Infrastructure Third Task

For every additional element, why you are adding it?

I've added a monitoring tool, and firewalls and used HTTPS instead of HTTP.

First, I added a monitoring tool because it monitors the heart, you should check your system condition all the time if the server is working probably and not overloaded if the data is being received in time, or if there is some delay.

A firewall will block certain traffic from certain ports and IP addresses, and this will add a layer of security.

Finally using HTTPS will encrypt the connection between the client and the server which also will add security to my system and the client.

What are firewalls for?

Firewalls carefully analyze incoming traffic based on pre-established rules and filter traffic coming from unsecured or suspicious sources to prevent attacks. Firewalls guard traffic at a computer's entry point, called ports, which is where information is exchanged with external devices. For example, "Source address 172.18.1.1 is allowed to reach destination 172.18.2.1 over port 22."

Think of IP addresses as houses, and port numbers as rooms within the house. Only trusted people (source addresses) are allowed to enter the house (destination address) at all—then it's further filtered so that people within the house are only allowed to access certain rooms (destination ports), depending on if they're the owner, a child, or a guest. The owner is allowed to any room (any port), while children and guests are allowed into a certain set of rooms (specific ports).

Why is the traffic served over HTTPS?

Traffic served over HTTPS (Hypertext Transfer Protocol Secure) is encrypted and secured, providing several important benefits for both website owners and users. Here are key reasons why HTTPS is widely used and recommended:

Data Encryption:

HTTPS encrypts the data exchanged between a user's browser and the web server. This encryption ensures that the information, such as login credentials, personal details, and other sensitive data, is secure during transmission.

Encryption prevents unauthorized parties from intercepting and deciphering the data as it travels across the internet. This is especially crucial when users access websites over untrusted networks, such as public Wi-Fi.

Data Integrity:

HTTPS ensures data integrity by using cryptographic techniques to detect any tampering or modification of the transmitted data. If a third-party attempts to alter the data during transmission, the integrity checks provided by HTTPS will identify the tampering, and the connection may be terminated or flagged.

Authentication:

HTTPS uses digital certificates to authenticate the identity of the website. This means users can trust that they are connecting to the intended, legitimate website and not a malicious or fraudulent one.

When a website has an HTTPS certificate, the browser indicates this with a padlock icon, indicating a secure connection. Extended Validation (EV) certificates provide even more visual cues, displaying the organization's name in the browser's address bar.

Trust and User Confidence:

Users have come to associate HTTPS with a secure and trustworthy online environment. Websites without HTTPS may be flagged as "Not Secure" by browsers, potentially discouraging users from interacting with the site or entering sensitive information.

Many modern browsers show warnings for non-HTTPS sites, and some features, such as geolocation and certain APIs, may only work on secure connections.

SEO (Search Engine Optimization):

Search engines, including Google, prioritize secure websites in their search rankings. Websites using HTTPS may receive a slight boost in search engine rankings compared to non-secure counterparts.

This encourages website owners to adopt HTTPS for better visibility and credibility in search engine results.

Compliance:

Many regulations and standards, such as the General Data Protection Regulation (GDPR), Payment Card Industry Data Security Standard (PCI DSS), and others, require the use of HTTPS to ensure the security and privacy of user data.

What monitoring is used for?

Just as the heart monitor in a hospital that is making sure that a patient’s heart is beating and at the right beat, software monitoring will watch computer metrics, record them, and emit an alert if something is unusual or that could make the computer not work properly happens.

“You cannot fix or improve what you cannot measure” is a famous saying in the tech industry. In the age of data-ism, monitoring how our software systems are doing is an important thing.

How is the monitoring tool collecting data?

- Application monitoring: getting data about your running software and making sure it is behaving as expected.
- Server monitoring: getting data about your virtual or physical server and making sure they are not overloaded (could be CPU, memory, disk or network overload)

Explain what to do if you want to monitor your web server QPS?

Use Server Logs:

Web servers often log requests and responses. Analyzing these logs can provide valuable information about the number of queries per second. You can use tools like grep, awk, or dedicated log analyzers to extract and analyze relevant data.

Web Server Metrics:

Utilize built-in metrics provided by your web server software. For example, if you are using Apache, you can enable the mod_status module to access a web-based status page that includes various performance metrics, including requests per second.

Monitoring Tools:

Deploy dedicated monitoring tools that can capture and display real-time metrics. Tools like Prometheus, Grafana, or Nagios can help you set up dashboards to visualize QPS and other relevant metrics. These tools often support various plugins or integrations with popular web servers.

Load Balancer Metrics:

If your web server is behind a load balancer, the load balancer may provide metrics related to request rates. Check the documentation of your load balancer to understand how to access and interpret these metrics.

Application-Level Monitoring:

Instrument your web application to log or expose metrics related to the number of queries it processes. This may involve using application-level libraries or frameworks that offer built-in monitoring capabilities.

Database Monitoring:

Since web servers often interact with databases, monitor database performance metrics. Many database systems provide tools or interfaces for monitoring queries per second. For example, in MySQL, you can use tools like MySQL Workbench or command-line queries to retrieve such information.

External Services:

Consider using external monitoring services that can periodically send requests to your web server and record response times. These services often provide dashboards with QPS and other performance metrics.

Custom Scripts:

Develop custom scripts or tools tailored to your specific needs. For example, you can write a script that periodically queries the server and calculates the QPS based on the number of requests over a specific time interval.

Why terminating SSL at the load balancer level is an issue?

Because the traffic behind the load balancer is unsecure and if you're not 100 hundred percent sure about the server you're serving so don't do SSL termination and now we're vulnerable to attacks from hackers.

Why having only one MySQL server capable of accepting writes is an issue.

Single Point of Failure:

If there is only one MySQL server that can handle write operations (acting as the master in a master-slave configuration), it becomes a single point of failure. If the master server fails or experiences downtime, write operations cannot be processed, leading to potential service disruptions.

Limited Scalability:

A single write-capable server limits the scalability of your database system. As your application grows and the volume of write operations increases, a single server may struggle to handle the load efficiently. This can result in performance bottlenecks and slower response times for write-intensive operations.

Reduced Fault Tolerance:

Without additional write-capable servers, there is limited fault tolerance. In the event of a hardware failure, software issue, or any other problem affecting the master server, the entire write functionality is affected until the issue is resolved.

High Availability Concerns:

High availability is challenging to achieve with only one server capable of accepting writes. To ensure continuous availability and minimize downtime, it's common to have multiple servers distributed across different locations or data centers. This allows for failover mechanisms, where another server can take overwrite operations if the primary server fails.

Concurrency Issues:

A single write-capable server may face challenges handling concurrent write requests efficiently. As the number of concurrent write operations increases, there is a risk of contention and slower response times. Distributing write operations across multiple servers can help distribute the load and improve concurrency.

Scalability Challenges in Read-Heavy Workloads:

In scenarios where there is a significant imbalance between read and write operations, having only one write-capable server may limit the scalability of read operations. Read-heavy workloads can benefit from distributing read operations across multiple servers to handle increased demand.

Why having servers with all the same components (database, web server and application server) might be a problem?

- Security. Your web server lives in a DMZ, accessible to the public internet and taking untrusted input from anonymous users. If your web server gets compromised, and you've followed the least privileged rules in connecting to your DB, the maximum exposure is what your app can do through the database API. If you have a business tier in between, you have one more step between your attacker and your data. If, on the other hand, your database is on the same server, the attacker now has root access to your data and server.
- Scalability. Keeping your web server stateless allows you to scale your web servers horizontally pretty much effortlessly. It is very difficult to horizontally scale a database server.
- Performance. 2 boxes = 2 times the CPU, 2 times the RAM, and 2 times the spindles for disk access.

Name: Amr Mohamed Mahdi Alnas
Cohort: 17