**Нагрузочное тестирование**

**Нагрузочное тестирование** – это автоматизированный процесс, имитирующий одновременную работу определенного количества пользователей на каком-либо общем ресурсе.

**Нагрузка** – это совокупное количество попыток выполнить операции на

общем ресурсе.

(Создается пользовательской активностью/нагрузочными скриптами)

**Как указать ab сделать нагрузку в 10000 запросов, 500 из которых будут направлены одновременно?**

**Команда:** ab -n 10000 -c 500 http://192.168.1.100/test.php

* Если сервер мощный, то запросы обработаются без проблем и будет виден результат тестирования, в противном случае сервер может отвечать очень медленно либо вовсе не отвечать.

**Тестирование ab, httperf и Siege другие web-сервера**

1. **Nginx** – популярный легковесный веб-сервер
2. **Microsoft IIS** – сервер от Microsoft, поддерживающий ASP.NET
3. **Tomcat** – сервер для запуска Java-приложений
4. **Caddy** – современный веб-сервер с автоматической настройкой HTTPS
5. **Lighttpd** – легковесный веб-сервер, оптимизированный для высокой производительности.

**Влияет ли использование скриптовых языков программирования на производительность web-сервера**

**Нагрузка на выполнение:**

Скриптовые языки требуют интерпретации, что добавляет накладные расходы по сравнению с компилируемыми языками.

* + каждый запрос может обрабатываться дольше, если скрипты не оптимизированы

**Потребление ресурсов:**

PHP-скрипты могут быть ресурсоёмкими, особенно если они включают сложные запросы к БД или неэффективный код.

* + плохо написанные скрипты могут привести к увеличению использования ЦП и памяти, что замедлит время отклика сервера.

**Балансировщик нагрузки**

Используется для распределения входящего сетевого трафика между несколькими серверами.

**Методы балансировки нагрузки в Nginx:**

1. **Round Robin** – запросы распределяются равномерно между всеми доступными серверами по кругу.
2. **Least Connections** – направляет трафик на сервер с наименьшим количеством активных соединений.
3. **IP Hash** – использует хеширование IP-адреса клиента для определения какой сервер будет обрабатывать запрос.
4. **Generic Hashing** – похоже на хеширование по IP, но позволяет использовать любую часть запроса в качестве входных данных для функции хеширования.
5. **Sticky Sessions** – поддерживает постоянство сессий, направляя все запросы от конкретного клиента к одному и тому же серверу.