A novel approach for mitigating the effects of the TCP SYN flood DDoS attacks

Mitko Bogdanoski
Goce Delcev University of Štip
73 PUBLICATIONS **69** CITATIONS
SEE PROFILE

Dimitar Bogatinov
Goce Delcev University of Štip
11 PUBLICATIONS **1** CITATION
SEE PROFILE

ISSN 1 746-7233, England, UK
World Journal of Modelling and Simulation
Vol. 12 (2016) No. 3, pp. 217-230

A novel approach for mitigating the effects of the TCP SYN flood DDoS attacks

Mitko Bogdanoski[1]*, Aleksandar Toshevski[2], Dimitar Bogatinov[1], Marjan Bogdanoski[2]

[1] Military Academy "General Mihailo Apostolski", an Associate Member of Goce Delcev University
[2] European University of Republic of Macedonia

(*Received May 11 2015, Accepted June 01 2016*)

Abstract. Today's modern society greatly depends on computer systems. Security is a basic need for any computer system. This is more than acceptable if we consider that any disruption of the normal function of the computer and networks may lead to catastrophic consequences. The most frequently attacks conducting malicious activities against the networks and systems are the Distributed Denial of Service (DDoS) attacks. The paper concerns the TCP (Transmission Control Protocol) vulnerability that gives space for a type of DoS (Denial of Service) attack called TCP-SYN Flood DDoS attack which is well-known to the community for several years. It explains in more detail the TCP SYN Flood DDoS attacks and methods for preventing and mitigating the effects of these attacks. Furthermore, the paper proposes a novel method consisting of five modules which can be used for mitigation and protection against the considered TCP SYN Flood attack, as well as against other similar flooding based attacks.

Keywords: attack, bayes classifier, TCP SYN flood DDoS, IDS

1 Introduction

The weakness of the TCP protocol on SYN flood attacks was discovered back in 1994 by Bill Cheswick and Steve Bellovin. Two years later the article was published[6] which gives advises for mitigating this type of attacks. The SYN Flood attack was far more effective compared to the previously known denial-of-service (DoS) attacks.

Rather than to rely on simple brute-force tactics for spending the network resources, SYN flood attack uses fewer packets to overwhelm the victim. As a response to this attack the Internet community quickly developed a multitude of techniques to prevent or limit the effects of the attack. Some of these techniques are used on the Internet, servers and routers and some of them have become an important part of the implementation of the TCP protocol in certain operating systems. Although these newly-emerging techniques seriously deviate from the specification of TCP, none of these techniques have been standardized by the IETF (Internet Engineering Task Force)[10]. Cyberoam[28] warns that the latest version of Orbit Downloader - famous program for downloading videos from YouTube, transforms computers into SYN attack bots. It is known that after installing Orbit Downloader, it automatically starts sending large amounts of traffic to random IP and makes DDoS attack.

1.1 Description of the tcp syn flood attacks

TCP SYN flood attacks are one of the most dangerous and easiest to perform a DDoS attack (Fig. 1). It utilizes the congenital "weaknesses" of the TCP protocol, which of course was not considered as weaknesses when the protocol was designed[4, 14].

* Corresponding author. *E-mail address*: mitko.bogdanoski@ugd.edu.mk

Some of the characteristics of successful TCP SYN flood attack are the following:

- The attacker tries to create a large number of server connections;
- The SYN RECEIVED state is created when the victim receives a connection request (a packet with a set of SYN flag) and allocate memory resources to it;
- After receiving a connection request (a packet with a set of SYN flag), the server places this half-open connection in the backend queue (backlog) and returns a response to the client (package with SYN and ACK flags);
- When there is no feedback from the server, it again starts to sends SYN + ACK packet until there is a time out, and then finally removes the half-open connection from the backlog queue;
- The total process of SYN request for some operating systems takes about three minutes;
- SYN flood attack creates so many half-open connections that crowd the system and it can no longer receive new requests;
- Connections are in a SYN RECEIVED state until the backlog queue of the server gets full;
- The operating system can handle only a certain amount of half-open connections in the backlog queue that is controlled by the size of the backlog queue. For example, the default size of the backlog Debian Squeeze is 2048 bytes. When it reaches this size, the system can no longer receive connection requests;
- It is important to know that every TCP port has its own backlog queue, but only one variable of the TCP/IP stack controls the size of the backlog queue for all the gates;
- The attacker additionally spoofs the IP address of SYN packets so it can increase the efficiency of the attack.

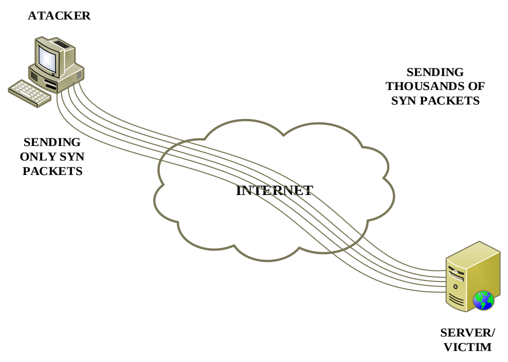


Fig. 1: TCP SYN Flood DoS attack

Consequently, for a short period of time, the system cannot complete the initialization, because the alleged address either does not exist or the host is not aware of the connection, and it does not give a reply to the SYN+ACK packets sent from the system.

There is an advanced distribute form of SYN attack, that belongs to a group of DDoS attacks (Fig. 2). The distribution of the attack makes it very dangerous for the victim mainly because of the increased volume of traffic directed to the victim's site. Tracking of the distributed attack is very hard, which is the main reason for difficulties in defending against TCP SYN DDoS attack.

There are more or less effective methods to mitigate this type of DDoS attack, and in each case, proper packet filtering is a sustainable/feasible solution. The creation of these packet filters requires a modification to the TCP/IP stack, which will help in reducing the effects of the SYN attacks and allows usage of the server by the legitimate users. It should be noted that some SYN attacks are not always trying to affect the server, but they are trying to overflow Internet bandwidth of the target.

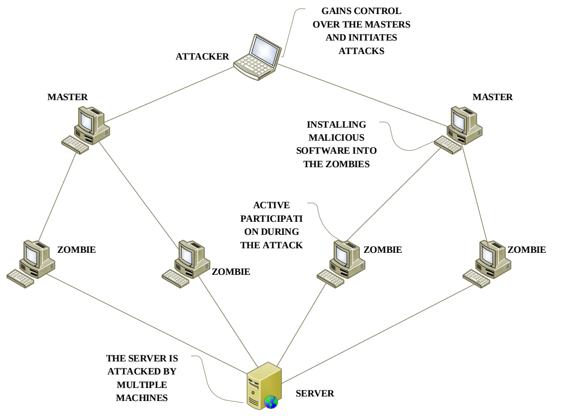

GAINS CONTROL OVER THE MASTERS AND INITIATES ATTACKS
ATTACKER
MASTER
MASTER
INSTALLING MALICIOUS SOFTWARE INTO THE ZOMBIES
ACTIVE PARTICIPATI ON DURING THE ATTACK
ZOMBIE
ZOMBIE
ZOMBIE
ZOMBIE
THE SERVER IS ATTACKED BY MULTIPLE MACHINES
SERVER

Fig. 2: Distributed TCP SYN flood attack

2 Simulation of the TCP SYN flood DDOS attack

The TCP SYN flood DoS attack was conducted against the production web server using a virtual machine with the following Linux script that spoofs a random IP address and a random source port:

```
#!/usr/bin/perl
# synspoofflood.pl

use Net::RawIP;

sub genIP(){
$range = 255;
$iA = int(rand($range));
$iB = int(rand($range));
$iC = int(rand($range));
$iD = int(rand($range));
return $iA. ”.” . $iB. ”.” . $iC . ”.” . $iD;
}

sub attack(){
($dst,$dport) = @ARGV;
$a = new Net::RawIP;
while(1) {
$src_port = rand(65534)+1;
$src = sub genIP();
$a->set({ip => {saddr => $src,daddr => $dst},tcp => {source => $src_port,dest => $port, syn => 1}});
$a->send;
}
}
if($#ARGV == 1) {
attack();
}
else {
print ”Target Port \ n”;
}
```

Due to the attack, the server and the spoofed addresses that were on the same network had suffered a huge load and damage.

The presentation of the effects of the attack are further shown using one of the most reliable simulation tool - Riverbed Modeler. The simulation scenario consists of attacker, bots (controlled by the attacker) and legitimate users that use the server for FTP and Web browsing (Fig. 3).

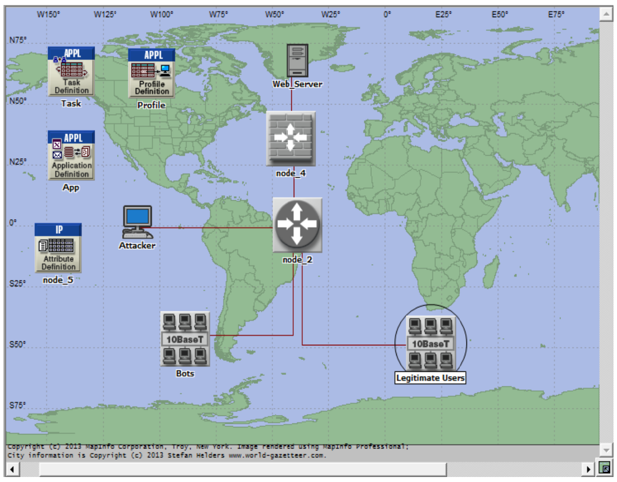


Fig. 3: Simulation scenario

Fig. 4, taken from Wireshark, shows SYN request packets sent to the web server from our public IP address. The following characteristics are considered:

- Frame,
- Ethernet frame
- IP header
- TCP header

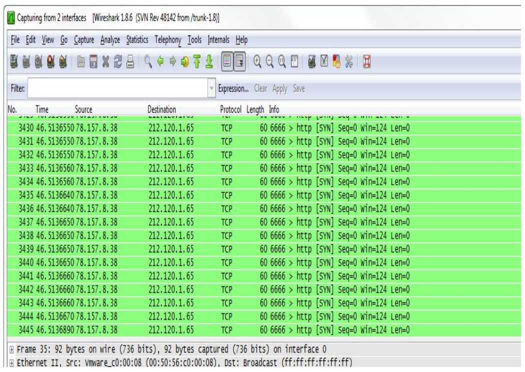
The frame carries data as: arrival time, frame number, frame length and the protocols that the frame includes in. Ethernet frame contains the source MAC address (address of the network card), a destination MAC address (address of the router) and the type of packet that the frame carries, in our example 0x0800 (IPv4). The header of the Internet Protocol contains information as: source IP address (78.157.8.38), destination IP address (212.120.1.65), checksum and other header data. TCP header gives us very important information about the source port (generated as a random number greater than well-known ports), the destination port (80), the sequence number, verification number (ACK), the length of the header, etc..

The number of active connections on the server during the two scenarios is shown in Fig. 5. The figure is a summary graph of the two scenarios where the red line shows the established connections in normal use, and the blue line shows connections composed of normal users and attackers. The period of 50 seconds is the period of the establishment of normal communications in the simulation and therefore there are not results which will show the established connections. After this initial period, the scenario which presents the network under TCP SYN Flood DDoS attacks shows exponential growth of the number of connections that in a short period of time reaches a value of a thousand connections. This value remains unchanged until the end of the simulation, which is the maximum number of web connections the server can serve.

During this period all other connections are rejected. Red line represents the connections from the legitimate users ranging up to hundreds of connections.

Fig. 6 shows canceled connections among legitimate users trying to access the server. Actually, the canceled connections occur as a consequence of the attack.

The considered DoS attacks not only block the user's access to the services, but also damages the server. Fig. 7 shows the CPU load which ranges from 20% in normal use, up to one 100% at the peak of the attack, and remains so until the end of the scenario. Such attacks can leave serious consequences to the equipment like physical damage to the processor and the memory. This effect can be transmitted on the network and other



Capturing from 2 interfaces [Wireshark 1.8.6 (SVN Rev 48142 from /trunk-1.8)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.  Time       Source       Destination    Protocol Length Info
3430 46.5136550 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3431 46.5136550 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3432 46.5136550 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3433 46.5136560 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3434 46.5136560 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3435 46.5136640 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3436 46.5136640 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3437 46.5136650 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3438 46.5136650 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3439 46.5136650 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3440 46.5136650 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3441 46.5136660 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3442 46.5136660 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3443 46.5136660 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3444 46.5136670 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
3445 46.5136890 78.157.8.38  212.120.1.65   TCP      60 6666 > http [SYN] Seq=0 Win=124 Len=0
⊞ Frame 35: 92 bytes on wire (736 bits), 92 bytes captured (736 bits) on interface 0
⊞ Ethernet II, Src: Vmware_c0:00:08 (00:50:56:c0:00:08), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
```

Fig. 4: TCP SYN packets (Wireshark)

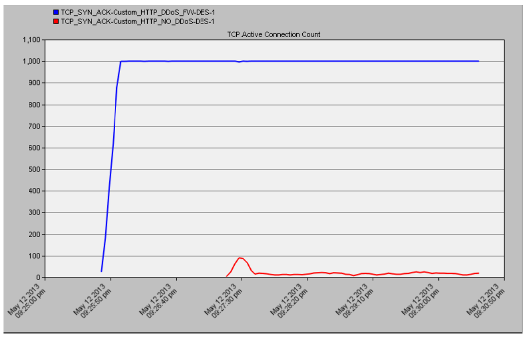


Fig. 5: Established TCP connection to the server

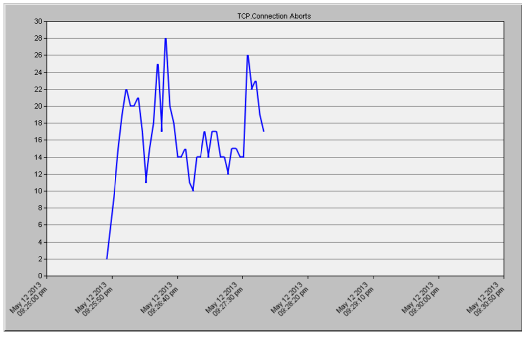


Fig. 6: Aborted connections from legit users

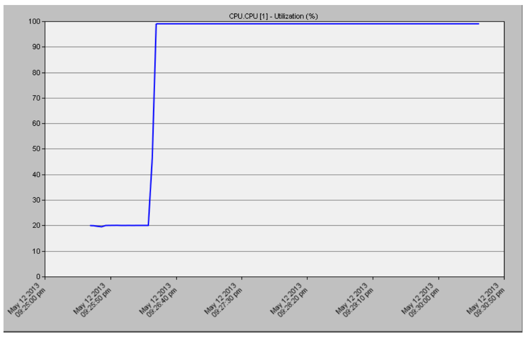


Fig. 7: CPU utilization

servers, due to the size of the traffic generated by the attacker. This traffic affects the links' bandwidth causing high and unnecessary load on the router that handles all that traffic.

3 Mechanisms for detection of the TCP SYN food DDOS attacks

Valid and early detection of the attack is crucial for secure communications and providing desired quality of service. There are many proposed methods and mechanisms which fall into many different categories based on the way they work, the methods they are use and detection complexity. The mechanisms range from simple ones that are based on manual recognition, statistical analysis, neural networks to more sophisticated approaches that are based on data mining and artificial intelligence.

3.1 Bloom-filter

The Bloom-filter is space-efficient data structure designed by Burton Howard Bloom in 1970, which can be implemented on routers with pattern recognition capabilities. This filter can use different algorithms and it can be used for packet control and malicious attempts recognition. In [14] a new type of DoS attack, LDoS (Low-rate Denial of Service) is presented. This attack was firstly noticed in 2003. LDoS is more complicated than DDoS, because it exploits security flaws in the TCP protocol. It differs from the traditional DDoS flooding attacks because it uses a form of short periodical bursts with different duration and amplitude. The proposed scheme in [31] is based on TCP SYN-SYN/ACK protocol pairs considering the packet header information. This scheme was primarily designed to be used as a defense mechanism against previously mentioned LDoS attacks. The authors propose Counting Bloom Filter (CBF) which can be used to avoid the effects of ACK retransmission. They also propose change point detection method in order to avoid the dependence of detection on sites and access patterns. So as to record the TCP session statistics for IP Time to live (IP-TTL) of SYN packets[14] adopted a Traceback-based Bloom Filter (TBF). When the attack starts, the SYN packets and statistics of the IP-TTL are matched to differentiate the attacking packets and record the IP-TTL. Experiments confirmed that the effectiveness of this method is with 98.65% accuracy of the attackers rejection.

3.2 Statistical analysis

The authors of [23] discuss the relevance of the modeling and analyzing linear time series models for characterizing network traffic the edge router network traffic and detecting low rate TCP SYN DoS attacks. In most of the analyzed time period during the DoS attacks detection, it is assumed that the underlying stochastic process is stationary and stable. However, analyzes have shown that these assumptions are not correct. Moreover, the authors studied two different transformations of series of half-open connections, differencing and averaging. They found that the averaging transformation leads the process to the region of instability, while the differencing brings back the process to stability. They also simulated TCP SYN attacks on three sets of data and studied the prediction errors for various threshold values, to show the efficiency of the model in detecting the attack. They found that for the threshold value of 5, there is a 0% probability of false negative and 11% false positive assessments.

Furthermore, the authors of [17] proposes a statistical method that is based on the mean value for detection of SYN flood attacks. The process of matching is made by comparing the differences between the mean values of total incoming traffic arrival rate and normal traffic arrival rate. The main advantage of this technique is the small processing load, because this scheme does not hold three-way handshaking states, but only count the SYN and ACK packets. This technique cannot deal with a low rate attack efficiently, since this attack follows the normal distribution.

3.3 Fuzzy logic and neural networks

Many researcher[8], [35] have adopted the Fuzzy logic and neural networks in designing and implementing intrusion detection systems for DoS attacks. Fuzzy logic provides a mathematical strength to capture the uncertainties, and fits in solving nonlinear systems[24].

The authors of [34] proposed a system that consists of two blocks (Fig. 8). The first block is the incoming packets classification block, where the header of each captured packet is checked to see if it is a TCP SYN packet; if the fragment offset value of the header is zero, then it is a TCP packet. If the SYN flag bits are one, then it is a SYN packet. The packet classification block collects TCP SYN packets for a specified time t (which in this case is set to 5 seconds) and send them to the fuzzy logic system–the second block of the proposed system.

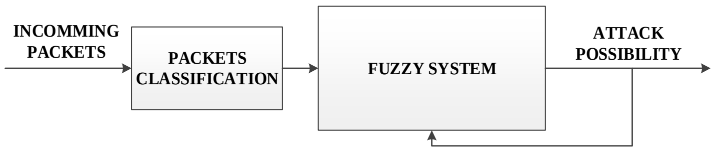


Fig. 8: A system based on fuzzy logic[34]

The detection accuracy of the proposed system was compared with Cumulative Sum (CUSUM) of five attacks, and it showed high accuracy and low number of estimated false negative attacks. This mechanism generates an earlier alarm about the attacks compared with the CUSUM algorithm, which is an ideal algorithm for identifying DoS attacks based on measurements of the mean value of the previous traffic, and after detection - comparison with the threshold value.

3.4 Data mining and machine learning

There are many other mechanisms for detecting DoS attacks that are different from the previous mentioned. Data mining techniques[33] were created and adopted to discover useful information and hidden relationships in large data repositories.

In [30], the network traffic and the status model of the network packet protocol were extracted by the FCM (Fuzzy C-Means) clustering algorithm and Apriori association. The proposed system was feasible also in LAN networks.

Proposed model in [36] is based on a simple proactive anti-DDoS framework. The framework firstly analyzes the characteristics of DDoS attacks and then selects the variables based on the findings and finally uses k-nearest neighbor (K-NN) to classify DDoS SYN attacks at each phase. The results showed that this method can successfully classify DDoS phases.

3.5 Generic mechanisms

As we already mentioned there are many more existing defense mechanisms against TCP SYN Flood attacks. Among the other existing mechanisms[25] in RFC 4987 suggests several effective mechanisms for mitigating the effects of the DoS attacks. They propose the following mechanisms:

- Filtering false IP addresses,
- Increasing the backlog queue[11],
- Decrease the SYN-RECEIVED timer (so will reduce the period for which the TCP connection may remain in a semi-open position),
- Recycle the oldest half-open connection,
- SYN cache[1],
- SYN Cookies[19, 38],
- ICMP Route Redirection[5],
- Limit the number of connections,
- TCP congestion control algorithms[3],
- Firewall and proxy[12].

4 A novel method for mitigating the effects of the TCP SYN FLOOD DDOS attack

The proposed method in this paper is practical and based on the basic tools that can be found in the operating system and Internet. It consists of: detection module, classification module, defense module, user panel with statistics module and notification module (Fig. 9).

4.1 Module 1: detection

The detection is performed with a simple command that oversees current requests to a web server when they are in SYN_RECV state. An example of such a command is the following Linux command:

```
netstat–n–p TCP|grep SYN_RECV|grep: 80|wc –l >> i.txt
```

or

```
tcpdump –i eth0 –nn 'tcp port 80' and 'tcp [13] == 2' – c100
```

Counter i -counts the current client requests for establishing a connection. If the value of i is greater than a certain threshold, for example, 80% of the size of the backlog queue, then the IP addresses obtained from the above netstat detection attack command are entered into the log table, and the number of repeats of the detected addresses is taken as an additional field. In Linux it can be done using the following command:

```
netstat-anpt|grep SYN_RECV|grep: 80|awk '{print $5}'|cut -d: -f1|uniq -c|sort -nr >> /var/log/DDoS_attack.log
```

Next step is to transfer the addresses from the log file to the Bayes classifier, where the addresses are detected whether they belong to the attackers or to the legitimate users acquiring connection problems.

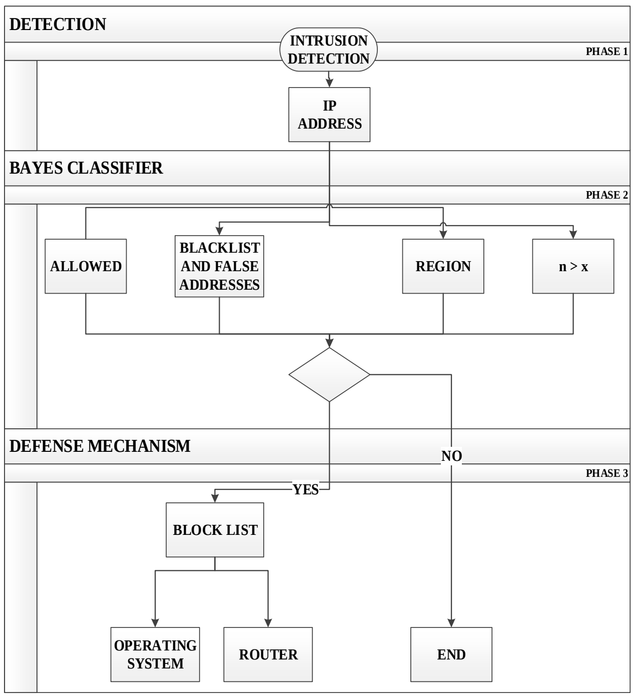


Fig. 9: Algorithm for detection and protection of TCP SYN Flood DoS attacks

4.2 Module 2: bayes classifier

In our method we chose to use ”Naive” Bayes Classifier due to its efficiency and fast classification. This module could be written in any of the programming languages such as C and can be located on the web server, for saving time. It will read the IP address of the log file, compare it with four attributes, and classify into yes or no, ie, block or not.

The attributes include:

- Table with IP addresses that have established half-open connections,
- Table permitted that contains local address (127.0.0.1, etc), which belong to our network and is predefined by the user,
- Table with known false Bots and addresses. This table can be updated daily. The web service [24] offers a database with known malicious IP addresses in different formats (for Linux, Cisco, etc..).
- Table with addresses by region. There is a high possibility that the address is from Asia, Africa, South America or other location that is not typical for a visitor to our server and belongs to the attacker or the zombies. There are several commercial Geo-location databases, and their pricing and accuracy may vary. Ip2location, MaxMind Ipligence and Tamo Soft are selling databases that can be easily integrated into web applications, and offer API & code (in ASP, PHP, .NET and Java) that can be used to download data from the database. Vendors that offer commercial databases also offer Free Lite or Community versions that provide a mapping of the IP addresses. Some examples are Ip2Country.net and Webhosting.info. Iplocation also offers free Geo-location service.
- A table with the number of connections per address, which is used to extract the fourth attribute. Table classes in which the belonged address is classified are:
- Table-Yes
- Table-No

The algorithm should be trained with a certain number of supervised learning tasks. This means training in specific instances with unknown examples that the classifier has not previously met. The entire set of input values is called Development Set and it is divided in Training Set and Evaluation Set. The evaluation set should not have the same values because there will be a false high results.

Table 1: The training algorithm

Address	Allowed	Blacklist	Region	No. of connections	Class
127.0.0.1	yes	no	local	5	no
182.38.180.21	no	yes	China	1	yes
5.39.83.94	no	no	France	15	yes

4.3 Module 3: defense mechanism

Once the classifier gives addresses that are part of the attack, the next step would be taking actions to prevent their activities. We propose the first defensive wall to be the router because good defense should be closer to the source. An appropriate solution is if it is managed the primary defense to be located on the router–this solution will allow the router to stop the malicious traffic to reach and overload the server.

4.3.1 Configuring the router

In order router to be correctly configured we use bash scripts that convert the classifier output into IP access lists directly on the router, connecting to the SSH (Secure Shell). The scripts would look like the following:

```
#!/bin/bash
#script.sh
echo “Please insert the password”
stty -echo
read password
stty echo
export ssh=‘./ssh.sh’
export telnet=‘./telnet.sh’
export error=‘./report_errors.log’
export temp=‘./tmp_rtr.log’
export cmdcisco=‘./command.txt’
export list=‘./list.txt’
export password
export command
rm -f $error
rm -f $ssh
rm -f $telnet
cat $list — while read router;
do
if [“$router” !=“”]
   then
      if[! -f $ssh]
         then
         echo ‘expect 2>&1 << EOF’>> $ssh
         echo ‘spawn ssh admin@router’ >> $ssh
         echo ‘expect {‘>> $ssh
         echo “Password:” {send “$password̊”}’ >> $ssh
         echo ‘timeout {exit}’ >> $ssh
         echo ‘ }’ >> $ssh
         echo ‘expect “#” ’ >> $ssh
         cat $cmdcisco — while read command
```

```
    do
        echp ‘send \“$command\ r\”’
        echo ‘expect “#”’
    done >> $ssh
    echo ‘send “exit\r”’ >> $ssh
    echo ‘expect “closed”’ >> $ssh
    echo ‘exit’ >> $ssh
    echo ‘EOF’ >> $ssh
    chmod + x $ssh
fi
time -p $ssh > $temp 2>&1
CODE_RED = $?
auth=‘cat $temp |grep -c “Password:”’
if [“$auth” -gt “1”]
    then
    echo “Authentication problem on $router !”
    echo “$router : wrong login/password”>> $error
    continue
fi
temps=‘grep “real” $temp | sed ‘s/real /§/’ | cut -d‘§’ -f2 | cut -d‘ ’ -f1 | cut -d‘.’ -f1’
if $temps -ge 10 -a ! “‘grep ‘closed’ $temp”’
    then
    echo “The router $router does not answer!”;
    echo “$router : connection timed out” >> $error
    continue
fi
if [“$CODE_RED” != “0”]
    then
    #Error connecting to SSH
    if [! -f $telnet]
        then
        echo ‘excpect 2>&1 << EOF’ >> $telnet
        echo ‘spawn telnet $router’ >> $telnet
        echo ‘send “admin°”’ >> $telnet
        echo ‘expect “Password:”’ >> $telnet
        echo ‘send “$password°”’ >> $telnet
        echo ‘expect “#”’ >> $telnet
        cat $cmdcisco | while read command
        do
            echo ‘send \”$command\ r\”’
            echo ‘expect “#”’
        done >> $telnet
        echo ‘send “exit\r”’ >> $telnet
        echo ‘expect “closed”’ >> $telnet
        echo ‘exit’ >> $telnet
        echo ‘EOF’>> $telnet
        chmod +x $telnet
    fi
    $telnet > $temp 2>&1
fi
```

```
        CODE_RED= $?
        auth=‘cat $temp | grep -c “Password:”’
        if [“$auth” -gt “1”]
           then
           echo ‘Authenticatoin on $router !’
           echo ‘$router : wrong login/password’ >> $error
        elif [‘grep “Hostname lookup failure” $temp’]
           then
           echo ‘The device $router does not exist’
           echo ‘$router : does not exist’ >> $error
        elif [‘grep ‘Unknown host’ $temp’]
           then
           echo ‘Entered IP address or name for $router is incorrect !”
           echo “$router : wrong spelling” >> $error
        elif [ “‘grep ‘send: spawn id exp4 not open’ $temp’”]
           then
           echo “/! ERROR in the procedure. Check the log file $router !!!”
           echo “$router : Expect script execution failed !” >> $error
           cp $temp $router.error.log
        elif [ “‘grep ’Authentication failed’ $temp”’ ]
           then
           echo “ Wrong password for $router !”
           echo “$router : wrong log-in/password” >>$error
        elif [ “‘grep ‘Connection refused’ $temp”’ ]
           then
           echo “ Remote connection to $router disabled!”
           echo “$router : vty connection disabled” >>$error
        elif [ “‘grep ’No route to host’ $temp”’ ]
           then
           echo “$router DNS Alias has invalid IP address !”
           echo “$router : No route to host” >> $error


        elif [ “‘grep ‘ProCurve’ $temp”’ ]
           then
           echo “/! Cisco command not recognized by the device. Check the log file $router !!!”
           echo “$router : Unrecognized command found” >> $error
           cp $temp $router.error.log
        elif [ “‘grep ‘Connected to ’ $temp“‘-o ”’grep ‘Connection closed by foreign host.’ $temp”’ ]
           then
           echo “$router Telnet OK !”
        elif [ “‘grep ‘Registered on the connection terminal’ $temp“‘ -o ”’grep ‘Connection to ’ $temp”’]
           then
           echo “$router SSH OK !”
        elif [ “$CODE_RED” != ”0” ]
           then
           echo “Problem connecting$router !”
           echo “$router : connection problem” >> $error
        fi
        fi
     done
rm -f $temp exit
```

The list of devices and its addresses managed by the previous script is the following:

```
# list.txt
10.1.2.1
R1
```

The list of commands to create IP access lists for the router is the following:

```
#commands.txt
conf t
ip access-list extended blockrule
10 deny 5.39.83.94 0.0.0.0 any
20 permit any any
!
interface fastethernet 0/1/0
ip access-group blockrule in
exit
```

4.3.2 Configuring the operating system

The secondary defense would be placed on the server, on its own defense mechanism used in case of internal network attacks. Next, we present the shell script that generates IPtable rules, which will be cleared on the exit of the script. This solution (script) rejects suspicious TCP connections in SYN_RECV state, which are classified for blocking. The script for protection against flooding is as follows.

```
#!/bin/bash
att_log=/var/log/DDOS_attack.log
blackilist=/tmp/blacklist.txt
trap “echo ;echo Caught EXIT signal;iptables -F;echo Iptables entries cleared” EXIT
while true;
for att_ip in $att_log
do
  while read line
    do
      if [ “$line” = “$att_ip”]
        then
      fi
      iptables -I INPUT -s $att_ip -p tcp -j REJECT –reject-with tcp-reset echo “$att_ip” >> /tmp/blacklist.txt
    done
  sleep $1
done
```

The exit signal stops the script, and cleans the chains and rules added to the iptables.

4.4 Module 4: user panel and statistics

User Panel is the part that users can log on and monitor the equipment and the statistics in real time. There is an opportunity to create new users with different privileges. The equipment monitoring can be done with the Zabbix-like open source tool that offers an API (Application Programming Interface) for access to its data as:

- Access to history and other data collected from monitoring,
- Monitoring services and their availability,
- Graphs for better data presentation.

By having tables with generated statistics:

- Addresses set on the blacklist,

- List of addresses and subnets,
- Number of requests from the same network,
- Number of normal requests per hour,
- Number of normal daily requirements,
- List of known allowed addresses.

4.5 Module 5: notification

This module will be responsible for notifying system administrators in case of an event. The intrusion detection will cause sending emails and text messages during work hours or making a phone call to the administrators after working hours. Sending the e-mail would be performed using Zabbix or Nagios via email server. Sending SMS will be carried through Zabbix or Nagios that will be associated with Nokia SMS Gateway with SIM card. Outside of work hours, to ensure that some of the administrators will notice the attack, it will be enough to implement an analog modem that will ring list of persons on duty.

5 Summary

Internet and network connectivity have great importance not only for communication, entertainment, socializing and information, but for people's entire lives. The protection of data exchanged via the global network is a major issue in modern society. Availability of some services, such as web pages or Internet access, is guaranteed by the SLA (Service-Level Agreement) for availability and because of that the companies that offer these services must prevent any type of attack. There are many examples of security breaches. Neither the companies nor organizations that engage a large number of researchers and experts to protect their assets and systems, and that invest a lot in this sector are immune to these attacks. One of many cases is the fall of the U.S. Predator drone in Nevada in 2011 due to a virus in the navigation system. Flooding attacks are fairly easy to perform and could cause catastrophic consequences on the wired and wireless networks and systems, ranged from degrading the quality of services, loss of confidentiality to their complete loss. Most of the flooding attacks are done based on some of the vulnerabilities of the TCP protocol, which leads to TCP SYN Flood DoS attacks. The paper gives some simulation results which show the effects of this attack. Moreover, it covers several existing methods and techniques against this type of DoS attack. At the end it proposes a novel method for mitigating this type of attack.

References

[1] M. Bellaïche, J.-C. Grégoire. Avoiding ddos with active management of backlog queues. **in:** *Network and System Security (NSS), 2011 5th International Conference on*, IEEE, 2011, 310–315.

[2] D. J. Bernstein. Syn cookies, 1996.

[3] M. Bogdanoski, A. Risteski. Wireless network behavior under ICMP ping flood dos attack and mitigation techniques. *International Journal of Communication Networks and Information Security*, 2011, **3**(1): 17.

[4] M. Bogdanoski, T. Shuminoski, A. Risteski. Analysis of the SYN flood DOS attack. *International Journal of Computer Network and Information Security*, 2013, **5**(8): 1.

[5] R. Braden. T/TCP–TCP extensions for transactions functional specification. 1994.

[6] C. C. Center. Cert advisory ca-1996-21 TCP SYN flooding and ip spoofing attacks, 1996.

[7] C.-L. Chen. Detecting distributed denial-of-service attack traffic by statistical test. **in:** *Communications and Networking in China, 2008. ChinaCom 2008. Third International Conference on*, IEEE, 2008, 1253–1257.

[8] C.-L. Chen. A new detection method for distributed denial-of-service attack traffic based on statistical test. *J. UCS*, 2009, **15**(2): 488–504.

[9] Cisco. Characterizing and tracing packet floods using CISCO routers, 2005.

[10] W. M. Eddy. TCP SYN flooding attacks and common mitigations. 2007.

[11] W. M. Eddy. TCP SYN flooding attacks and common mitigations. 2007.

[12] A. Esterhuizen, A. Krzesinski. Tcp congestion control comparison. 2012.

[13] H. Eychenne. iptables(8) - linux man page, 2015.

[14] B. Hang, R. Hu, W. Shi. An enhanced SYN cookie defence method for tcp ddos attack. *Journal of Networks*, 2011, **6**(8): 1206–1213.

[15] L. M. Ibrahim. Anomaly network intrusion detection system based on distributed time-delay neural network. *Journal of Engineering Science and Technology*, 2010, **5**(4): 457–471.

[16] Iplocation. Iplocation, 2015.

[17] C. James, H. A. Murthy. Time series models and its relevance to modeling TCP SYN based DOS attacks. **in:** *Next Generation Internet (NGI), 2011 7th EURO-NGI Conference on*, IEEE, 2011, 1–8.

[18] R. Kawahara, K. Ishibashi, T. Mori, N. Kamiyama, S. Harada, S. Asano. Detection accuracy of network anomalies using sampled flow statistics. **in:** *IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference*, IEEE, 2007, 1959–1964.

[19] J. Lemon. Resisting SYN fooding dos attacks with a syn cache. **in:** *USENIX BSDCon*, 2002.

[20] Y. Ling, Y. Gu, G. Wei. Detect SYN flooding attack in edge routers. *Int. J. Secur. Appl*, 2009, **3**: 31–45.

[21] Microsoft. Windows firewall with advanced security administration with windows powershell, 2015.

[22] MIT. Darpa intrusion detection evaluation, 2000.

[23] D. Nashat, X. Jiang, S. Horiguchi. Router based detection for low-rate agents of ddos attack. **in:** *2008 International Conference on High Performance Switching and Routing*, IEEE, 2008, 177–182.

[24] H.-V. Nguyen, Y. Choi. Proactive detection of ddos attacks utilizing k-nn classifier in an anti-ddos framework. *International Journal of Electrical and Electronics Engineering*, 2010, **4**(4): 247–252.

[25] H.-V. Nguyen, Y. Choi. Proactive detection of ddos attacks utilizing k-nn classifier in an anti-ddos framework. *International Journal of Electrical and Electronics Engineering*, 2010, **4**(4): 247–252.

[26] A. W. Pape. Why IPS devices and firewalls fail to stop ddos threats. 2013.

[27] Rutgers. Iplocation, 2013.

[28] SecurityFocus. Securityfocus, 2013.

[29] C. Sun, J. Fan, L. Shi, B. Liu. A novel router-based scheme to mitigate SYN flooding ddos attacks. *IEEE INFOCOM (Student Poster)*, 2007.

[30] P.-N. Tan, et al. *Introduction to data mining*. Pearson Education India, 2006.

[31] H.-r. Tang, C. Xu, X.-g. Luo, J.-q. OuYang. Traceback-based bloomfilter ips in defending SYN flooding attack. **in:** *2009 5th International Conference on Wireless Communications, Networking and Mobile Computing*, IEEE, 2009, 1–6.

[32] S. Ting, W. Ip, A. H. Tsang. Is naive bayes a good classifier for document classification? *International Journal of Software Engineering and Its Applications*, 2011, **5**(3): 37–46.

[33] T. Tuncer, Y. Tatar. Detection SYN flooding attacks using fuzzy logic. **in:** *Information Security and Assurance, 2008. ISA 2008. International Conference on*, IEEE, 2008, 321–325.

[34] L. A. Zadeh. Fuzzy sets. *Information and control*, 1965, **8**(3): 338–353.

[35] R. Zhong, G. Yue. Ddos detection system based on data mining. **in:** *Proceedings of the Second International Symposium on Networking and Network Security (ISNNS10)*, 2010, 062–065.

[36] R. Zhong, G. Yue. Ddos detection system based on data mining. **in:** *Proceedings of the Second International Symposium on Networking and Network Security (ISNNS10)*, 2010, 062–065.

[37] L. Zi, J. Yearwood, X.-W. Wu. Adaptive clustering with feature ranking for ddos attacks detection. **in:** *Network and System Security (NSS), 2010 4th International Conference on*, IEEE, 2010, 281–286.

[38] A. Zuquete. Improving the functionality of syn cookies. **in:** *Advanced Communications and Multimedia Security*, Springer, 2002, 57–77.