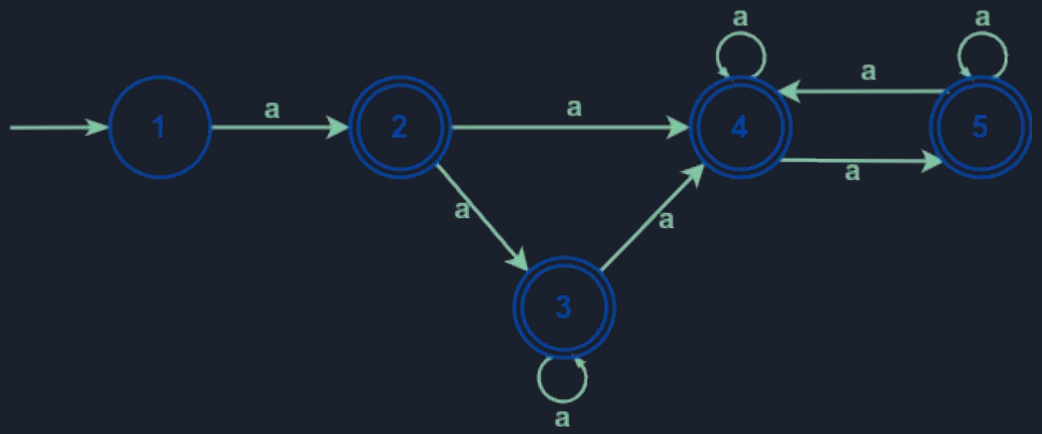
- ReDOS -
Regular Expression Denial of Service

- Maryam Saeedmehr -

What is ReDOS ?

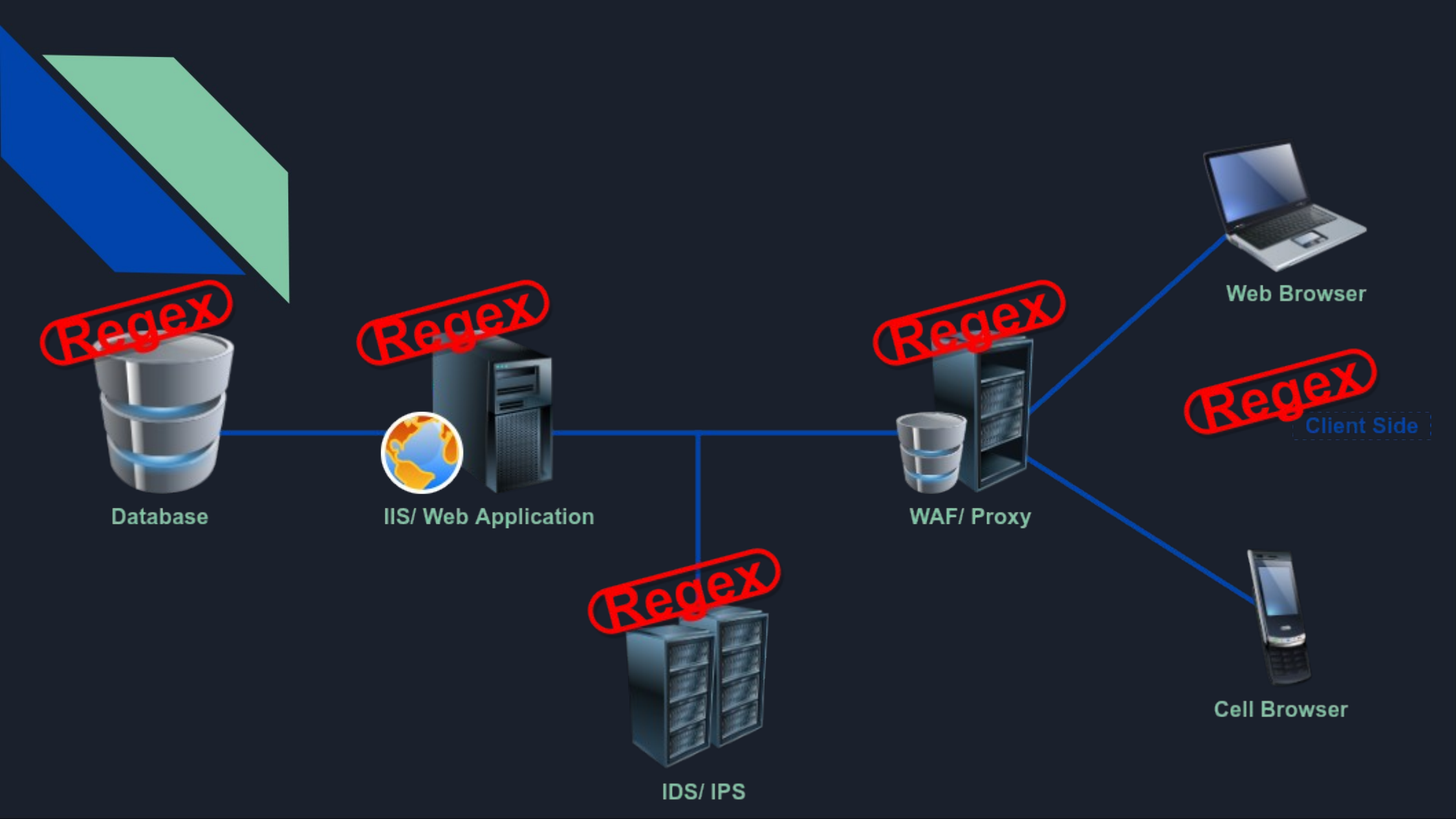
- Type of Denial of Service attack
- Most Regular Expression implementations may reach extreme situations that cause them to work very slowly
- Mainly we use Regex for validations



- For Regular Expression : ^ (a+)+ &
  - 16 possible paths for “aaaaX” as input
  - 65536 paths for “aaaaaaaaaaaaaaaaX” as input

Impacts

- Mainly the impact of this attack is related to performance of the targeted system
- It can also cause the system to get unavailable


Regex
Database
Regex
IIS/ Web Application
Regex
IDS/ IPS
Regex
WAF/ Proxy
Web Browser
Regex
Client Side
Cell Browser

Example

- Checking if the username and password are same

```
username = request.get("username")
password = request.get("password")
if password = = username
    return "Weak Password !"
```

- Input :
  - Username : ^([a-z]+.)+[A-Z][a-z]+$
  - Password : aaaaaaaaaaaaaaaaaaaaaaaaaaaaaa!

Some Points

- Nested quantifiers can be exponentially dangerous – e.g. `(a*)*`
- Quantifying a disjunction can be exponentially dangerous – e.g. `(a|a)*`
- Concatenated quantifiers cab be polynomially dangerous – e.g. `abc.*def.*`
- If you are performing a partial match, e.g. “search this free-form text for an email address”, then the regex engine adds an implicit `.*?` to the beginning of your regex.

Famous attacks caused by ReDOS

- *Cloudflare* had an outage in 2019. The root cause was a runaway regex evalutaion.
- *Stack Overflow* had an outage in 2016, also caused by a runaway regex.

DEMO

How to prevent ?

- Use safe regex engines, e.g. re2 npm package
- Discover vulnerabilities with fuzz testing, test your regex with random inputs
- Sanitize and filter user’s inputs

Thank you
for your attention

Find Demo codes in my Github