A simulation study of an application layer DDoS detection mechanism

AMIR RABIEE
ARAXI MEKHITARIAN
Preface

This bachelor thesis is conducted in collaboration with The Swedish Armed Forces. We would like to thank our supervisor Ross Tsagalidis from the The Swedish Armed Forces for guiding and providing us with relevant information throughout the whole period.

We would also want to express our deepest gratitude to our examiner Vladimir Vlassov and supervisor Hooman Peiro Sajjad for contributing with deep knowledge, discussions and guidance.
Abstract

Over the last couple of years the rise of application layer Distributed Denial of Service (DDoS) attacks has significantly increased. Because of this, many issues have been raised on how organizations and companies can protect themselves from intrusions and damages against their systems and services. The consequences from these attacks are many, ranging from revenue losses for companies to stolen personal data. As the technologies are evolving, application layer DDoS attacks are becoming more effective and there is not a concrete solution that entirely protects against them.

This thesis focuses on the available defense mechanisms and presents a general overview of different types of application layer DDoS attacks and how they are constructed. Moreover this report provides a simulation based on one of the defense mechanisms mentioned, named CALD. The simulation tested two different application layer DDoS attacks and showed that CALD can detect and differentiate between the two attacks.

This report can be used as a general information source for application layer DDoS attacks, how to detect them and how to defend against them. Furthermore the simulation can be used as a basis on how well a relatively small-scaled implementation of CALD can detect DDoS attacks on the application layer.

Keywords: Distributed Denial of Service attacks, DDoS, application layer, detection, defense
Abstract

Under de senaste åren har ökningen av Distributed Denial of Service (DDoS) attacker på applikationslagret ökat markant. På grund av detta har många frågor uppkommit om hur organisationer och företag kan skydda sig mot intrång och skador mot sina system och tjänster. Konsekvenserna av dessa attacker är många, allt från intäcksförluster för företag till stulen personlig data. Eftersom tekniken utvecklas, har DDoS attacker på applikationslagret blivit mer effektiva och det finns inte en konkret lösning för att hindra dem.

Denna rapport fokuserar på de tillgängliga försvarsmekanismer och presenterar en allmän översikt över olika typer av DDoS-attacker på applikationslagret och hur de är uppbyggda. Dessutom bidrar den här rapporten med en redovisning av en simulering baserad på en av de försvarsmechanismer som nämns i rapporten, CALD. Simuleringen testade två olika attacker på applikationslagret och visar att CALD kan upptäcka och skilja mellan de två attackerna.


Nyckelord: Distributed Denial of Service attacks, DDoS, application layer, detection, defense
**Terminology**

**Bandwidth** – The rate in which data is being sent to a server

**CALD** – A defense mechanism for application layer DDoS attacks

**DDOS** – Distributed Denial of Service

**DNS** – Domain Name System that is used to translate domain names to IP addresses

**Flashcrowd** – A massive amount of legitimate traffic generated during a short period of time

**Flooding** – Occupies every link to the server which constrains how well a server can respond to requests

**HTTP** – Hypertext transfer protocol

**HTTP GET** – Retrieving data from server

**HTTP POST** – Inserting/ update data from server

**HTTP request header** – Parameters set in request header for when sending and receiving messages

**INVITE packets** – Part of the SIP protocol to specify contact information

**IP address** – Identification of a computer using the Internet Protocol

**OSI-model** – Used as a model to describe computer communication in 7 different layers

**Proxy** – Used as a middle hand in requests from the client to the server

**Server** – Used to service clients in need of resources from the server

**Session** – A conversation between a client and a server

**Socket** – An endpoint through which two parts can communicate

**Spoofed IP** – Camouflaging as legitimate IP address with purpose to hide real identity

**Traceback** – Finding the originator of an IP address

**TCP connection** – Communication through an established protocol

**Unauthorized/non-legitimate users/clients** - Users that requests a massive amount of resources
# Table of Contents

1 Introduction .......................................................................................................................... 7  
   1.1 Background .................................................................................................................... 7  
   1.2 Problem ........................................................................................................................ 9  
   1.3 Purpose .......................................................................................................................... 9  
   1.4 Goal .............................................................................................................................. 9  
       Benefits, Ethics and Sustainability .................................................................................. 10  
   1.5 Methodology ................................................................................................................. 11  
   1.6 Delimitations .................................................................................................................. 11  
   1.7 Outline .......................................................................................................................... 12  

2 Application layer DDoS attacks and defense mechanisms ............................................. 13  
   2.1 Botnet – structure of DDoS attacks .............................................................................. 13  
   2.2 DDoS flooding attacks in the application layer ............................................................. 14  
       2.2.1 Reflection/amplification based flooding attacks ...................................................... 14  
           2.2.1.1 SIP flooding attack .......................................................................................... 15  
           2.2.1.2 DNS amplification flooding attack ................................................................. 15  
       2.2.2 HTTP flooding attacks ......................................................................................... 15  
           2.2.2.1 Session flooding attack ................................................................................. 15  
           2.2.2.2 Request flooding attack .............................................................................. 16  
           2.2.2.3 Asymmetric attack ....................................................................................... 16  
           2.2.2.4 Repeated One-Shot attack .......................................................................... 16  
           2.2.2.5 Slow request/response attack .................................................................... 17  
   2.3 Defense mechanisms for application layer DDoS attacks ........................................ 18  
       2.3.1 Anomaly detection ............................................................................................... 18  
       2.3.2 Destination-based defense mechanisms ............................................................. 18  
           2.3.2.1 DDoS-Shield ................................................................................................. 18  
           2.3.2.2 Clustering analysis ...................................................................................... 19  
           2.3.2.3 CALD .......................................................................................................... 19  
       2.3.3 Hybrid defense mechanisms .................................................................................. 19  
           2.3.3.1 Defense and Offense Wall ........................................................................... 20  
           2.3.3.2 AYAH .......................................................................................................... 20  

3 Detailed description of CALD .......................................................................................... 21  
   3.1 Abnormal traffic detection ............................................................................................ 21  
   3.2 DDoS attack detection ..................................................................................................... 23  
   3.3 Filter ............................................................................................................................... 25  

4 Method ............................................................................................................................... 26  
   4.1 Literature study .............................................................................................................. 26  
   4.2 Simulation ....................................................................................................................... 26  

5 Simulation of CALD ........................................................................................................... 28  
   5.1 Implementation .............................................................................................................. 28  
       5.1.1 Classes ................................................................................................................ 28  
       5.1.2 Definitions .......................................................................................................... 28  
   5.2 Simulation structure ...................................................................................................... 29  
       5.2.1 Test 1 - Repeated request application layer DDoS attack .................................... 29  
       5.2.2 Test 2 - Recursive request application layer DDoS attack .................................. 29  
       5.2.3 Test parameters .................................................................................................... 30  
   5.3 Environment .................................................................................................................... 30  

5
6 Evaluation ............................................................................................................................................. 31
  6.1 Deciding threshold value and traffic observation interval ...................................................... 31
  6.2 Analyzing entropy values ........................................................................................................... 34

7 Conclusions ...................................................................................................................................... 37

8 References ....................................................................................................................................... 38

Appendix ............................................................................................................................................... 41
1 Introduction

Over the last couple of years the digitalized industry has grown and the usage of Internet based services has increased immensely. Due to this progress, many challenges arise for keeping systems and services available and data secured against non-legitimate users.

Cyber-attacks are a type of offensive attacks carried out by individuals or organizations. These attacks can target computer information systems, computer networks and personal computers through malicious ways derived from an anonymous source that steals, modifies or destroys a target. There are many types of cyber-attacks with different purposes and it is a necessity to detect and prevent such attacks in order to keep data secured and maintaining systems and services available. [1]

1.1 Background

In the Internet society today, one of the major threats and one of the hardest security problems is the distributed denial of service (DDoS) attack [2]. A DDoS is a type of cyber-attack and is in simple terms an intensified variant of a Denial of Service (DoS) attack. This is an attempt to make resources or services unavailable to users by sending a large volume of traffic [3].

DDoS attacks can target several layers in the Open Systems Interconnection (OSI) model as shown in Figure 1. There are two layers where the majority of DDoS attacks happen; A) the network (third) layer and B) the application (seventh) layer. The DDoS attacks on the network layer, also called “Volumetric DDoS Attacks”, are the most occurring one [4]. As of this, defense mechanisms on the network layer are becoming more effective, the patterns are shifting and the attackers are focusing on the application layer.
In contrary to network-based DDoS attacks, application layer attacks are resource-based. In this case the perpetrators will copy legitimate user traffic behavior on the network-layer and can therefore bypass proxies and servers on the target side. This results in difficulties when trying to identify attackers and prevent DDoS attacks on the application layer. [2, p. 893]

When users (either attackers or legitimate clients) enter services on the application layer a session is created containing a minimum of one request [5]. These attackers in the application layer are harder to detect because of their ability to act as a flash crowd and disguise themselves as legitimate user traffic. With scenarios like these server resources such as disk bandwidth, database bandwidth and sockets are more prone to be attacked. [6]

Defense against application layer attacks can be divided in to four different classifications, attack prevention, attack detection, attack source identification and attack reaction. Attack prevention is based on preventing an attack before it reaches its final destination by filtering fake IP packets. When an attack occurs, the attack detection mechanism decides which direction to take before any further action is made. The task of attack source identification is to find the source of the attack even if the source is spoofed or containing inaccurate information. In this way, the victim can reduce the damage from DDoS attacks and deter potential attackers. The final category, attack reaction, is used for elimination or reducing the effects of a DDoS attack and try to distinguish between credible and bad traffic. [7]
1.2 Problem
Given the new techniques and technologies that have evolved, attackers are developing more refined methods to make resources unavailable to users. Because of these new techniques, DDoS attacks are increasing in size and frequency, which leads to more traffic and bandwidth generated for these purposeful attacks. In 2005 the bandwidth set aside for DDoS was at 10 Gbps, in 2009 at 49 Gbps and in 2010 at 100 Gbps. This represents an increase of 102% from 2009 to 2010 and a 1,000% increase from 2005 to 2010. [4]

DDoS attacks are also evolving in both strategy and tactics. By combining such attacks with infectious malware, attackers can breach unauthorized data such as bank accounts. These types of attacks can also lead to massive amount of revenue loss for companies that do not have the necessary approach to mitigate or prevent such an intrusion. According to Neustar Annual DDoS Attacks and Impact Report, 60% of companies had been DDoS attacked in 2013. And 14% of companies said that a DDoS outage would mean losses of 50k-100k dollars per hour while 29% reported over 100k dollars per hour. [24]

The problems reported above shows the great impact DDoS attacks have on organizations and companies. These attacks grow, evolve and increasingly adapt to the defense mechanisms that exist today. More specifically, application-layer DDoS attacks are relatively new and there is insufficient documentation on how to protect companies and organizations against these attacks.

1.3 Purpose
Application layer DDoS attacks are growing in size and being costly for companies and organizations. There is not a concrete and effective way to defend against these attacks. This thesis will provide a better overview of the existing application layer DDoS attacks and how to detect, mitigate and prevent them. The main purpose of the thesis is to implement an application layer DDoS defense mechanism called CALD. This is done in order to show how this specific approach can detect two types of DDoS attacks on the application layer.

1.4 Goal
The goal of this thesis is to give an overview of application layer DDoS attacks for interested parties such as private individuals, organizations and companies. This information could be used to answer questions about application layer DDoS attacks such as, how to detect, mitigate and prevent them, how they are structured and what different types of attacks there are.
The goal of the simulation of CALD is to provide a better insight on how to construct and implement a small-scaled application layer DDoS attack detection mechanism. The implementation can be used as a preliminary test experiment for companies when developing real defense mechanisms for application layer DDoS attacks.

Benefits, Ethics and Sustainability

This thesis will help students that are keen to learn more about what an application layer DDoS attack is, and the different methods that can be used to execute such an attack and defend against them. Moreover authoritative personnel in need of better insight on the tools/mechanisms presently available on how to mitigate such attacks will benefit from this project. Reports shows that a DDoS attack can cost up to 440,000$ for a company, therefore protecting against intrusions and securing valuable and private information is essential [8]. The usage and evaluation of the defense mechanisms that are currently developed to protect against DDoS attacks can also be beneficial for companies being damaged by such attacks. This can lead to better ways of defending services offered by companies and therefore maximize revenue.

There are several ethical issues raised when working with this thesis and one of the most critical is the availability of methods on how one can generate a DDoS attack. This causes a rather infectious debate on whether to expose such harmful tools on the Internet that can be used to instigate malignant attacks or not, as the tools can be used to harm services and systems. As of this, the project could lead to ethic issues on how to use these tools for a greater purpose. Many of these programs that have been used in major takedowns of servers and websites are also easily configured for the most novice computer-user [9]. Exposing such tools to users with malicious goals can lead to a more hostile outlook towards protecting oneself against DDoS attacks as it gives incentives for attacking rather than defending.

Detecting a DDoS attack in real time and defending against them could also lead to better economic and environmental sustainability. A DDoS attack exhausts the available servers by using up the resources provided by them. Therefore a way to defend against such attacks would be to purchase enough material (servers, routers etcetera) [10]. This can in turn consume further resources in order to manufacture them, which in extension is not a sustainable way to deal with defending against DDoS attacks.
1.5 Methodology

A quantitative research method was used in this thesis, which was based on gathering and processing data. The quantitative method was divided into a literature study and a simulation.

The literature study focused on general information of different types of DDoS attacks on the application layer, the structure of DDoS attacks and collection of papers on existing defense approaches. These papers were retrieved from digital libraries such as IEEE, ScienceDirect, ACM Digital Library and DiVA. Available papers online from IT-companies were also used in the literature study. These papers gave a perspective on the financial impact that DDoS attacks has on organizations and companies.

A simulation with two tests was done with one of the DDoS defense approaches mentioned in Chapter 2. This was made to test if the mechanism was able to detect different types of DDoS attacks.

1.6 Delimitations

A delimitation in this thesis that needs to be taken in consideration is the focus of only one category mentioned in section 1.1. The chosen category is attack detection, which is a vital step in defending against application layer DDoS attacks.

When developing and testing application layer DDoS attacks, earlier reports such as [11-12] had access or/and gathered application layer DDoS attack datasets/logs from web sites. Simulating a full advanced application layer DDoS attack needs this type of resources and that is not obtainable for this thesis work. Creating an advanced application layer DDoS attack requires a botnet that needs several thousand infected computers to get a real DDoS effect [22, p. 39]. This is also a resource limitation in this thesis.
1.7 Outline

This thesis is structured as such,

- Chapter 2 – Gives general background information on the existing application layer DDoS attacks available today. It also provides different types of application layer DDoS defense mechanisms against these attacks on the application layer.

- Chapter 3 – Gives an in-depth insight and analyses one of the detection mechanisms mentioned in Chapter 2.

- Chapter 4 – Presents the methods that was used in order to conduct the thesis

- Chapter 5 – Presents the simulation of the defense mechanism, named CALD

- Chapter 6 – The results from the simulation in Chapter 5 is presented here

- Chapter 7 – Evaluation of the results from Chapter 6

- Chapter 8 – Final conclusions and thoughts such as improvement possibilities are discussed here
2 Application layer DDoS attacks and defense mechanisms

In this chapter, an overview on how a DDoS attack is constructed is presented. Furthermore, several DDoS attacks and defense mechanisms are presented here.

2.1 Botnet – structure of DDoS attacks

Several machines focusing on a specific target carry out a DDoS attack. These machines form a structure represented as an interconnected network called botnet.

A botnet consists of bots, which are computers that have been infected with malicious malware by a botmaster. The botmaster can control and communicate with the bots in the botnet by sending commands with purpose to launch a DDoS attack on the victim. [13] Figure 2 shows the whole architecture of a DDoS attack.
2.2 DDoS flooding attacks in the application layer

Application layer DDoS flooding attacks can occur in many parts of an application such as DNS, SIP and HTTP. In this part of the thesis the focus is on two categories, reflection/amplification based flooding attacks and HTTP flooding attacks.

2.2.1 Reflection/amplification based flooding attacks

A Distributed Reflector Denial of Service (DRDoS) is a form of DDoS attack, which hides the sources of the attack traffic of the botnet by using middle hands such as routers or Web servers. These middle hands are also called reflectors and are used to sending replies to the victim. By adding them, the DRDoS attack traffic becomes more distributed and amplified than a traditional DDoS attack and makes it harder to identify and traceback. [7, p. 12] The structure of a DRDoS attack is shown in Figure 3.

![Figure 3 - Structure of a DRDoS attack](image-url)
In the following sections two types of amplification attacks is mentioned, SIP flooding attack and DNS amplification attack.

2.2.1.1 SIP flooding attack
Session Initiation Protocol (SIP) is used for communication sessions such as Voice over IP (VoIP) applications. SIP is normally implemented on top of User Datagram Protocol (UDP) for simplifying communication between VoIP clients. The proxy servers using SIP requires open Internet access for connection between the clients in purpose to establish call setup requests for any client. One example of SIP DDoS attack is when the attacker floods the SIP proxy server with INVITE packets. Depending on if it is a network layer DDoS attack or application layer, there are two different ways of flooding. The former method uses spoofed source IP addresses while the latter uses non-spoofed. For this example mainly two victims can be targeted, either the SIP proxy server or the call receivers. If the proxy server is attacked, handling the SIP INVITE packets will exhaust the server resources and the network capacity. When a caller is attacked, he will be overflooded by VoIP calls and will not be receivable for legitimate clients calling. [7, (p. 11, 14)]

2.2.1.2 DNS amplification flooding attack
A common type of DDoS attack is Domain Name Server (DNS) amplification attack. The purpose of this attack is to flood the victim’s resources with response traffic from the DNS servers. A botnet initiates the attack by spoofing the victim’s IP address. Afterwards the botnet makes a DNS naming lookup request to a public DNS server, which sends a DNS response to the target. By doing so, the DNS server replies to the victim instead of the botnet. The size of the request made by the bots in the botnet is smaller than the response sent from the DNS server to the victim, therefore the attacker increases the amount of traffic against the victim. With the use of a botnet, a huge number of spoofed DNS queries are done, and the attack generates a massive volume of traffic against the target. The DNS amplification flooding attack is hard to prevent because the responses are from valid servers. [15]

2.2.2 HTTP flooding attacks
There are five types of HTTP flooding attacks; session flooding attack, request flooding attack, asymmetric attack, repeated one-shot attack and slow request/response attack. These attacks are described below.

2.2.2.1 Session flooding attack
A well-known session flood attack is the HTTP flooding attack, which is when HTTP GET or POST requests are sent to a web server or an application with the intent to make it unavailable. The traffic that is sent during these attacks is
often generated by a botnet. The traffic from the botnet resembles legitimate user traffic by establishing a valid TCP connection with an authentic IP address. The request rates from the attackers are higher than those from legitimate users; this consumes the servers resources e.g. sockets, CPU or memory and makes it unavailable. [16] By creating HTTP requests in different ways the attackers can maximize the attack power or avoid discovery [14].

For example the World Wide Web (WWW) is one of the most popular applications that is running on the Internet and is frequently under threat of DDoS attacks. A botmaster can direct the botnet to send HTTP requests on a target web page on the Internet, interpret the replies and recursively follow the links. By doing this, it is very difficult for the victim to protect himself because the HTTP request traffic sent by the botnet is very similar to ordinary traffic on the World Wide Web. [7, p. 11]

2.2.2.2 Request flooding attack
Attackers send requests faster than normal clients and take advantage of sending multiple requests per session. By limiting the request session rates the attackers can dodge defense mechanisms, which control the session-rate limitation. A famous example in this category is the single-session HTTP GET/POST flooding attack, which takes advantage of HTTP 1.1 that permits numerous requests within a single HTTP session. [17]

2.2.2.3 Asymmetric attack
In contrast to flood attacks asymmetric attacks use heavy resource requests which cause more damage for each established session. Each request damages the server by sending further heavier workload than the last one. Compared to request flood attacks, the rate for each request does not need to be higher than normal, the aim is to send more difficult requests each session to be processed by the server. An example is an asymmetric DDoS attack using complex database queries. The asymmetric attack requires less work and is sent in a lower request rate, which makes this attack more difficult to detect. [17, p. 26-27] The main purpose of this attack is to consume resources like CPU and memory and results in impairing the quality of service [18].

2.2.2.4 Repeated One-Shot attack
Repeated one-shot attacks are derivatives of asymmetric attacks. In contrary to asymmetric attacks where it has multiple requests per session the attacker distributes one heavy request per session where the session rate is higher than normal. By doing this, the attacker benefits from the ability to close sessions directly after a request is sent and thus evades detection. [17, p. 26-27] The repeated one-shot attacks can better avoid detection than flooding attacks and
asymmetric attacks. But the main purpose of this attack is the same as the ones mentioned above, to degrade the server and to hinder access to legitimate users. [18, p. 25]

### 2.2.2.5 Slow request/response attack

There are many different types of slow request/response attacks, a couple are listed in the following paragraphs.

A **Slowloris attack** shuts down a Web server by using HTTP GET requests with a single one or a limited set of machines. The requests made by the attacker do not include a complete set of HTTP request headers. These requests constantly increase and updates in number and never closes, this results in sockets being occupied by the attacker, which leads to unavailability of the server. [16, p. 4]

A **HTTP fragmentation attack** has the same purpose as the Slowloris attack. This method can be carried out by initiating multiple sessions on each bot in a botnet and in this way shut down a Web server by occupying the connection slots for a long time without being detected. Instead of changing the headers of a HTTP packet as in the Slowloris attack, the attacker establishes a legitimate connection to the server and then fragments the valid HTTP packets into smaller pieces. These small fragments will be sent back in the slowest pace that the server time out tolerates. [16, p. 4]

**Slowpost attack** (also named R-U-Dead-Yet (RUDY) or slow request body attack) is very similar to Slowloris attack. It sends HTTP POST commands slowly to shut down a Web server in which the attribute of the packet called Content-Length Header value is large. By having this specified, the target server thinks the attacker is going to send a lot of traffic and keeps the connection open. The attacker will then send traffic at a slow rate until every byte has been sent specified in the attribute. [16, p. 4]

**Slowreading attack** (or slow response attack) works in the opposite way from the slowpost attack. Instead of sending requests slowly as in slowpost attack, the attacker rather slowly reads the responses by having a smaller receiving widow-size compared to the server’s send buffer. Even if no data is being sent or read the connection is still maintained, therefore the attacker can obtain a lot of connections open and eventually cause a DDoS attack. [16, p. 4]
2.3 Defense mechanisms for application layer DDoS attacks

In this chapter a brief explanation of anomaly and anomaly detection is presented. Thereafter two categories of defense mechanisms for application layer DDoS attacks are described, these are called destination-based defense and hybrid defense mechanism.

2.3.1 Anomaly detection

According to Varun et. al anomaly detection “refers to the problem of finding patterns in data that do not conform to expected behavior.” [19]. These abnormal patterns are often referred to as anomalies or outliers. Anomaly detection is utilized in many different areas such as credit card fraud, insurance, military and intrusion detection in cyber security.

In anomaly detection it is essential that the anomalies from the gathered data can extract critical and necessary information in many different areas. An example of when to use anomaly detection is when abnormal traffic from an infected / hacked computer is sent to a destination.

An anomaly detection approach can be observed as a defined area where there is normal data, and everything outside of that area is counted as an anomaly.

2.3.2 Destination-based defense mechanisms

Destination-based defense mechanism is front-end based and responsible for two things, detecting and reacting on the victim side. This defense mechanism can be deployed in the edge routers, access routers of the victims’ destination system or reverse proxies. The advantage of this mechanism is the relative easiness of implementing it. Anomalies in the system can be detected by observing incoming traffic and model it. [16, p. 8] In this section some of the destination-based defense mechanisms are mentioned with regards to the structure and design.

2.3.2.1 DDoS-Shield

DDoS-shield is a destination-based defense mechanism against application layer DDoS attacks developed by Ranjan et. al. [17]. This mechanism is incorporated in to a reverse-proxy [17, p. 28]. DDoS-shield works with statistical tools for detecting specific HTTP session attributes and protect against attackers that uses methods that limits legitimate client traffic.

The DDoS-shield is divided in to a suspicion assignment mechanism and a DDoS-resilient scheduler [17, p. 26-27]. The suspicion mechanism uses elements such as session rate, request rate and workload of each session from legitimate users and compares them with incoming traffic. These steps bring a
continuous suspicion value. The scheduler checks the suspicion value and selects if the traffic is legitimate and should pass the shield.

2.3.2.2 Clustering analysis
Ye et. al. [12] presents a clustering analysis method for data-gathering for application layer DDoS attacks. The method focuses on different features such as request rate, average size of objects requested in the session, average popularity of all objects in the session and average transition probability [12, p. 1038]. This is used to extract normal user behavior. The incoming data are categorized by clustering it from the users’ sessions, which extracts a model. This model can distinguish if incoming traffic is legitimate or attack traffic.

2.3.2.3 CALD
CALD is a detection mechanism used against several types of DDoS attacks imitating flash crowds [11]. It can distinguish between four types of abnormal traffic: repeated request DDoS, recursive request DDoS, repeated workload DDoS and flash crowd.

CALD is divided into three different phases. The first phase is the abnormal traffic detection, which is implemented in a front-end sensor. When this phase detects abnormal traffic an “ATTENTION signal” is sent to the next phase, which is the DDoS attack detection. When the ATTENTION signal reaches the DDoS attack detection, this phase calculates the frequency of the incoming source IP address and its visited web site and items. In this way an estimated average frequency of the resources, such as pictures and webpages on the website is calculated.

When the frequency is calculated the entropy can be decided which also is named as mess extent. The value of the entropy can determine what type of DDoS attack it is or if it is a flash crowd. The last phase is filtration. This will filter and remove the non-legitimate IP addresses while legitimate traffic continues to have access to the web server.

2.3.3 Hybrid defense mechanisms
Hybrid defense mechanisms are positioned over several locations such as source, target and transitional network. This technique is a collaboration between clients and servers to prevent the attacks and accomplish detection and defense. For example detection can be implemented at the target side and response at other nodes in the network. Some hybrid defense mechanisms are mentioned below. [16, p. 10]
2.3.3.1 Defense and Offense Wall
The Defense and Offense Wall (DOW) is used for detecting and filtering request flooding attacks and asymmetric attacks. It is based on an anomaly detection method called K-means clustering [20, p. 2-3]. It is divided into three stages. First the K-mean clustering method is used to create a normal client behavior profile with legitimate traffic data. Second, a cluster distance based method is used for detecting abnormalities. Third, the filter removes suspicious traffic sessions.

2.3.3.2 AYAH
Sivabalan et. al. [21] introduces an approach for detecting and defending against DDoS attacks based on signature detection and Are You A Human (AYAH) Webpage. Every user accessing the server will have its behavior analyzed. Thereafter a signature generator will be used to generate a signature, based on the web pages request rates. When multiple requests are sent to the server, verification can be made because the user request has an embedded signature. By doing this, legitimate and attack traffic can be separated. If the server is in a state above a specific threshold called the Low Load Threshold, the users will be sent to an AYAH webpage [21, p. 580]. The response from this webpage can determine if the user is an attacker or not.
3 Detailed description of CALD

Chapter 2 describes various mechanisms to defend application layer DDoS attacks, however not from an in-depth perspective. From the listed defense mechanisms from section 2.3, CALD was chosen and implemented. CALD was chosen because its intended main purpose is to detect application layer DDoS attacks. In the following sections a detailed description of the structure and all components of CALD is presented.

As mentioned before in 2.3, CALD is a detection mechanism whose purpose is to protect Web services against application layer DDoS attacks masquerading as flash crowds [11, p. 1]. CALD is divided into three phases which can be seen in Figure 4 which is based on a figure taken from CALD.

![Figure 4 shows the structure for CALD [11, p. 248]](image)

Phase one is abnormal traffic detection, phase two is DDoS attack detection and phase three is filter. These are described below.

3.1 Abnormal traffic detection

The abnormal traffic detection is the first phase/function of CALD and is implemented in a front-end sensor [11, p. 1]. The main purpose of this function is to detect sudden changes in HTTP GET requests, i.e. anomaly detection, sent to the front-end sensor. This function does not take any action if no anomalies are detected. If abnormal information is detected from the incoming HTTP traffic, an "ATTENTION" signal is sent to the next phase (DDoS attack detection), which further analyses the data and makes a decision. The traffic received is used to distinguish between different types of
application-layer DDoS attacks and flash crowds. Several steps are taken before sending an ATTENTION signal, these are described below.

The first measurement is to analyze the incoming traffic. This can be done in many different ways but CALD predicts traffic intensity by using an Auto Regression model (AR model). In regression, previous values have an effect on future values, therefore the AR model uses previous observed traffic to predict the change of traffic intensity in the future.

Initially, the HTTP GET traffic stream is monitored. A time series \{y_1, y_2, \ldots, y_t\} is formed by the traffic intensity which are studied in constant time intervals. The traffic intensity is calculated in this thesis “by the total number of packages received in a time interval” [22, p. 39]. The traffic intensity is predicted from earlier observations with the help of the AR model. If major changes are detected, it can potentially be an application-layer DDoS attack or a flash crowd. The AR model that predicts the current traffic intensity is:

\[ y_t = \sum_{k=1}^{p} a_k x_{t-k} + e_t \]  \hspace{1cm} (1) \hspace{1cm} [22, p. 39]

The variable \( y_t \) is the prediction of \( x_t \), which is the observation value at a certain time \( t \). The variable \( a_k \) is a “stationary model parameter”, which means that it does not vary when time changes, and \( e_t \) is the observation error. [22, p. 39]

Secondly, at a certain time \( t \), the difference between the observation \( x_t \) and the prediction \( y_t \) gives the residual or model error \( d_t \), which can be seen in equation (2). [11, p. 3] From that specific residual under time \( t \), a standard deviation \( \sigma_d^2 \) can be calculated, as seen in equation (3).

\[ d_t = |y_t - x_t| \]  \hspace{1cm} (2) \hspace{1cm} [11, p. 249]

Thirdly the standard deviation \( \sigma_d^2 \):

\[ \sigma_d^2 = \frac{\sum_{i=t-p}^{t} (d_i - \text{avg}(d_{t-p}))^2}{p} \]  \hspace{1cm} (3) \hspace{1cm} [11, p. 250]
Now, a threshold seen in equation (4) can be made for determining if the traffic is abnormal or not. If \( d_t \) is higher than \( k \sigma_d^2 \) abnormal traffic is detected and an ATTENTION signal is sent to the DDoS attack detection phase. In the opposite case when no abnormal traffic is detected, the abnormal traffic detection phase sends a DISMISS signal to the DDoS attack detection function which inactivates itself. The constant \( k \) adjusts the sensitivity of the threshold and is not set to a specific value.

\[
d_t > k \sigma_d^2 \quad (4) \quad [11, \text{ p. 250}]
\]

### 3.2 DDoS attack detection

The DDoS attack detection is the second phase of CALD. When an ATTENTION signal is received, the DDoS attack detection is activated. By calculating the entropy or mess extent of the incoming traffic, this function can decide what type of DDoS attack occurred, or if there is a flash crowd. In order to calculate the entropy and distinguish between different attacks and flash crowds several steps are taken.

To understand the meaning of mess extent/entropy, the definition is, "suppose there is a set \{n\}, if the elements in set \{n\} are positioned dispersedly, then the Mess Extent is higher. Otherwise, if the elements in set \{n\} are converged in some points by any organized form, the Mess Extent is lower and close to 0" [11, p. 251]. The dispersion is equivalent to how many different resources being requested, i.e. requests from different source IPs to different resources will show a higher mess extent value than a request to one resource only.

CALD can detect four types of abnormal traffic described below.

1. **Repeated Request application layer DDoS attack**: the focus is mostly on one or two resources on a specifically selected website.
2. **Recursive Request application layer DDoS attack**: the bots attack the same number of resources on several different web pages, which means that traffic is spread in different directions but continues to focus on the same resources at each attack.
3. **Repeated Workload application layer DDoS attack**: this attack uses fewer bots but hurt the website even more. Its main objective is to constantly request large images and database searches.
4. **Flash crowd**: is when huge number of legitimate users visit a website. This creates a wave of visitors known as flash crowds. Since legitimate users usually interest in a specific site or resource the HTTP GET requests of flash crowds are spread.

[22, p. 40]
In order to process the traffic in real time a frequency vector called \( RFV \) is created for every resource on the website, as well as for every source IP address with the specific resource that is requested \([22, p.38]\). There are \( i = 1 \) to \( m \) resources, such as images and webpages, in the website. This means that the frequency for resource 1 in the website is \( favg^1 \) in the \( RFV \), for resource 2 is \( favg^2 \) until resource \( m \) which is \( favg^m \). The \( RFV \) can be seen in equation (5). The variable \( n \) in equation (6) indicates how many times a source IP address requested a specific resource.

\[
RFV = \{favg^1, favg^2, ..., favg^m\} \quad (5) \quad [22, \text{p. 38}]
\]

\[
favg_n = \left( \frac{1}{T_2-T_1} + \frac{1}{T_3-T_2} + \cdots + \frac{1}{T_n-T_{n-1}} \right)/n \quad (6) \quad [11, \text{p. 250}]
\]

For identifying whether it is a DDoS attack or a flash crowd, the entropy for the abnormal traffic is calculated in equation (7) and then compared in equation (8). The variable \( S \) is the RFV of source IP addresses and \( T \) is the URLs of the Web pages required by the attackers and clients. In CALD the mess extents gives the distribution of the sources \( S \) and the targets \( T \). CALD periodically calculates the mess extent of the abnormal traffic.

The formula for calculating the entropy is:

\[
En = \sum_{i=1}^{m} favg_n^i \log (favg_n^i) \quad (7) \quad [22, \text{p. 39}]
\]

By comparing the abnormal traffic received from phase 1, and determining if a certain type of DDoS attack or flash crowd has occurred, can be seen in equation (8). The entropy values are numbered as: 1) Repeated Request application layer DDoS attack, 2) Recursive Request application layer DDoS attack, 3) Repeated Workload application layer DDoS attack and 4) Flash crowd.

\[
\frac{En(S)_2}{En(T)_2} > \frac{En(S)_1}{En(T)_1} > \frac{En(S)_3}{En(T)_3} > \frac{En(S)_4}{En(T)_4} \quad (8) \quad [22, \text{p. 41}]
\]
3.3 Filter

The last phase of CALD is the filter. After calculating the entropy from equation (8), if the value of the entropy for a specific source IP is indicated as a DDoS attack, the IP address is seen as anomalous. As seen in Figure 4, when abnormal traffic reaches the filter the anomalous IP address gets dropped and legitimate IP addresses passes to the web server. This phase uses Bloom filter for determining which source IP addresses that are going to be dropped or continued to the Web server, but this part of CALD is not taken in consideration in this thesis.
4 Method

There are mainly two types of research methods for writing a scientific report, a quantitative and a qualitative [23, p. 1]. For this thesis a quantitative method was chosen. A quantitative method includes collecting and processing larger data sets; therefore this research method is the most eligible.

In contrast to a quantitative method, qualitative methods are usually used when exploring new research areas. This method includes interviews and case studies. This research method was therefore not used in this thesis.

Furthermore, the quantitative research method was divided into two parts, a literature study and a simulation of CALD with two tests described in section 5.2.1 and 5.2.2.

4.1 Literature study

The literature study that was carried out, aimed at gathering information on existent defense mechanisms and how the attacks are structured. In order to obtain valid and credible information on defense mechanisms and application layer DDoS attacks, academic articles were found from digital libraries such as, IEEE Xplore Digital Library, ACM Digital Library, ScienceDirect and DiVA. Articles published in these databases are all peer-reviewed, which means that a certain standard and quality has been reached.

The information collected from the databases that provided the relevant data for the thesis was done in search iterations. The iterations were designed and structured in a way that gave the maximal relevance for the thesis. Keywords such as “DDoS”, “Distributed Denial of Service attacks”, “application layer”, “detection” and “defense” were used throughout the iterations.

Papers such as [4] and [24] published by companies with IT-security background were also read and used. This was done in order to understand the financial drawbacks of DDoS attacks on organizations and companies.

4.2 Simulation

The quantitative research method required data to be gathered and therefore a simulation on application layer DDoS attack was done. To obtain data from application layer DDoS attacks, a simplified implementation of CALD was conducted. The simulation on application layer DDoS attacks were based on one of the defense mechanisms, CALD, and if it can detect different types of DDoS attacks on the application layer.
There are many simulation tools available for creating DDoS attacks that could have been used in the implementation of CALD. Some examples of these tools are DDoSim, Low Ion Orbit Cannon and High Ion Orbit Cannon [25]. The simulation in this thesis did not use any of these tools because some of them did not focus on the application layer and others were excluded because they did not work correctly. The tools that did work did not provide relevant data for the simulation of CALD. Therefore an implementation using the programming language Java was made.

Socket programming in Java was chosen for the implementation of CALD because of the Socket and ServerSocket classes. These classes establish a communication link between machines using the TCP protocol. The Socket class creates a client socket and is an endpoint for communication with the ServerSocket. An instance of ServerSocket listens for incoming requests from client sockets. This in turn processes the requests and may return a response back to the client. All sockets send and receive data through the I/O streams classes, which are two buffers named InputStream and OutputStream. The client socket’s OutputStream are connected to server socket’s InputStream and vice versa. [26]
5 Simulation of CALD

The simplified implementation considers only two types of abnormal traffic; repeated request application layer DDoS attack and recursive request application layer DDoS attack.

5.1 Implementation

This chapter gives a description of classes that were used and how some terms were defined for the simplified implementation of CALD.

5.1.1 Classes

Java ServerSocket and Socket classes from the Java Library were used. The classes are relatively easy to implement and can create an authentic network traffic connection between a Web server, clients and bots. The network connection uses the TCP protocol instead of UDP because it is connection oriented which application layer DDoS attacks demand.

The interaction between the users, bots and the server are all multithreaded by creating them as Thread objects from the Thread class. Because of this the Runnable Interface was implemented for the classes that represented the clients and bots.

5.1.2 Definitions

The following definitions were used for the simulation.

*Website*

The server provides resources from a potential website, these resources are numbered as 0, 1 and 2 and can be requested both from the bots and clients. These resources represents pictures that are downloaded from the website.

*Server*

The server creates a so-called “ClientHandler” for every established connection from clients and bots. This handler services every request from clients and bots made to the Web server. The server is used in a one-way communication messaging conversation. This means that the clients and bots will only be used for requesting resources on the server and not receiving a reply.

*Legitimate clients*

In reality, one and the same client can request multiple resources from a website before it breaks its connection to the server, however in this
implementation that is not taken into consideration. The client requests a single resource once, and then breaks the connection with the server.

*Bots*
In this implementation the bots can request either multiple resources or a specific one depending on what type of attack being carried out. The bots requests at a faster rate than legitimate clients.

*Traffic observation interval (TOI)*
Measures the incoming amount of requests from the attackers and clients during a specified time interval.

### 5.2 Simulation structure
The simulation implements the first and the second phases from CALD, the third is not implemented because the goal of the simulation is to detect when a DDoS is occurring and not filter anomalous IP addresses.

The implementation during the first phase, abnormal traffic detection, logs every incoming request in a 1 second interval. It calculates the amount of requests, and compares it with the threshold value. If the current request amount is higher than the threshold, abnormal traffic is detected and the second phase is activated.

The second phase will start logging vital information such as timestamps for every request made by both the clients and bots as long as abnormal traffic is detected. These timestamps will be stored until the simulation is over. After the simulation the entropy for the source IP and target resources will be calculated for deciding if the abnormal traffic was a DDoS attack or not.

#### 5.2.1 Test 1 - Repeated request application layer DDoS attack
Every client and bot repeatedly request for the same heavy resource. The chosen resource is picture 0 in this simulation. For normal clients the rate for requests is lower than for the attackers.

#### 5.2.2 Test 2 - Recursive request application layer DDoS attack
Every client and bot repeatedly request 3 resources, picture 0, 1 and 2. The rate of requests for normal clients is lower than for the attackers.
5.2.3 Test parameters
These are the parameters used for Test 1 and Test 2.

- Duration: 300 seconds
- Legitimate clients: 50
- Attackers: 2
- Clients request once between 10-20 seconds
- Attackers requests every 300-500 millisecond
- Threshold for mess extent: 5

5.3 Environment
The environment used for the tests made for this project was all running on the same experiment setup. The experiment setup consisted of one computer with the following hardware and software.

### Hardware

<table>
<thead>
<tr>
<th>CPU</th>
<th>Intel Core i5 1,3 Ghz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>4GB 1600 Mhz DDR3</td>
</tr>
</tbody>
</table>

### Software

<table>
<thead>
<tr>
<th>Language</th>
<th>Java 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>OS X El Capitan version 10.11.5</td>
</tr>
</tbody>
</table>
6 Evaluation

In this chapter the results and evaluation from the tests done in section 5.2.1 and 5.2.2 are presented. In section 6.1 the data for deciding the threshold and TOI value for detecting abnormal traffic is shown and explained to why they were chosen. In section 6.2 the entropy values for the source IP addresses and the target resources are calculated and analyzed.

6.1 Deciding threshold value and traffic observation interval

Before simulating CALD and testing the two different attacks, two interconnected parameters had to be decided, the threshold and the TOI value. The threshold and TOI value chosen for the tests in section 5.2.1 and 5.2.2 was determined from Figure 5 and 6. For deciding these values Test 1 was chosen but Test 2 could have been used aswell, which DDoS attack being tested does not affect the threshold and TOI value.

Figure 5 shows how long time it took to detect abnormal traffic for the first time with five different threshold values and with different TOIs. The threshold value is used for comparison to know if abnormal traffic has been detected. This data was calculated without any attackers involved.

Elapsed time until abnormal traffic is detected without attackers
One observation from Figure 5 is that the threshold values 1 and 2 can be seen in all runs of Test 1 with the different TOIs. What can be drawn from this is, that lower threshold values leads to a high sensitivity for when to detect abnormal traffic, despite the TOI value. Some abnormalities from Figure 5 can be seen such as threshold value 2 with TOI 1 and threshold value 3 with TOI 14.

The threshold value 3 is shown in a few runs with specific TOIs, these are 15-13 and 10-9 seconds. This means that the sensitivity for abnormal traffic is dependent on what the TOI is set to for this value. This indicates a correlation between the threshold values and TOIs. The threshold value 5 and 4 are not represented in the diagram. This means that these values were too high regardless of the TOI used.

The elapsed time for detecting abnormal traffic is different depending on which threshold value that was used. We know that Figure 5 only shows legitimate traffic. In order to let legitimate traffic pass through, the threshold value needs to be lower than the amount of requests from legitimate traffic.

Figure 5 shows that when having only legitimate incoming traffic, threshold values higher than 3 are not shown in the graph. This indicates that when having higher threshold values than 3, legitimate traffic is not considered abnormal and can proceed to the website without any problems.

With regards to what is mentioned above, the threshold value needs to be higher than 3 for detecting abnormal traffic generated from legitimate and non-legitimate clients.

One abnormality that was measured in Figure 5 is for the threshold value 2 and TOI of 1 second that spiked up to 175 seconds in several measurements. One possible explanation for this, is that the specified threshold value was not reached by incoming traffic in that specific TOI.

Figure 6 shows how long time it took to detect abnormal traffic with five different threshold values and with different TOIs. This data was calculated with attackers.
We can observe from Figure 6 that the time it took to detect abnormal traffic with the attackers and legitimate clients was proportional to the value that TOI was set to. This meant that these TOI values was sufficient for detecting abnormal traffic the first time. This was repeated for all TOI values except 4, 2 and 1.

We know that the first phase in CALD needs to detect abnormal traffic as soon as possible. If the TOI is too large, there is a risk that abnormal traffic is lost during the interval. Therefore TOIs higher than 4 are considered too high.

Figure 6 shows that TOI values of 4 and 1 differ the most for different thresholds. The first TOI and the fourth TOI value both have uncertainties when measuring the time for detecting abnormal traffic. From earlier observation the threshold value need to be higher than 3. The TOI value 4 with the threshold 4 and 5 gives a double detection time compared to its TOI value. For TOI value 1 the threshold value of 4 and 5 also has doubled detection time but the TOI is lower. This will lead to better capturing of abnormal traffic despite the same qualities. Therefore the TOI value chosen was 1 second and the threshold 5. We did not choose threshold value 4 because we thought it was too close to the minimum value of 3 and the margin was needed.

---

Figure 6 - First time abnormal traffic is detected with attackers
6.2 Analyzing entropy values

Figure 7 shows the entropy values for the specific IP addresses of the attackers and their chosen resource. The blue staples correspond to the first test done in section 5.2.1. This is when attackers are requesting only one resource. As shown, also in Figure 7, 50.1 and 51.1 are two attackers with IP addresses 50 and 51 that both have requested resource 1. The blue staples were collected in three runs of Test 1. The attackers with IP address 50 and 51 first requested resource 0 in the first run, resource 1 in the second run and resource 2 in the third run. The red staples correspond to the second test from section 5.2.2 when the attackers randomly requested the three resources.

Figure 7 – Attacker IP addresses with requested resource and their entropies

Figure 7 shows the differentiation between the entropy values calculated for the different attacker IP addresses and the resource requested. This shows a clear distinction between the two DDoS attacks. From equation (8) in section 3.1.2 we know that the entropy for repeated request application layer DDoS attack is lower than the one for recursive request attack. Figure 7 shows that CALD can distinguish between the two attacks that were tested.
The amount of requests for resource 2 during the whole duration of Test 1 can be seen in Figure 8. This is from section 5.2.1 when a repeated request application layer DDoS attack was conducted. The requests are made from both the attackers and clients. One can see a change in amount of requests being made between 73 seconds until the 194-second mark. The amount of requests from attackers and client during the attack period are between 5 and 8. Furthermore, visualization for requests of resource 0 and 1 can be found in Appendix.

In Figure 9 visualization over all the three resources requested randomly is shown. This result is from section 5.2.2 when a recursive request application layer DDoS attack was simulated. The requests are made from both the attackers and clients.
Figure 9 show how much is being requested from legitimate clients to an uprise when the attackers starts to send requests. The change in amount of requests being made can be observed between 74 seconds until the 197-second mark. This shows the impact of the requests made by the attackers in that specific time interval. The amount of requests from attackers and client during the attack period are between 5 and 8.
7 Conclusions

DDoS attacks on the application layer are still on the rise and continue to evolve. As of this there are many detection mechanisms for defending against these attacks, which are presented in this thesis. The thesis sufficiently covers the many ways of application layer DDoS attacks and the existing defense mechanisms. This gives a general information base for DDoS attacks on the application layer.

One of the main focuses of the thesis was the implementation of a relative small-scaled DDoS attack on the application layer. The results from the implementation have been satisfying to our needs and goal. This can be shown from the results described in Chapter 7. Moreover one of the goals of the thesis focused on detecting application layer DDoS attacks and the results clearly show that it has been reached. The thesis also gives a general information basis that covers for example how DDoS attacks on the application layer are constructed and existing defense mechanisms for application layer DDoS attacks.

Future work

One improvement that could have been made to the simulation was the abnormal traffic detection implementation. When receiving requests, this phase does not detect the abnormal traffic instantly. Instead it waits until the duration of TOI is over and then checks if it is abnormal traffic or not.

A suggested solution to this could be, instead of only having a fixed TOI (in our case 1 second), a moving time frame that updates the incoming traffic and saves the earlier requests made could have been implemented. Because of this, if abnormal traffic has occurred before the TOI is over, the moving time frame detects it instantly and the next phase of CALD is activated. This would give a better accuracy in finding abnormal traffic since we do not have to wait for the TOI to finish. For our implementation, the website receives requests every 10-20 seconds from legitimate clients, therefore in our case a TOI value of 1 second is still satisfying.
8 References


Appendix

This appendix provides two graphs that show the incoming requests during simulation 1, Repeated request application layer DDoS attack. Figure 10 is when resource 0 is requested and Figure 11 is when resource 1 is requested.

**Figure 10** – Visualization for request of resource 0 over the whole simulation

**Figure 11** – Visualization for request of resource 1 over the whole simulation