КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 2 (14), 2021
ISSN 2663 - 4023
DOI 10.28925/2663-4023.2021.14.186191
УДК 004.942, 004.715

Гречанінов Віктор Федорович
кандидат технічних наук,
завідувач науково-дослідного відділу ІПММС НАН України, Київ, Україна
ORCID ID: 0000-0001-6268-3204
vgrechaninov@gmail.com

АНАЛІЗ І ПРОЕКТУВАННЯ РОЗПОДІЛЕНИХ СИСТЕМ НА ОСНОВІ КЛАСТЕРНИХ ТЕХНОЛОГІЙ

Анотація. У даній роботі виконано огляд деяких сучасних рішень, що забезпечують аналіз і проектування розподілених систем на основі кластерних технологій. Ці рішення ґрунтуються на використанні теореми CAP, яка стверджує, що для розподіленої комп'ютерної системи неможливо одночасно забезпечити виконання більше двох із трьох властивостей: узгодженості даних, доступності та стійкості до розділення. Стосовно до даної теорії наводиться формалізація визначення властивостей для інформаційних і розподілених систем. Розглядається використання механізмів масштабованості в двох напрямках: вертикального масштабування (при якому відбувається нарощування обчислювальної потужності одного сервера) та горизонтального масштабування (при якому виконується нарощування великої кількості серверів в межах одного кластеру). Також, описується етап аналізу розподілених систем, де основна увага приділяється горизонтальному масштабуванню. У роботі наводиться приклад створення програмного оточення для проведення експериментальних досліджень в області оцінки роботи розподілених систем. В основі побудованої розподіленої системи знаходиться кластер, який об'єднує корпоративну мережу з трьома локальними підмережами та до 40 серверів. Кожен сервер може мати кілька віртуальних машин з різними операційними системами. Наведено приклади програмного коду, який створює поди на яких встановлюється веб-сервер nginx, горизонтальне масштабування створеного поду та перевіряє поточний стану створених об'єктів і запуск подів. Проведена робота надає можливість отримання і аналізу експериментальних даних на наступних етапах дослідження кластерних систем за допомогою навантажувальних впливів на веб-сервери і бази даних.

Ключові слова: розподілені системи, кластерні технології, теорема CAP, аналіз і проектування розподілених систем, горизонтальне і вертикальне масштабування.

ВСТУП

В даний час ще не повністю вирішена проблема підвищення навантаження на обчислювальні й інформаційні ресурси розподілених систем, які викликані обробкою великих обсягів даних. Існує безліч прикладів, коли Web-сервери і сервіси мають проблеми управління та аналізу даних великого обсягу [1-3]. При цьому, в великих соціальних інтернет-проектах велика кількість користувачів одночасно формують запити, які вимагають від систем управління даними не тільки великої пропускної здатності і низьких затримок, але і масштабованості, надійності і певних гарантій узгодженості даних.

Незважаючи на велику популярність від використання сучасних інформаційних технологій, певний накопичений досвід застосування і універсальність застосування кластерних технологій, традиційні сервісні додатки дуже часто не можуть задовольнити сучасні вимоги обчислювальних систем [4-6]. Це призводить до появи великої кількості спеціалізованих розподілених систем, здатних більш якісно справлятися з виникаючими завданнями.

Мета роботи

Метою даної роботи є розгляд етапів аналізу і проектування для вивчення розподілених систем на основі кластерних технологій, формалізація властивостей теореми CAP стосовно до розподілених систем, створення програмного оточення для проведення експериментальних досліджень в області отримання кількісних і якісних оцінок роботи розподілених систем.

Постановка задачі

Постановка задачі спирається на теорему щодо властивостей інформаційних систем і має наступне формулювання.

Нехай є деяка розподілена система, до складу якої входять вузли (сервери) і поди. Необхідно виконати аналіз на основі теореми CAP [7] стосовно до розподілених систем, проектування програмного оточення і установки відповідного програмного забезпечення для проведення подальших експериментальних досліджень з метою отримання компромісних рішень за властивостями доступності (Availability) і стійкості до розділення (Partition Tolerance) виходячи з теореми CAP. За основу отримання таких рішень взяти властивість узгодженості (Consistency) як не змінюваний параметр.

Рішення проблем аналізу розподілених систем на основі кластерних технологій

При побудові сучасних розподілених систем на основі кластерних технологій в першу чергу розглядається використання механізмів масштабованості в двох напрямках. Перший напрямок використовує вертикальне масштабування при якому відбувається нарощування обчислювальної потужності одного сервера. Другий напрямок використовує горизонтальне масштабування при якому виконується нарощування великої кількості серверів в межах одного кластеру. Особливістю вертикальної масштабованості є присутність деякого порогу (або межі) і висока вартість витрат на реалізацію поставлених вимог в області підвищення продуктивності серверу. Тому найбільш прийнятним варіантом вирішення більшості завдань в розподілених системах є горизонтальне масштабування. Даний механізм масштабування передбачає збільшення кількості серверів або віртуальних машин при зростаючих навантаженнях під час обробки великих обсягів запитів.

Однак, горизонтальне масштабування системи управління даними є досить складним завданням, так як традиційні веб-сервера і SQL-орієнтовані бази даних спочатку створювалися для роботи на одній машині. При цьому, такі додатки погано адаптовані до роботи в кластерах і хмарних структурах. Це призводить до розробки нових концепцій в роботі розподілених систем і створення підходів, які вирішують проблеми горизонтального масштабування.

Незважаючи на те, що різні підходи спрямовані на вирішення конкретних проблем масштабованості, кожне рішення має ті чи інші переваги і недоліки. Тому, вибір того чи іншого підходу залежить від розв'язуваної задачі. При цьому, у багатьох випадках успішно визначити підхід при вирішення конкретного завдання допомагає теорема CAP або теорема Брюера [7]. Дана теорема використовується для обґрунтування компромісів при проектуванні розподілених систем і ґрунтується на тому, що побудова інформаційних систем або розподілених структур не може виконуватися при одночасному виконанні наступних трьох властивостей [7-10]: узгодженості (Consistency) [7,10], доступності (Availability) [7,11] і стійкості до розділення (Partition Tolerance) [7,12,13].

У той же час, теорема CAP досі є мало формалізованою для багатьох областей сучасних інформаційних технологій [4,6]. Проте є публікації, де теорема CAP досить добре формалізується і доводяться непогані результати її роботи для деяких окремих випадків, які з успіхом використовуються для інформаційних систем [8]. Однак, як зазначає Деніел Абаді (Daniel Abadi) в [6], формулювання теореми CAP не враховує такої важливої властивості системи, як величина затримки при відповіді на запити користувачів.

Рішення проблем проектування розподілених систем на основі кластерних технологій

Визначимо на основі постановки задачі і теореми CAP або Брюера [7] згідно з якою маємо такі визначення щодо розподілених систем.

Визначення якості узгодженості (Consistency) - всі вузли в кожен момент часу мають узгоджені дані, тобто всі користувачі в будь-який момент часу отримують на однакові запити однакові дані. Стосовно до розподілених систем дане визначення ґрунтується на тому, що від системи будь-яка отримана відповідь є самою останньою версією даних (тобто будуть гарантовано однаковими), незалежно від вузла, до якого прийшов запит. Якщо протягом цього часу надійде будь-якої запит, то він повинен дочекатися завершення синхронізації даних між вузлами розподіленої системи, тобто повинна виконуватися деяка транзакція. При цьому, можлива наступна формалізація: "якщо операція B почалася після успішного завершення операції A, то тоді операція B повинна бачити стан системи в момент завершення A чи ж в новому стані".

Визначення якості доступності (Availability) - при виході з ладу будь-яких серверів, вузли, що залишилися, повинні продовжувати функціонувати і виконувати обробку запитів користувачів. Стосовно до розподілених систем ця властивість ґрунтується на тому, що кожен вузол (якщо він не вийшов з ладу) завжди повинен відповідати на запити. При цьому, не має значення, чи повертає він останню копію даних чи ні. Стосовно до постановки задачі, дане визначення ґрунтується на тому, що від системи повинна бути отримана відповідь, яка за часом не перевищує певне граничне значення. При цьому можлива наступна формалізація.

Нехай існують відповіді на безліч запитів до системи, які представлені деяким часом на їх реалізацію. Тоді, прийнята границя часу на основі експертних оцінок або експериментальних даних буде визначати доступність до ресурсів розподіленої системи. Наявність запитів до ресурсів розподіленої системи, що перевищують цю границю часу, буде визначати таку систему як недоступну.

Визначення якості стійкості до розділення (Partition Tolerance) - при збоях в мережі система поділяється на окремі групи серверів, які не зв'язані між собою. Такі групи вузлів повинні продовжувати функціонувати. Стосовно до розподілених систем і постановки завдання, дане визначення ґрунтується на тому, що в групі кластеру має перебувати максимально можлива кількість вузлів, що забезпечують доступність до розподілених ресурсів.

Створення програмного оточення

Програмне оточення створено для проведення експериментальних досліджень в області оцінки роботи розподілених систем. В основі такої розподіленої системи знаходиться кластер, який об'єднує корпоративну мережу з трьома локальними підмережами з кількістю серверів до 40 одиниць. Кожен сервер може мати кілька віртуальних машин з різними операційними системами. Як сервер, так і віртуальні машини здатні створювати поди які також беруть участь в горизонтальному

масштабуванні. Наприклад, програмний код, який створює поди на яких встановлюється веб-сервер nginx виглядає наступним чином:

```
root@ubu02:/home/khoshaba# microk8s kubectl get pods
No resources found in default namespace.
root@ubu02:/home/khoshaba# microk8s kubectl create deployment test-nginx --image=nginx
deployment.apps/test-nginx created
root@ubu02:/home/khoshaba# microk8s kubectl get pods
NAME                          READY   STATUS              RESTARTS   AGE
test-nginx-59ffd87f5-bdlt7   0/1     ContainerCreating   0          10s
```

Горизонтальне масштабування створеного поду виконується наступним чином:

```
root@ubu02:/home/khoshaba# microk8s kubectl scale deployment test-nginx --replicas=3
deployment.apps/test-nginx scaled
```

Перевірка поточного стану створених об'єктів і запуск подів виконується наступним чином:

```
root@ubu02:/home/khoshaba# microk8s kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
test-nginx-59ffd87f5-bdlt7   1/1     Running   0          2m33s
test-nginx-59ffd87f5-mqffm   1/1     Running   0          69s
test-nginx-59ffd87f5-r47lz   1/1     Running   0          69s
root@ubu02:/home/khoshaba#
root@ubu02:/home/khoshaba# microk8s kubectl expose deployment test-nginx --type="NodePort" --port 80
service/test-nginx exposed
root@ubu02:/home/khoshaba# microk8s kubectl get services test-nginx
NAME         TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
test-nginx   NodePort   10.152.183.89    <none>        80:30690/TCP   11s
```

Далі, при необхідності можливе використання дашборду та інших систем управління кластерними ресурсами з метою отримання і аналізу експериментальних даних на наступних етапах дослідження розподілених систем.

Перспектива подальших досліджень

Дана робота є початковим етапом в дослідженні одного з напрямків використання кластерних систем. Подальша робота буде полягати в проведенні навантажувальних впливів на вибрані об'єкти дослідження (веб-сервери, бази даних) з використанням теореми CAP. При цьому, передбачається отримати кількісні та якісні оцінки компромісних рішень щодо властивостей доступності (Availability) і стійкості до розділення (Partition tolerance) для фіксованого параметра властивості узгодженості (Consistency).

ВИСНОВКИ

В даній роботі:

- виконано огляд деяких сучасних рішень, що забезпечують аналіз і проектування розподілених систем на основі кластерних технологій. Ці рішення ґрунтуються на використанні теореми CAP.
- на основі теорії CAP наводиться формалізація визначень властивостей для інформаційних і розподілених систем;
- описується етап аналізу розподілених систем, де основна увага приділяється горизонтальному масштабуванню;
- наводиться приклад створення програмного оточення з метою отримання і аналізу

експериментальних даних на наступних етапах дослідження кластерних систем за допомогою навантажувальних впливів на веб-сервери і бази даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1 Cooper, B. et al. (2008). PNUTS: Yahoo!'s Hosted Data Serving Platform. *VLDB Endowment (VLDB 08)*, 1277-1288.
2 Sobel, J. (2008). *Scaling Out.* Facebook Engineering Notes. www.facebook.com/note.php?note_id=23844338919&id=9445547199.
3 Fox, A., Gribble, S. D., Chawathe, Y., Brewer, E. A., Gauthier, P. (1997). Cluster-based scalable network services. *ACM SIGOPS Operating Systems Review*, *31*(5), 78–91. https://doi.org/10.1145/269005.266662.
4 Brewer, E. (2012). CAP twelve years later: How the "rules" have changed. *Computer*, *45*(2), 23–29. https://doi.org/10.1109/mc.2012.37.
5 Brewer, E. A. (2001). Lessons from giant-scale services. *IEEE Internet Computing*, *5*(4), 46–55. https://doi.org/10.1109/4236.939450.
6 Abadi, D. (2010). Problems with CAP, and Yahoo's Little Known NoSQL System. *DBMS Musings, blog.* http: //dbmsmusings. blogspot.com/2010/04/problems-with-cap-and-yahoos-little.html.
7 Brewer, E. A. (2000). Towards robust distributed systems. У *the nineteenth annual ACM symposium*. ACM Press. https://doi.org/10.1145/343477.343502
8 Mahajan, P.. Alvisi, L., Dahlin, M. (2011). *Consistency, Availability, and Convergence, tech. report UTCS TR-11-22.* Univ. of Texas at Austin.
9 Shapiro, M. (2011). Convergent and Commutative Replicated Data Types. *Bulletin of the EATCS*, 104, 67-88.
10 Gilbert, S., Lynch, N. (2002). Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News*, *33*(2), 51–59. https://doi.org/10.1145/564585.564601.
11 DeCandia, G. (2007). Dynamo: Amazon's Highly Available Key-Value Store. У *21st ACM SIGOPS Symp. Operating Systems Principles (SOSP 07) ACM* (p. 205-220).
12 Hale, C. (2010). You Can't Sacrifice Partition Tolerance. http:// codahale.com/you-cant-sacrifice-partition-tolerance
13 Du, B., Brewer, E. A. (2008). DTWiki: A Disconnection and Intermittency Tolerant Wiki. У 17th Int' l Conf. World Wide Web [Online]. ACM (p. 945-952).

КІБЕРБЕЗПЕКА: освіта, наука, техніка
CYBERSECURITY: EDUCATION, SCIENCE, TECHNIQUE
№ 2 (14), 2021
ISSN 2663 - 4023

Victor Grechaninov
Ph.D. in technology,
Head of the Research Department of IPMMS of the NAN of Ukraine, Kyiv, Ukraine
ORCID ID: 0000-0001-6268-3204
vgrechaninov@gmail.com

ANALYSIS AND DESIGN OF DISTRIBUTED SYSTEMS BASED ON CLUSTER TECHNOLOGIES

Abstract. This paper reviews some modern solutions that provide analysis and design of distributed systems based on cluster technologies. These solutions are based on the use of the CAP theorem, which states that for a distributed computer system it is impossible to simultaneously satisfy more than two of the three properties: consistency, availability and partition tolerance. In relation to this theory, a formalization of the definition of parameters for information and distributed systems is given. The use of scalability mechanisms in two directions is considered: vertical scaling (in which the computing power of one server is increased) and horizontal scaling (in which a large number of servers are built within one cluster). Also, the stage of the analysis of distributed systems is described, where the focus is on horizontal scaling. The paper gives an example of creating a software environment for experimental research in the field of evaluation of distributed systems. The built distributed system is based on a cluster that combines a corporate network with 3 local subnets and up to 40 servers. Each server can have several virtual machines with different operating systems. Examples of program code are given that create pods on which the nginx web server is installed, horizontal scaling of the created pod, and checks the current state of the created objects and triggering events. This work provides an opportunity to obtain and analyze experimental data at subsequent stages of the study of cluster systems using the load effects on web servers and databases.

Keywords: distributed systems, cluster technologies, CAP theorem, analysis and design of distributed systems, horizontal and vertical scaling.

REFERENCES

1 Cooper, B. et al. (2008). PNUTS: Yahoo!'s Hosted Data Serving Platform. *VLDB Endowment (VLDB 08),* 1277-1288.
2 Sobel, J. (2008). *Scaling Out.* Facebook Engineering Notes. www.facebook.com/note.php?note_id=23844338919&id=9445547199.
3 Fox, A., Gribble, S. D., Chawathe, Y., Brewer, E. A., Gauthier, P. (1997). Cluster-based scalable network services. *ACM SIGOPS Operating Systems Review, 31*(5), 78–91. https://doi.org/10.1145/269005.266662.
4 Brewer, E. (2012). CAP twelve years later: How the "rules" have changed. *Computer, 45*(2), 23–29. https://doi.org/10.1109/mc.2012.37.
5 Brewer, E. A. (2001). Lessons from giant-scale services. *IEEE Internet Computing, 5*(4), 46–55. https://doi.org/10.1109/4236.939450.
6 Abadi, D. (2010). Problems with CAP, and Yahoo's Little Known NoSQL System. *DBMS Musings, blog.* http: //dbmsmusings. blogspot.com/2010/04/problems-with-cap-and-yahoos-little.html.
7 Brewer, E. A. (2000). Towards robust distributed systems. In *the nineteenth annual ACM symposium*. ACM Press. https://doi.org/10.1145/343477.343502
8 Mahajan, P.. Alvisi, L., Dahlin, M. (2011). *Consistency, Availability, and Convergence, tech. report UTCS TR-11-22.* Univ. of Texas at Austin.
9 Shapiro, M. (2011). Convergent and Commutative Replicated Data Types. *Bulletin of the EATCS*, 104, 67-88.
10 Gilbert, S., Lynch, N. (2002). Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News, 33*(2), 51–59. https://doi.org/10.1145/564585.564601.
11 DeCandia, G. (2007). Dynamo: Amazon's Highly Available Key-Value Store. In *21st ACM SIGOPS Symp. Operating Systems Principles (SOSP 07) ACM* (p. 205-220).
12 Hale, C. (2010). You Can't Sacrifice Partition Tolerance. http:// codahale.com/you-cant-sacrifice-partition-tolerance
13 Du, B., Brewer, E. A. (2008). DTWiki: A Disconnection and Intermittency Tolerant Wiki. In 17th Int' l Conf. World Wide Web [Online]. ACM (p. 945-952)