Інформаційні технології

УДК 004.852

Волошин Владислав Сергійович

бакалавр комп'ютерних наук

Національного технічного університету України

«Київський політехнічний інститут»

Волошин Владислав Сергеевич

бакалавр компьютерных наук

Национального технического университета Украины

«Киевский политехнический институт»

Voloshyn V.

Bachelor of computer science

The National Technical University of Ukraine

«Kyiv Polytechnic Institute»

КОНФІГУРАЦІЯ ВЕБ-СЕРВЕРІВ В УМОВАХ ВИСОКОГО НАВАНТАЖЕННЯ

КОНФИГУРАЦИЯ ВЕБ-СЕРВЕРОВ В УСЛОВИЯХ ВЫСОКОЙ НАГРУЗКИ

WEB SERVER'S CONFIGURATION IN CASE OF HIGH LOAD

Анотація: Були розглянуті методи підвищення надійності та продуктивності веб-сервера.

Ключові слова: веб-сервер, високонавантажена система, кешування, балансування навантаження.

Аннотация: Были рассмотрены методы повышения надежности и производительности веб-сервера.

Ключевые слова: веб-сервер, высоконагруженная система, конфигурация, кеширование, балансировка нагрузки.

Abstract: investigated high availability and performance methods for web server.

Keywords: web server, high load system, caching, load balancing.

Сучасний стан відкритих веб-систем вимагає надійності та достатньої продуктивності для того, аби задовольнити великий попит на них. При зростанні популярності сервісу з'являються проблеми з навантаженням, що неодмінно призводить до погіршення роботи системи. Система стає високонавантаженою в той самий момент, коли при налаштуванні зі стандартною конфігурацією перестає справлятись з навантаженням. Так як більшість сучасних веб-систем користуються в своїй роботі готовими рішеннями в обслуговуванні http-запитів, а саме веб-серверів, їхня конфігурація потребує ретельного аналізу.

Підвищити продуктивність веб-сервера можливо шляхом використання 2 методів, таких як: конфігурація внутрішніх параметрів веб-сервера та впровадження змін в архітектурі веб-системи.

Конфігурація веб-сервера має бути максимально простою та легкою у підтримці і модернізації. Завдяки правильному налаштуванню параметрів сервера можливо підвищити швидкість сайту та віддачі контенту. Досягти цього можливо використовуючи кеш, модернізуючи характеристики веб-сервера як кількість запитів до сервера, об'єм службових даних в запиті, розмір даних передачі. Крім того, необхідно звертати увагу і на клієнтську частину, а саме оптимізацію відображення змісту сайту веб-браузером.

- **Gzip**

Всі сучасні браузери підтримують роботу зі стисканням Gzip. Веб-сервер стискає зміст відповіді перед відправленням його клієнту, а браузер розпаковує його в момент отримання. Таким чином можливо зекономити до 70% розміру файлу. Для клієнтів це буде означати більш високу швидкість роботи сайту.

- **HTTP/2**

Всі сучасні веб-сервери як і веб-браузери підтримують протокол HTTP/2. Нова специфікація набагато швидша та продуктивніша за HTTP 1.1. На відміну від тестового HTTP 1.1, HTTP/2 – бінарний. Тому протокол більш ефективний при парсингу, більш компактний при передачі та схильний до меншої кількості помилок. В HTTP/2 використовується мультиплексування, яке дозволяє використовувати браузеру одне TCP з'єднання для всіх запитів. Разом із мультиплексуванням з'явилася і пріоритезація трафіку. Запитам можна назначити пріоритет на основі важливості та залежності.

- **Google PageSpeed**

Google Pagespeed – це набір інструментів, розроблений компанією Google, який використовується для аналізу швидкості сайту та її оптимізації. Модуль pagespeed для веб-серверів, таких як Apache та Nginx, дозволяє автоматизувати багато задач пов'язаних з оптимізацією продуктивності. Таким чином він дозволяє покращити характеристики веб-сервера без складної конфігурації. Модуль може стискати статичний контент сайту та навіть оптимізувати картинки.

- **Мініфікація файлів**

З іншої сторони існують спеціальні програмні продукти, "project builders", що опрацьовують текстові файли, прибираючи пусті символи. Вони створюють мініфіковані копії файлів. При створенні великого

проекту, дані копії здатні оптимізувати роботу сайту. Прикладами таких програм є Gulp, Grunt, написані на Javascript.

В процесі стискання всі коментарі до коду, переноси рядків, зайві символи табуляції та порожні символи видаляються, тобто ті символи, які не впливають на працездатність сайту. Такий підхід дозволяє зекономити 10-20% від оригінального розміру файлу.

З точки зору архітектурних рішень можливі наступні варіанти:

- **Кешування**

Кешування – це один із способів оптимізації web-систем. Дані, що лежать далеко від користувача потребують значного часу на їх передачу. В будь-якій програмі зустрічаються повільні операції, результати яких можна зберегти на деякий час. Це дозволить виконувати менше таких операцій, а більшості користувачів видавати заздалегідь збережені дані. Тобто основна мета кешування для веб-серверів – зменшення часу на отримання контенту, аби користувач якомога швидше отримав свої дані.

За даними компанії Yahoo, до 80% трафіку сайту можна «зрізати» за рахунок грамотного кешування. Тому кешування, як відносно просту технологію, впроваджують усюди: в браузерах, в проксі-серверах та ін.

- **Черги задач**

Черги задач дозволяють виконувати важкі операції асинхронно, не сповільнюючи систему. Цей принцип дозволить збільшити швидкість роботи певної ділянки системи, обслуговувати більшу кількість користувачів, використовувати різні мови розробки в одній програмі. Черги повідомлень є сполучною ланкою між різними процесами в системі і забезпечують надійний і здатний до масштабування інтерфейс взаємодії з іншими підключеними системами.

- **Балансування навантаження**

Балансування навантаження є ключовою складовою високо доступних систем, що часто використовується для покращення

продуктивності та надійності веб-сайтів, програм та інших сервісів, розподіляючи навантаження серед багатьох серверів.

Користувач відсилає запит на сервер, на якому встановлене спеціальне програмне забезпечення, що відповідає за балансування навантаження. Далі сервер пересилає запит користувача на backend сервер, який вже віддає контент користувачу.

Як зрозуміло з призначення балансування, сервери, що слугують балансувальниками, повинні передавати запити лише на «здорові» backend сервери, тобто ті, які здатні опрацьовувати трафік користувача. Для відстежування стану backend серверів регулярно з певним інтервалом проводяться перевірки.

- **CDN**

CDN (Content Delivery Network) – це спеціальна технологія, яка дозволяє відвідувачу отримувати контент сайту з різних географічних місць.

При великих відстанях сервера від користувача має місце затримка при передачі даних. Це призводить до того, що швидкість сайту буде відрізнятися для різних місцезнаходжень. Для рішення цієї проблеми і існує CDN. Принцип його роботи досить тривіальний. Для того, аби користувач отримав дані швидше, вміст сайту переноситься територіально ближче до нього. Тобто сервера добавляються в потрібних місцях і туди копіюється сайт.

Тема конфігурації веб-серверів є досить цікавою і представляє собою широке поле для подальших досліджень. Можна зробити висновок, що для збільшення показників продуктивності веб-серверів в реальних навантажених проектах слід переходити від простого налаштування параметрів до модернізації архітектурної складової системи. Потрібно зосереджувати увагу на розширенні веб-інфраструктури за рахунок розподілу на взаємопов'язані частини, що складатимуть єдину систему,

тим самим звільняючись від проблеми масштабування та нестачі ресурсів. Таким чином досягаються кращі результати у надійності системи та її працеспроможності.

Література:

1. Мультиплексування в HTTP/2. – Режим доступу: https://blog.planethoster.net/wp-content/uploads/2015/11/HTTP2-explication.jpg
2. Http Archive: Interesting Stats – Режим доступу: http://httparchive.org/interesting.php
3. Оптимізація і масштабування Web програм. – Режим доступу: https://ruhighload.com/
4. Habrahabr: «Стратегия кеширования в приложении»– Режим доступу: https://habrahabr.ru/post/168725/
5. DigitalOcean: What is Load Balancing? – Режим доступу: https://www.digitalocean.com/community/tutorials/what-is-load-balancing