Task 2

Imagine a server with the following specs:

- **4 times Intel(R) Xeon(R) CPU E7-4830 v4 @ 2.00GHz**
- **64GB of ram**
- **2 tb HDD disk space**
- **2 x 10Gbit/s nics**

The server is used for SSL offloading and proxies around 25000 requests per second. Please let us know which metrics are interesting to monitor in that specific case and how would you do that? What are the challenges of monitoring this?

From the description, It looks like SSL offloading decrypts and verifies data on the load balancer. The load balancer has mainly two components, Frontend, and Backend. Monitoring load balancer meaning monitoring frontend and backend metrics along with load balancer health metrics.

1. **Frontend Metrics**
    a. HTTP Request per second (Throughput)
    b. HTTP server error
    c. Number of bytes received
    d. Number of bytes sent
    e. Error rate
2. **Backend Metrics**
    a. Average backend response time and latency
    b. Average time spent by a request in the queue
    c. Uptime
    d. Network bandwidth
3. **Health Metrics**
    a. Number of healthy backends
    b. CPU usage
    c. RAM (Memory) usage
    d. Disk space and usage

When monitoring these metrics, it's important to understand which components are being impacted. Those metrics mentioned above will help to look first. For example, how to know if a service is taking too long to process a request? The first metric to check for response time is Latency.

We can push all these metrics to metrics collection monitoring system/tools like Grafana, influxDB, Nagios, Zabbix, or SaaS tools like Datadog. These monitoring tools are also capable of sending alert functionalities in several ways including PagerDuty, Slack, Email, and other webhook compatible services.

Challenges of monitoring server metrics

There are some challenges that we face when monitoring this,

1. **Server health:** Continuous monitoring of server health and performance manually.
2. **Proactive monitoring:** Proactively tracking the performance metrics of a system to prevent downtime.
3. **Service monitoring:** Constant monitoring of the client's request and direct it to a particular host to provide the desired service.
4. **Fault/incident management:** Automating corrective actions to be performed after a server crash.