd may lead to demand of very large computing resources. Therefore, a threshold on the maximum amount of cloud resources to be used should be in place.

Cloud systems provide distributed infrastructure. This also applies to intrusion detection. It is possible to combine existing intrusion detection systems (IDS) and cloud infrastructures. This allows a better protection for all machines inside the cloud infrastructure as well as for the clients subscribed for the cloud service utilizing cloud-based IDS. Researchers have shown that combining IDS and cloud technologies shows significant improvements in attack detection rate, average packet analysis time, and process size [8]. The advantages gained by such an approach are immense. Nevertheless, these systems would suffer the same weaknesses as the IDSs that are being used. Once an attacker is able to bypass detection tools, it is a matter of time to get the service down. The reason is that the cloud instances are forwarding all of the received traffic to the main server. DROP-FAST prevents requests from bypassing the cloud instances and protects the main server.

Isolation of the protected server, improved response time, and better throughput are several criteria that need to be improved via usage of cloud computing. These goals have been achieved in a system called CLAD [9]. In the CLAD system protected server is completely isolated from the internet by cloud infrastructure. DNS settings are changed so that all requests are forwarded through at least one CLAD node. This allowed for better filtering and improved dropping location as well as response time and throughput. The system allows only for HTTP traffic to pass. Having filters on each CLAD node reduces the attack rate. This system is significant improvement for cloud-based DDoS defense solutions. However, the isolation is partial and the requests are still forwarded to the main server. DDoS attack on the main server is still possible. Therefore, in our mechanism we provide better protection by completely isolating the main server. DROP-FAST cloud instances are replicas of the main server. There remains no need to forward client requests since each replica can send a respond on its own.

3. DROPFAST Architecture

We propose applying distributed defense against distributed attacks. Non-distributed attacks are handled well by modern defense systems. It is the distributed attacks which pose the main threat nowadays. DROP-FAST is based on the idea of providing efficient and secure service by distributing the load from one main server to an infrastructure of cloud based replicas of the main protected server. Fig. 1 describes the outline of the general structure that DROP-FAST provides.

As seen from Fig. 1 the given scheme isolates the main server from all users, malicious and valid. All users send their requests to the nearest cloud replica. There have been previous approaches allowing main server isolation [9]. The advantage of DROP-FAST is that requests are not forwarded to the main server but can be handled by the nearest cloud replica. Cloud replica is closer to the clients compared to the main server. This decreases the response time significantly. Server load decreases as well as a result of traffic distribution over the cloud replicas. The requests handled by the replicas do not need to be forwarded to the main server, thus allowing

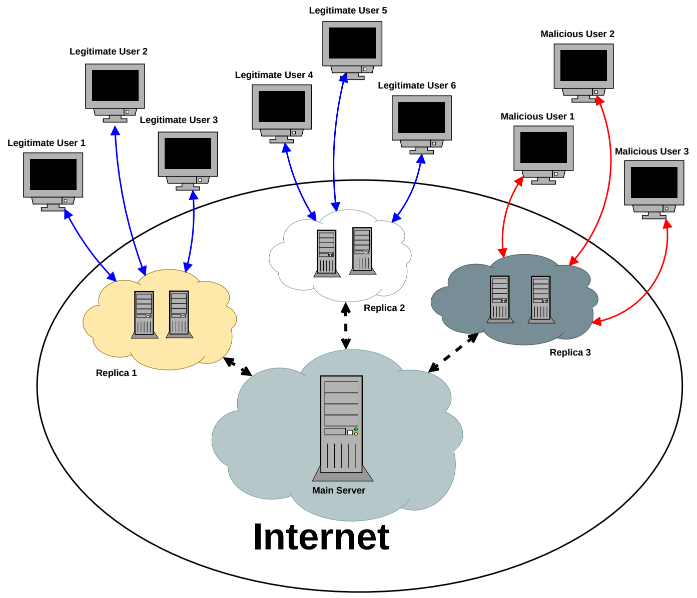

Fig. 1: DROP-FAST Architecture: Providing distributed service by replicating the main server throughout the cloud infrastructure

the main server to perform more efficiently. Several gateways can connect the main server with the internet and with clients depending on how large the network is. Therefore, appropriate locations can be chosen for placing the cloud replicas allowing the main server handle the requests coming from gateways connecting the server with parts of network without cloud replicas. Several issues should be handled in order for DROP-FAST scheme to work:

- Replication types
- Synchronization
- Load sharing
- Resource monitoring
- Security

3.1 Replication Types

Several types of replicating the main server on cloud exist. Various approaches could be applied based on the type and structure of data contained on the main server and contents. Short description of three main replication types is given.

3.1.1 Complete Copy

The main server is completely reproduced on the cloud replicas. All requests are handled by the cloud replica with no need of forwarding to the main server. Obviously, response time should be improved. DDoS risks are also greatly decreased because number of clients handled by each cloud replica is smaller than the number of clients that a single server would normally handle. Therefore, the total load is distributed among the cloud replicas. Since all requests are handled by cloud replicas directly, malicious requests do not reach the main server keeping it alive. The only communication between the replicas and the main server is synchronization of contents in order to keep data integrity. The main advantage is that failure of a single replica does not mean denial of service since other replicas would continue their service and main server would still be available.

3.1.2 Interest-based Copy

Replicas are partial copies of the main server. Both the main server and cloud replicas should be handling requests from clients. Depending on the content copied to the cloud replica, it will handle all the requests that it has the corresponding contents for, but forward the rest of the requests to the main server. Copying only part of the content makes it easier to synchronize and reduces the cost since the amount of necessary resources decreases. This approach might look similar to the CDN-based methods at the first glance. The main difference is that in CDN the contents of cache servers are determined by the frequency of requests from clients. Only the most popular data is stored. This opens up different attack channels such as cache pollution etc. We call this interest-based because we intend to copy only the content decided and agreed upon by the content providers. Content of cloud replicas would not necessarily change depending on what data is popular. This prevents cache pollution attacks. Advantage of using cloud systems is that a malicious user has no way of determining whether the data contained in the response packet is from the main server or cloud replica. The reason is that all the communication is done through the cloud replica. Nevertheless, an attacker might figure out what data is stored on the replica, leading to a possibility of an attack. Attackers could simply send huge amounts of requests only for the data residing on the main server. This would pose a serious threat if the attacker would launch an attack from all the locations where cloud replicas are placed. Therefore, the policy for choosing the data to be copied should be carefully analyzed and well planned. Due to risks stated above, we stick to the complete copy mechanism in our experiments.

3.1.3 Content Type-based Copy

The content could be divided into disjoint or partially joint sets or groups. This division can be carried out based on several criteria such as types of data (multimedia, documents, executable files, user files, etc.), or based on URL. Different cloud replicas would contain different portions of the data stored on the main server. URL splitting could be used if we decide to use content type-based copy mechanism. Different way of synchronization would be needed in case of using content type-based copy. There is a need to synchronize the cloud replicas between each other and the main server in order to keep data integrity.

3.2 Synchronization

Synchronization is a major concern in DROP-FAST. Same data should be available at different locations leading to unavoidable need of synchronization. We need synchronization in multiple directions. Modifications made on one of

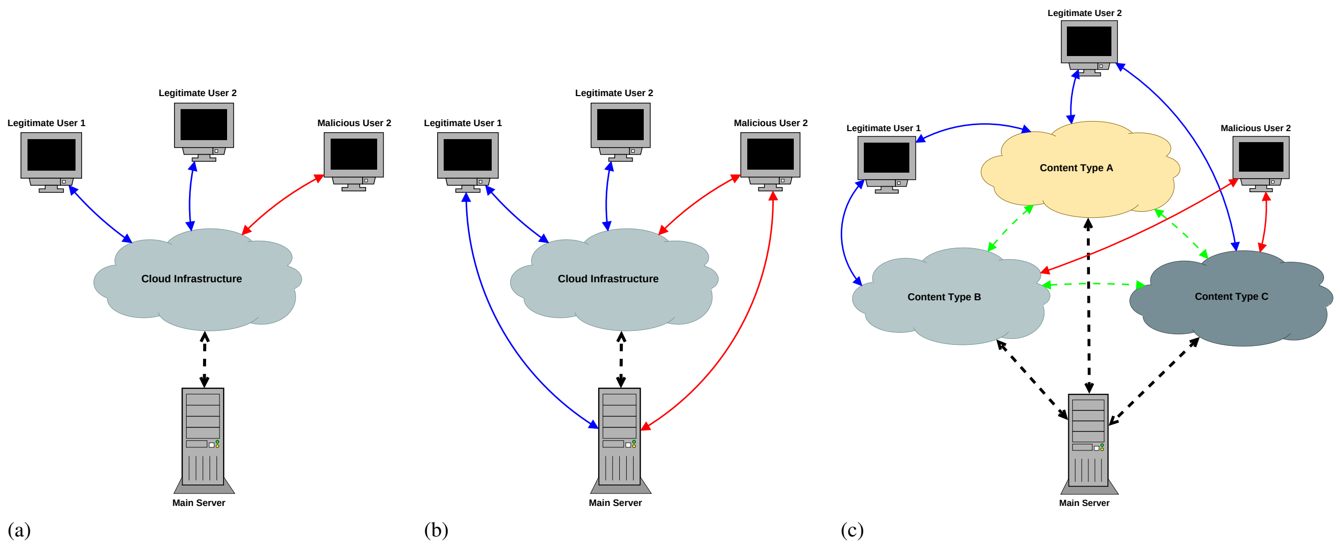

Fig. 2: Types of Content Copy: (a) Complete Copy (b) Interest-based Copy (c) Content Type-based Copy

the cloud replicas should be applied throughout the whole architecture including the main server. Various commercial software solutions are available for server synchronization. Modern synchronizations tools usually provide basic load sharing capabilities as well. For purposes of concurrent updates and distributed access to data making use of distributed file systems seems legitimate. Several free and commercially available tools for server synchronization exist.

3.3 Load Sharing

Depending on the network topology, each of the cloud replicas will be handling requests from a particular partition of the network. User behavior is hard to predict. Some parts of the network might have higher load compared to other parts. There is a need for strategy and policies in order to share the load and balance out the traffic to evenly divide it among closest replicas. Even though cloud infrastructures have large amounts of resources available, they cost money. It is a better choice to share the load among several replicas. Several load sharing and balancing approaches can be utilized. Most of the available methods are based on DNS configuration modifications or usage of multilayer switches.

3.3.1 DNS based Load Sharing

Load Sharing with Round Robin DNS is one of the straightforward ways of sharing that comes up to mind. BIND software should be used for this purpose. The idea is to simply rotate among several available IP addresses for the same domain name. The TTL (Time to live of DNS cache records) is set to relatively small values so that the A records (Address records linking a domain to an IP address) are renewed more often.

Round Robin is has low cost and is easy to implement. Nevertheless, there is no insurance in case of physical failure of one of the connected server machines. Caching problems might occur due to the need of having several requests per session. This will cause the DNS responses to the queries to cache and not being updated often.

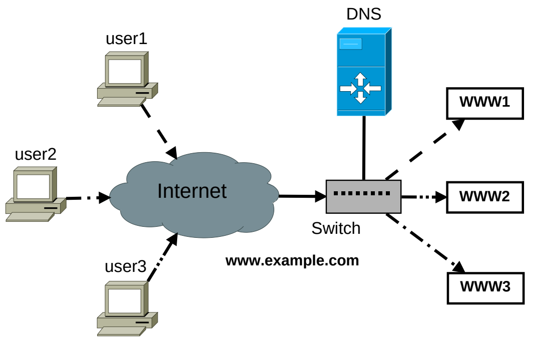

Fig. 3: Load Sharing using DNS Round Robin

In a systems such as DROP-FAST, where we try to share the load efficiently among several replica servers Dynamic Load Balancing [10] might be a better solution. For making use of this method we need to have some kind of resource monitoring to be in place on each of the replicas on the network. This is needed for determining the optimal decision of forwarding the traffic to the least busy server.

3.3.2 Switch based Load Sharing

Various network switches can manage the problem of load sharing and load balancing. Switches are able to automatically balance the traffic across several paths. Decisions as to which switch to use and what type of algorithms are utilized are specific to each case and network, they could change depending on the situation. Layer 7 switches could be used for performing load balancing on HTTP, HTTPS or TCP/IP traffic.

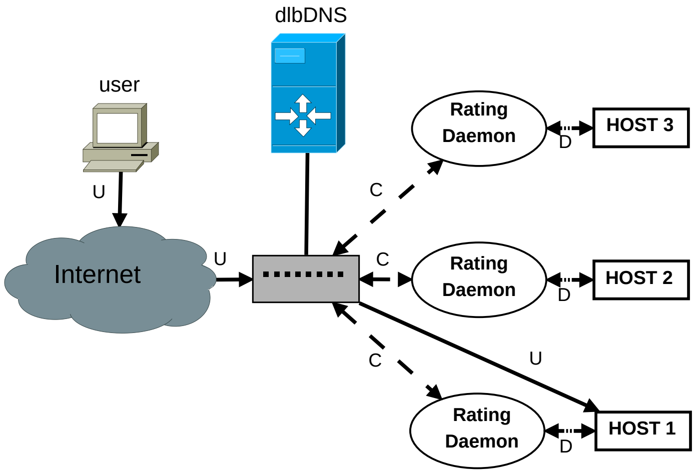

Fig. 4: Load Sharing using Dynamic Load Balancing (dlbDNS) : Path U - traced by user requests, Path C - communication between dlbDNS and rating daemons for locating the best server, Path D - process of updating server ratings by rating daemon

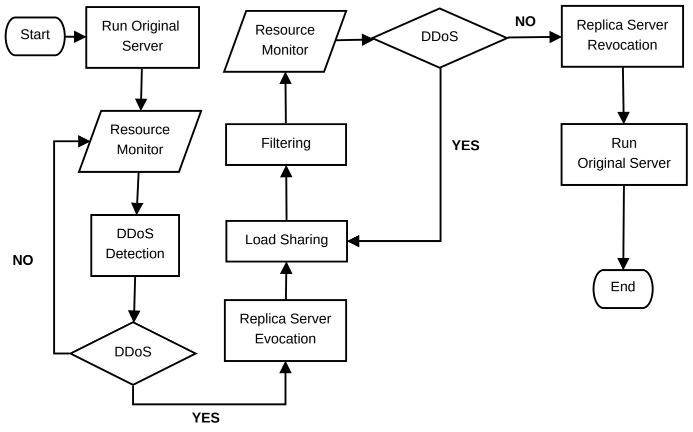

Fig. 5: DROP-FAST : Control Flow

3.4 Resource Monitoring

Resource monitoring is essential within the DROP-FAST mechanism. First, DROP-FAST needs it for implementing load sharing algorithms to gain load information from each individual replica server. Second, monitoring is needed for deciding whether the servers are under attack or not. Resources and network performance of the main server should be monitored in order to evaluate our ideas. Following metrics can be used:

Table 1: Criteria for Evaluation

Criteria	**Unit**	**Description**
Response Time	Seconds	Servers' Response Time
Server Load	Packets Per Second	Packet Rate of traffic reaching the main server
Response Sustainability	$\frac{N_{res}}{N_{req}}$	Ratio of number of responses and requests

Criteria in Table 1 and other metrics such as the CPU load or memory consumption information could be measured using existing software tools or writing some simple scripts. For evaluation purposes comparison between different states of the given system under evaluation should be given.

The DROP-FAST system has the following components:

- Main server (protected server)
- DROP-FAST core (DNS and monitoring functions, controls the whole mechanism)
- Clients (accessing the main server)
- Cloud replica servers (synchronized with the main server)

As you can see from Fig. 5 the flow starts with the main server running in a standalone mode.

3.4.1 Detection

Resource monitor is software running on the DROP-FAST core server. Performance of the main server is monitored by getting periodic reports from the main server or via port mirroring and analyzing the whole traffic of the main server from the network switch. Standard IDS should be used for detection purposes. The IDS should be capable of identifying DDoS attacks as well as flash crowd events.

3.4.2 Replica Evocation

In case of resource exhaustion above the given threshold limits or if an attack is detected by the IDS, DROP-FAST core server will send commands to replicas on cloud and evocate them. This process is simultaneous with the change in DNS configuration. DNS configuration should be changed using BIND or different methods depending on the load sharing strategy chosen. The clients are instructed through DNS to send their requests to the cloud replicas and not to the main server.

3.4.3 Resource Monitoring

DROP-FAST core and individual replica servers continue to exchange health status messages. While system is still loaded and is using all available resources the replicas continue to be active. Number of replicas increases as the attack gets stronger. Nevertheless, attacks might be strong and can lead to high cloud resource consumption and large number of replicas activating. Some clients or subscribers will suffer from high bills for cloud services. Therefore, we propose to put an upper limit on the maximum amount of cloud resources to be used. Threshold should be chosen by the subscriber.

4. Evaluation

In order to evaluate our idea, we have set up an experiment. We used the following configuration:

- Hardware:
  1) DROP-FAST core: Interl(R) Pentium(R) D CPU 3.20GHz / 2GB RAM

2) Main server: Inter(R) Core(TM)2 Duo CPU E6750 @ 2.66GHz / 4GB RAM
3) Replica: Inter(R) Core(TM) i5-2500k CPU @ 3.30GHz / 3GB RAM
4) Client: Sony VAIO, Inter(R) Core(TM)2 Duo CPU P8800 @ 2.66GHz / 4GB RAM
5) Switch: ipTIME SW2401

- Software: OS - Ubuntu 10.04 , Server - Apache/2.2.14 (Ubuntu), DNS - BIND9, Virtualization - VirtualBox
- Synchronization and Replication: Dropbox

The three machines have different roles in the experiment:

1) Main server: URL is server1.dropfast.com
2) DROP-FAST core: controlling center that governs the operation of the system
3) Replica server: URL is replica1.dropfast.com, runs two virtual machines

The content on the main server is saved under the Dropbox directory, so that any changes made on replicas or on the main server are effective throughout the infrastructure. Using Dropbox eliminates our need to take care of synchronization among the different servers.

The replica server is a host for two virtual machines running servers that listen on ports 8081 and 8082. The traffic is distributed randomly among the two replicas using JavaScript.

The DROP-FAST server acts as a control tower and listens for resource exhaustion messages from the main server. It also incorporates the DNS functions. For simplicity of the experiment we put a threshold of 100 KBps for signaling an attack[1]. Once the threshold is passed DROP-FAST alters the DNS configuration and redirects part of the traffic to the replica server. This leads to the load reduction on the main server side.

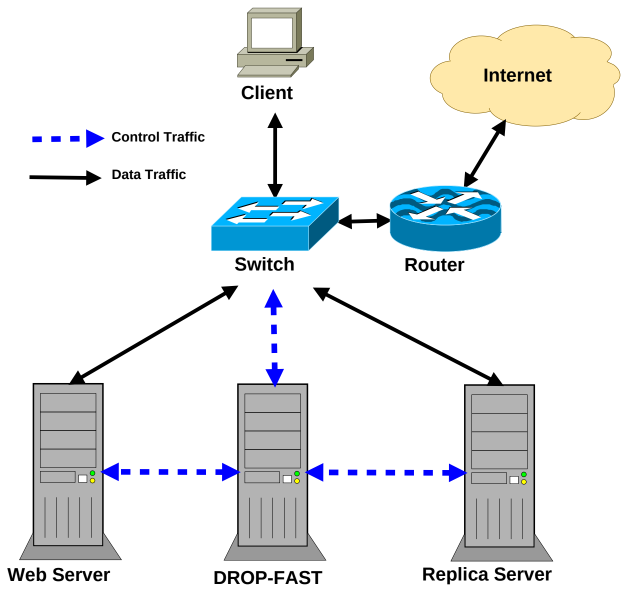

Fig. 6: Experimental Setup

[1]Threshold was chosen based on the attack tool that was used.

4.1 Modes of Operation

The system has two modes of operation:

1) Passive mode: system is operating under standard conditions with no attack or flash crowd event
2) Active mode: system is under attack or a flash crowd event occurs

In passive mode only the main server operates. DROP-FAST core runs and checks whether the incoming traffic is under the threshold value. The DNS records point to the main server with the URL server1.dropfast.com. All traffic flows in one direction only. Replica servers are not evoked.

The system switches to active mode when main server is under attack or a flash crowd event is detected. DROP-FAST core evokes the replica server hosting two web servers on virtual machines. DNS configuration is altered to enable traffic redirection to the replica server. Traffic is redirected based on load sharing policy. At this mode the traffic is distributed in several directions.

The experimental setup is schematically described in Fig. 6. The DROP-FAST core communicates with the main server and replicas internally as shown by the blue dashed arrows. This communication should not be visible externally and only the internal modules of the system must be aware of this communication. Insecure communication between the DROP-FAST core and the rest of the system poses a great risk to system security.

As a basic means of evaluation we measure the server load on the main server. First we monitor the main server load in the passive mode of operation to observe the incoming traffic rate. We then launch a HTTP flood attack. The server load is measured again to compare it to the load while the system was in passive mode. When main server is under attack DROP-FAST core changes the DNS configuration and the traffic is distributed between the main server and the replica server. One more measurement of the incoming traffic rate is made to check for any positive changes due to distribution of the load. You can see our results in Fig. 7.

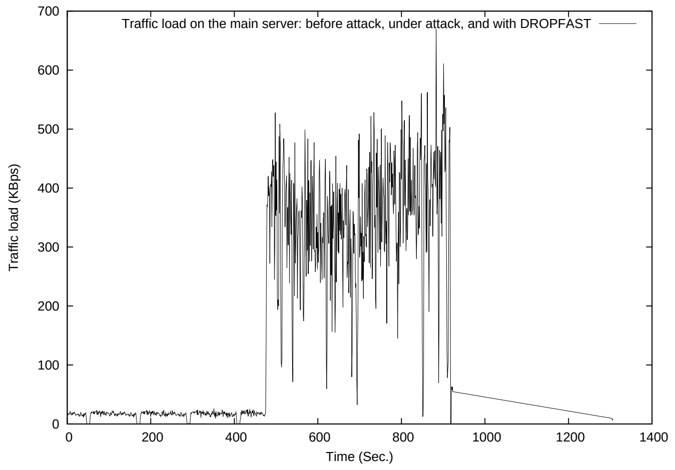

Fig. 7: Incoming traffic load on main server: passive mode, under attack, and active mode

The server load is stable while in passive mode. This means that no attack or anomaly is taking place. The system runs as intended and all the requests are handled by the main server. After 460 seconds a HTTP flood attack is launched. The load on the main server escalates quickly. This leads to DROP-FAST core activating the replica server and distributing the traffic. As a result of this action you can see how server load drops back to normal after approximately 900 seconds point.

Utilizing DROP-FAST made it possible to maintain normal response time even while under attack. Response time was monitored in passive mode, under attack, and in active mode. Results are presented in Fig. 8. Response time of main server is monitored while in passive mode. HTTP flooding attack is performed, hence rapidly increased response time is observed. Response time decreases rapidly as DROP-FAST is enabled.

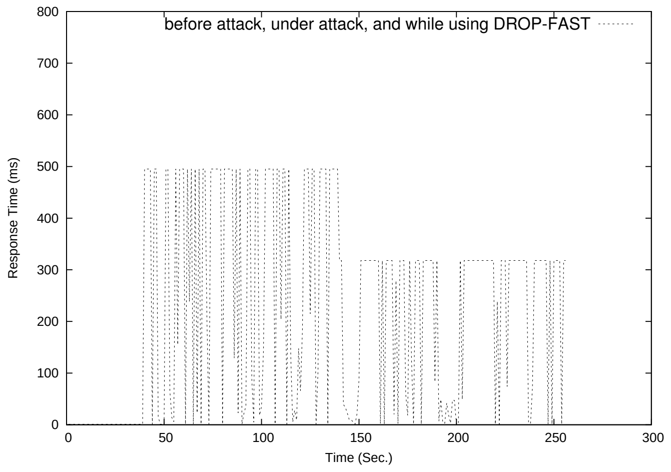

Fig. 8: Main servers response time: response time increases rapidly due to HTTP flooding at 40 seconds, response time remains high with lots of failed requests, response time reduces almost twice when DROP-FAST is applied.

It is important to note that we have performed a very basic experiment with only one replica server. The experiment was performed in a small lab network consisting of the three described computers only. The client accesses the servers via a Wireless LAN connection. Since the experimental topology is the possibly smallest one, it is impossible to make use of the advantage that is given by the use of best locations in network topology. We have only used one replica server running two virtual web servers. The system is stronger if more replica servers are available in advantageous network locations.

5. Conclusion

We have suggested a new approach to DDoS prevention using cloud technologies. We have stated several related works and elaborated on their pros and cons. Several important issues were discussed. These include but not limited to

- Replication methods,
- Synchronization,
- Load sharing,
- Resource monitoring,
- Security.

As a proof of concept we setup a basic experiment. Our network topology and network size are far from real internet like infrastructures. Nevertheless the results we have obtained suggest that DROP-FAST provides means to avoid DDoS by distributing the service. Experimental data shows improvement in server load and response time, proving viability of DROP-FAST mechanism. Experiments will be conducted to widen the idea further and apply the mechanism in real world networks.

In future, we plan to conduct further research in several directions:

1) Load Sharing: Develop sophisticated and efficient load sharing algorithm applicable to DROP-FAST.
2) Communication Protocol: Develop a secure and fast protocol to communicate among the main server, replica servers, and DROP-FAST core.
3) Topological Placement: Develop a method for optimal replica server placement decision for a given network topology.
4) Control Distribution: Develop a distributed control mechanism to substitute DROP-FAST core.

Acknowledgments

This research was supported funded by the R&BD Support Center of Seoul Development Institute and the South Korean government (WR080951).

References

[1] I. Muttik and C. Barton, "Cloud security technologies," *Information Security Technical Report*, vol. 14, no. 1, pp. 1 – 6, 2009. Malware.

[2] "Fingerprint sharing allience," tech. rep., ARBOR NETWORKS.

[3] E. Nygren, R. K. Sitaraman, and J. Sun, "The akamai network: a platform for high-performance internet applications," *SIGOPS Oper. Syst. Rev.*, vol. 44, pp. 2–19, Aug. 2010.

[4] I. CloudFlare, "An overview of cloudflare."

[5] Y. Gao, L. Deng, A. Kuzmanovic, and Y. Chen, "Internet cache pollution attacks and countermeasures," in *Proceedings of the Proceedings of the 2006 IEEE International Conference on Network Protocols*, ICNP '06, (Washington, DC, USA), pp. 54–64, IEEE Computer Society, 2006.

[6] M. Xie, I. Widjaja, and H. Wang, "Enhancing cache robustness for content-centric networking," in *INFOCOM, 2012 Proceedings IEEE*, pp. 2426–2434, March.

[7] "Availability challenge of cloud system under ddos attack," *Indian Journal of Science and Technology*, vol. 4, no. 6, pp. 2933 – 2937, 2012.

[8] H. Hamad and M. Al-hoby, "Article: Managing intrusion detection as a service in cloud networks," *International Journal of Computer Applications*, vol. 41, pp. 35–40, March 2012. Published by Foundation of Computer Science, New York, USA.

[9] P. Du and A. Nakao, "Ddos defense as a network service," in *Network Operations and Management Symposium (NOMS), 2010 IEEE*, pp. 894–897, April.

[10] V. C. Harish and B. Owens, "Dynamic load balancing dns," *Linux J.*, vol. 1999, Aug. 1999.