Security Research

Table of Contents

Preface

My services in my microservice architecture communicate with each other using public URLs. This means that if I start my application and go to the URL and provide a word in the URL. My service will return the amount of lives you should have. Right now my application is only running on my laptop, but if the application would be online up and running. People could just DOS attack my service which would take down my application. By protecting the URL, we can avoid getting a Denial-of-Service attack.

How do I protect the URLs my microservices use to communicate against DDoS attacks?

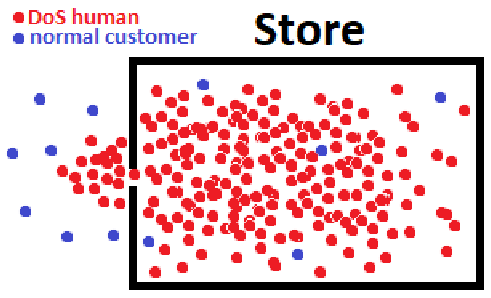
To answer this question, I have multiple sub questions. My goal is to answer all my sub questions to answer this, the main question of the research.

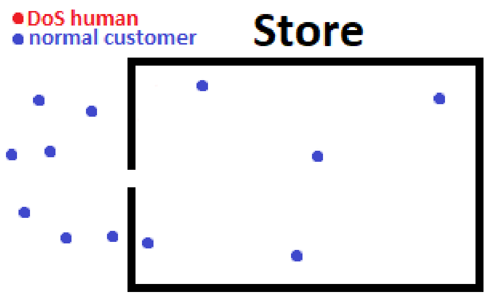
What is a DDoS attack?

A DDoS attack and a DoS attack are both different type of attacks that get confused with each other. Usually DoS attacks are also called DDoS attacks, but they have a key difference. We will go through both attacks so you understand the difference.

DoS Attack

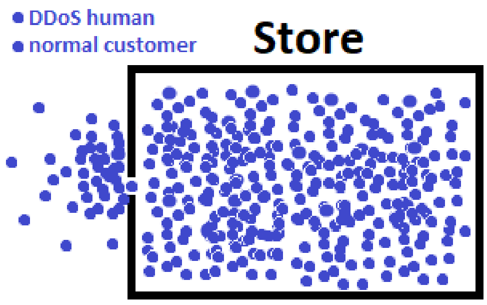
In a DoS (Denial-of-Service) attack, someone over the internet sends lots of traffic to the target. In my context this would mean someone over the internet could send lots of requests to my services. When there is more traffic on my website than my website can handle, my website won't function anymore. To visualize a DoS attack you can imagine a large group of humans that all look exactly the same, rushing in a supermarket. For a normal customer it's impossible now to go in the supermarket since it's entirely full of the same looking human being. If the manager comes in action, he knows who should get thrown out the store, since all humans with wrong intentions look exactly the same. This is also how a DoS attack can be stopped, identify who does the attack and block him. If we would visualize it in a drawing, you can just remove all red DoS humans and problem solved.





DDoS attacks

A DDoS (Distributed Denial-of-Service) attack is a bit more problematic. It's basically a DoS attack but this time the attacker controls other people's their pc's to execute the attack. If we translate it to the drawing again it would look something like the image below. As you can see, in a DDoS attack it's not possible anymore to separate real visitors from fake visitors.



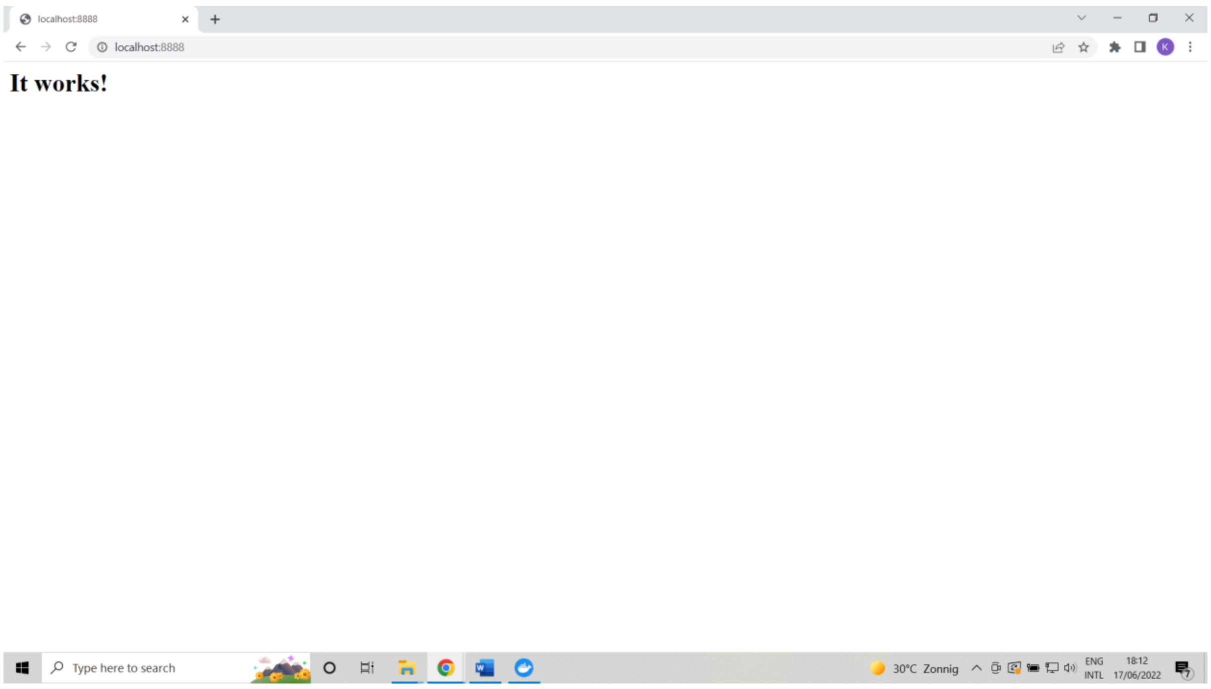
How do you prevent a DDoS attack on an URL?

To understand how to prevent yourself from a DDoS attack, I first learned how to do one.

DoS attacking a port of localhost using SlowLoris

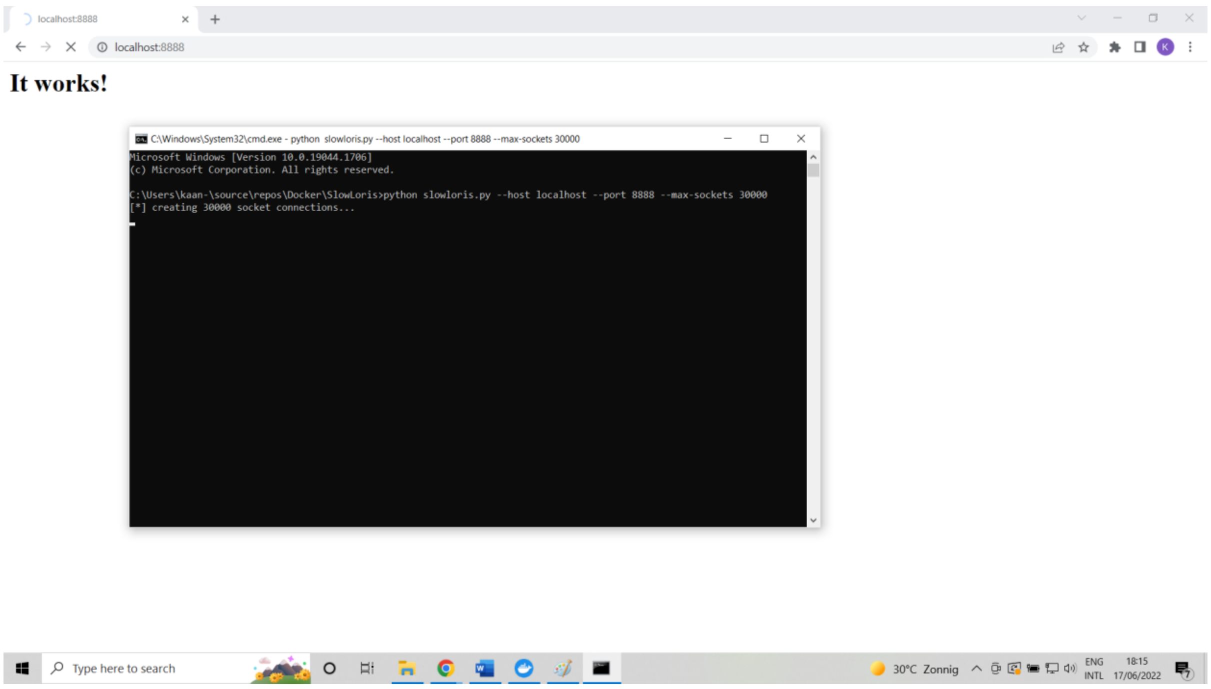
You read that correctly, I will DoS attack a port on my own laptop. I use my own laptop since I don't want to get involved into any illegal actions.

Here we have port 8888 on my localhost, page just works like expected (strange right, I know). I downloaded a python application called SlowLoris. *"The Slowloris attack, introduced by RSnake in 2009, is the antithesis of DDoS: Rather than using thousands of machines to all make HTTP requests as quickly as possible, Slowloris uses one machine to open many concurrent connections to the target computer and send data as slowly as possible. By sending new HTTP header packets just fast enough to keep each connection open, enough concurrent Slowloris connections prevents legitimate traffic from being handled by the server, as the server is too busy trying to manage the attacker's connections."* (Dorsey, 2017).



Running the following command in the SlowLoris directory opens 30000 connections with my page.

```
python slowloris.py --host localhost --port 8888 --max-sockets
30000
```


localhost:8888
localhost:8888
It works!
C:\Windows\System32\cmd.exe - python slowloris.py --host localhost --port 8888 --max-sockets 30000
Type here to search
30°C Zonnig
18:15
17/06/2022

```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\kaan-\source\repos\Docker\SlowLoris>python slowloris.py --host localhost --port 8888 --max-sockets 30000
[*] creating 30000 socket connections...
```

If I try to refresh the page, it makes me wait approximal 32 seconds, but I do reach the page in the end. But now we only attacked the port, but we didn't attack the URL.

Attacking the URL is not possible inside of SlowLoris so I'll have to find another software.

Attacking my applications' URL

Now I want to attack an API URL in my application. I got my backend running on port 80. The backend calculates your lives in the following link `http://localhost/Lives?word=help`. I wanted to use LOIC (Low Orbit Ion Cannon) to DoS my URLs, but this might also give me a virus. I installed a VM just to realise that doesn't solve the problem. So sadly I won't be able to DoS my own URLs for obvious reasons.

So how do you prevent a DDoS attack on a URL?

I looked up multiple sources that told me how to protect your URLs against DDoS attacks and the following solutions I encountered the most.

API Keys

One of the solutions is to add API keys to my microservices. An API key is a long string of characters. The purpose of the API key is to prevents randoms to call the API's. For example I talk with the Merriam-Webster dictionary API using an API key. This way they prevent everyone from just using it.

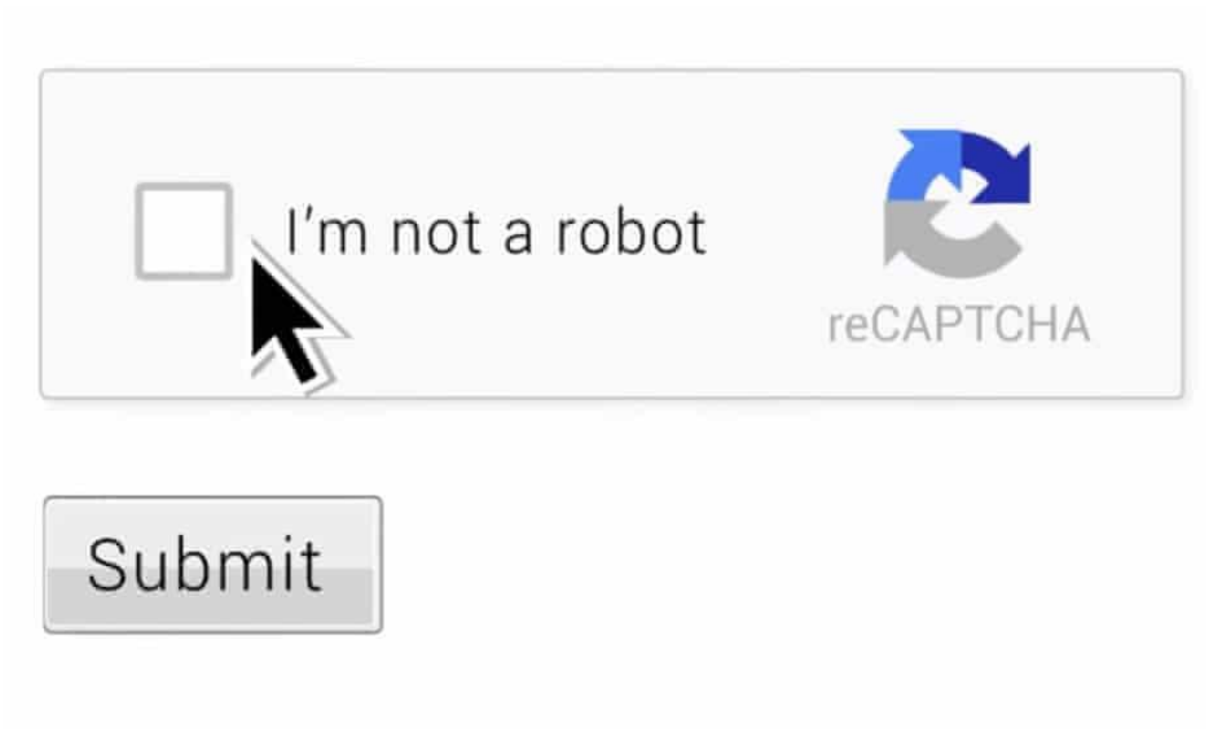
Rate Limits

Another solution is to add a rate limit to my service. By coincidence I have already done this for my API wrapper (word-service). Word-service makes use of Merriam-Websters dictionary API. Merriam-Websters API didn't include a rate limit, so I made one myself so I would never exceed the rate limit.

But I could also implement this rate limit in another way. It's possible to limit calls per IP address. This sounds like a good solution in my application, since if you just keep rate limits low and share them between everyone, the people who attacks my website can hit the limit instantly. But if I limit per IP the attacker can do for example 1 request per 5 minutes, and a normal visitor can also do 1 request per 5 minutes. This is effective against a DoS attack since a DoS attack makes use of 1 IP. But a DDoS attack would still be critical since it makes use of different IP's.

Authentication

By making visitors first login I can verify they are real. This way I filter away an entire botnet that is being controlled by an attacker. But it doesn't necessarily have to be a login. Does this look familiar? (look down) I can make visitors force to prove they aren't a robot when visiting my website.



Conclusion

To protect my API URLs there are multiple solutions I can realize. Summed up in short, API Key, Rate Limit, Authenticate. As if right now I have 50 hours to complete my entire project. So I won't actually change anything, but if I would. I would actually go for all 3.

Giving my services API keys to communicate sounds like an interesting solution to me. I have used API keys many times in multiple applications, but never have I made on for my own application. Not only is it exactly what I need, but also am I curious how you make it.

A rate limit isn't that much effort to build, I have done it before, and the more protection the better.

As in authentication, I won't build a screen that confirms you aren't a bot before you enter my website, but one of the bonus requirements I had was making a ranked system in my game. So letting people match against each other online based of their rank points. I wanted to do all this if I had extra time so this is the perfect place to mention it. To make all this I need people to login so I get two birds with one blow.

Bibliography

contributors, W. (2022, June 14). *Denial-of-service attack*. Opgehaald van Wikipedia: https://en.wikipedia.org/wiki/Denial-of-service_attack

DDoS Attacks. (sd). Opgehaald van imperva: https://www.imperva.com/learn/ddos/ddos-attacks/

Dorsey, B. (2017, June 11). *~~D~~DoS Apache Servers From a Single Machine*. Opgehaald van Medium: https://medium.com/@brannondorsey/d%CC%B6dos-apache-servers-from-a-single-machine-f23e91f5d28

OWASP API Security - Top 10 | OWASP. (sd). Opgehaald van OWASP: https://owasp.org/www-project-api-security/

Smarter Tech Decisions Using APIs. (2022, February 17). Opgehaald van Nordic APIs: https://nordicapis.com/how-to-secure-api-endpoints-9-tips-and-solutions/

What are DoS and DDoS attacks? (sd). Opgehaald van cloudfare: https://www.cloudflare.com/learning/ddos/ddos-attack-tools/how-to-ddos/