УДК 621.395.7

О.С. ГОРОДЕЦЬКА, Д.В. МИХАЛЕВСЬКИЙ, С.С. БІЛОШКУРСЬКИЙ
Вінницький національний технічний університет, Вінниця

ДОСЛІДЖЕННЯ СЕРВЕРНОЇ ЧАСТИНИ МЕРЕЖІ ДЛЯ ПІДТРИМКИ ОН-ЛАЙН ГРИ

Запропоновано структуру серверної частини мережі для підтримки он-лайн гри з використанням Couchbase серверу, бонд-інтерфейсів та двонаправленої "master" – "master" реплікації баз даних.

Ключові слова: серверна частина, он-лайн гра, Couchbase сервер, двонаправлена "master"–"master" реплікація.

O.S. HORODECKA, D.V. MIHALEVSKY, S.S. BILOSHKURSKY
Vinnytsia National Technical University, Vinnytsia

RESEARCH THE SERVER PART OF NETWORK TO SUPPORT ONLINE GAME

It was proposed a network architecture, which is based on implementation of NoSQL cluster, which significantly unloads the cluster with its database, and substantially increases the speed of the game and player's updating, due to using the Couchbase server - all the data what often and nonlinearly change during game sessions are stored in-memory.

We proposes the structure of the server part of the network for support online game that is based on using the bond interfaces for increasing the capacity of each server and to improve the system scalability and high availability of services.

To ensure high availability and performance database cluster is proposed to use bidirectional "master" - "master" replication. The two servers are identical data, since each server will act as a master server and as a slave server. That is, the first server in addition that will receive data from users will write log files, which will take second server information, and in turn will read data from log files of the second server. Thus, both servers are always identical data.

To provide high availability, productivity and efficiency of the database cluster it was proposed to use bidirectional "master" - "master" replication. On these two servers will be identical data each of server will act as a master server and as a slave server. That is, the first server in addition will receive data from users will write log files, which will take second server information, and in its turn will read data from log files of the second server. Thus, the both servers will always contain identical data.

Keywords: the server part of network, online game, Couchbase server, bidirectional "master" – "master" replication.

Втуп

Ігри є невід'ємною складовою сучасного світу. Згідно з останніми дослідженнями користувачі віддають перевагу он-лайн - іграм, оскільки зменшується шанс втратити здобуті цінності в грі, а також є можливість змагатись з іншими гравцями в режимі реального часу. Он-лайн ігри поділяються на декілька видів: браузерні, клієнтські та казуальні ігри.

Браузерні ігри являють собою категорію он-лайн ігор, в яких web-браузер виступає або в ролі операційної оболонки для ігор, дозволяючи грати в гру без інсталювання на комп'ютері додаткового ПЗ, або служить контейнером для додаткової віртуальної машини, яка безпосередньо виконує код гри (Java, Flash, Shockwave та аналогічні) [1]. В клієнтських іграх клієнт використовується для відображення графіки, ігрової фізики, а всі розрахунки стосовно кількості ігрових цінностей, положення персонажу на карті, взаємодії персонажу відбуваються на сервері. Казуальні ігри досить прості за структурою, найчастіше розраховані на одного користувача та є короткими: зазвичай гра починається і закінчується в рамках одного сеансу знаходження в Інтернеті. До казуальних ігор відносяться різні головоломки, віртуальні казино тощо. Саме під такий вид гри в роботі буде розроблятись мережа.

Постановка задачі

Аналіз літературних джерел свідчить, що на сьогоднішній день основними проблемами даних мереж є пропускна здатність мережевих інтерфейсів на всіх сервісах, а також проблема високої доступності кластеру баз даних. При одночасному підключені великої кількості користувачів, генерується суттєвий мережевий трафік, який впливає на роботу усієї системи. Також, при роботі системи весь час іде звернення до бази даних, запис нових даних, зчитування, оновлення, вибірка, що дуже завантажує кластер баз даних.

Таким чином, дана робота присвячена дослідженню серверної частини мережі для підтримки он-лайн гри, створенню структури такої мережі, яка б дозволила розвантажити кластер з базою даних, а також суттєво підвищить швидкість.

Основна частина

В роботі запронована загальна структура серверної частини, яка містить балансувач навантаження, що використовується для балансування запитів з мережі інтернет, ігрові сервера, бази даних та Couchbase-сервер, що підтримує технологію Memcached (рис. 1).

Memcached - це програмне забезпечення, яке реалізує сервіс кешування даних в оперативній пам'яті на основі кеш-таблиць. Принцип роботи полягає в тому, що звертання до бази даних SQL відбувається лише декілька разів, а не при кожній зміні параметрів користувача. Під час встановлення з'єднання ігровий сервер вичитує дані з ігрової SQL бази та записує ці дані в Memcached кластер, з яким надалі і працює [2]. Швидкість оброки запитів в оперативній пам'яті на порядок вища, ніж при роботі з дисковими масивами. При закритті сесії користувача ігровий сервер вичитує дані з Memcached кластеру і записує до ігрової SQL бази.

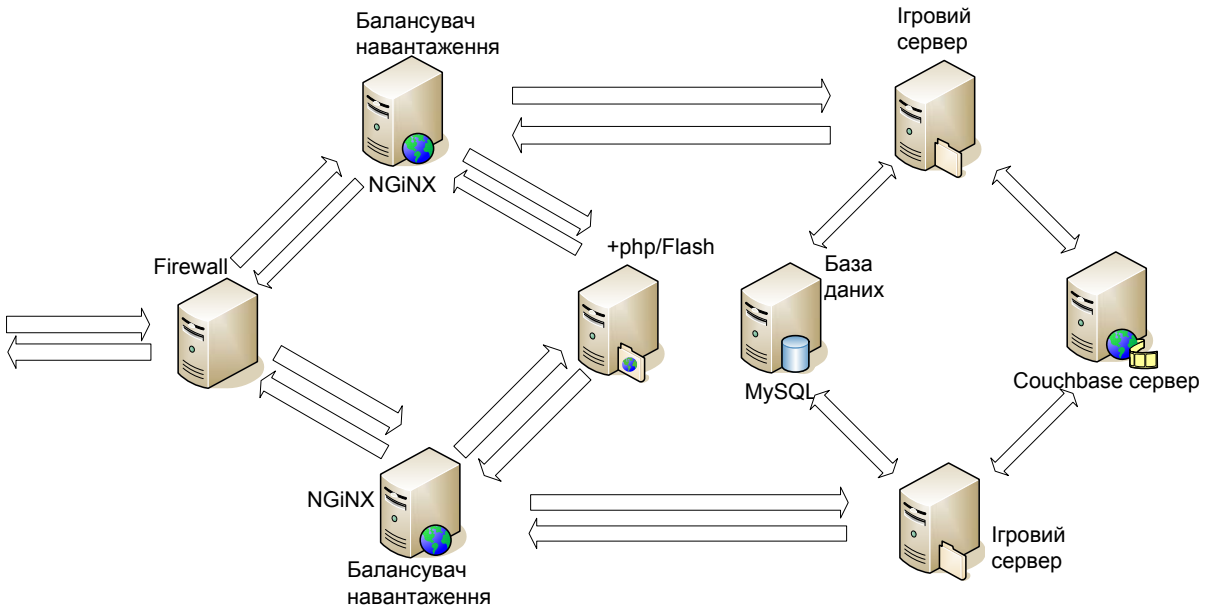

Рис. 1. Загальна структура серверної частини

Мережа на рис. 1 закрита мережевим екраном firewall від доступу ззовні. На вході стоїть балансувач навантаження load balancer, який може бути як апаратним так і програмним. Для здешевлення вартості мережі використовується програмний балансувач навантаження на основі веб-серверу Nginx, який вміє балансувати запити за технологією Round Robin, проксувати запити на інші сервіси та ін. Для налаштування маршрутів, виконання певного security policy компанії та зручного масштабування мережі в майбутньому буде використовуватись 8 портовий маршрутизатор. Для з'єднання серверів та розбиття мережі на VLAN можливо використовувати 24 портові комутатори з підтримкою LACP.

На вхід мережі приходить запит від користувача. Балансувач навантаження за технологією Round Robin проксує запит на один з серверів статік-кластера. Статік-кластер віддає користувачу статичну інформацію – картинки, аудіо, відео файли, Flash об'єкти. В той самий час, обробивши запит і отримавши SNID, Nginx проксує даний запис до ігрового серверу.

Ігровий сервер − Tomcat сервер − встановлює established session з клієнтом. Після чого формує запит з SNID і надсилає його до бази даних (MySQL), отримуючи дані про користувача: його рівень, кількість ігрової валюти, кількість бонусів, рівень проплат. Роль Couchbase серверу полягає у наступному: коли ігровий сервер отримав усі дані про користувача він починає з ними працювати. Кожну секунду, ігровий баланс, рівень, та багато інших параметрів користувача змінюється і це все не доцільно кожну секунду записувати і зчитувати з бази даних, оскільки, якщо грає лише 10 чоловік, кількість дискових операцій і трафік між ігровим сервером та базою даних не є занадто великим, але якщо присутні 100 000 одночасних активних користувачів, як показала практика, сервер бази даних спочатку не справляється з навантаження по дискових операціях і після накопичення великої черги – трапляється kernel panic.

Couchbase сервер – це NoSQL база даних, яка працює як в однонодовому режимі, так і в режимі кластера. В ній реалізована реплікація між нодами кластера, що забезпечує 100% збереження даних. Спілкування ігрового серверу та Couchbase серверу відбувається за допомогою json [3]. Всі дані зберігаються в ОЗП, що не викликає навантаження на диск. Таким чином досягається висока швидкість обміну та вибірки даних.

Ігровий сервер, вичитавши дані з MySQL, створює json об'єкт, який відправляє в необхідний бакет (так називається база даних в Couchbase) і вже з ним працює. У випадку, коли користувач закінчує гру, деякий час на ігровому сервері зберігається його сесія. Коли даний час вичерпаний, ігровий сервер забирає дані з Couchbase, обробляє їх та відправляє до MySQL, записуючи в необхідну базу даних та необхідну таблицю. Таким чином база даних використовується лише для зберігання даних. Дискові операції відбуваються лише у випадку, коли користувач логіниться та покидає гру. Також є можливість налаштувати синхронізацію, щоб наприклад кожні 5 хвилин дані з Couchbase серверу оновлювали записи в базі даних MySQL.

Дослідження даної узагальненої структури показали, що найбільш «вузькими» місцями є завантаження бази даних, оскільки без неї не зможе функціонувати гра, та пропускна здатність мережевих інтерфейсів. Для розв'язання цих питань пропонується організуват HA кластер, двонаправлену "master" − "master" реплікацію та бонд-інтерфейс.

HA Cluster (High-Availability Linux Cluster) – кластер серверів високої доступності. Принцип роботи полягає у наступному. На два сервери Node1 та Node2 встановлюється спеціальне програмне забезпечення – heartbeat, серверам присвоюється унікальна IP адреса, що випадково кидає користувачів між двома серверами. У випадку, коли один з серверів відказує, ця IP адреса автоматично присвоюється робочому

серверу [4].

Таким самим чином створюється бонд – інтерфейси на NoSQL кластері (рис. 2).

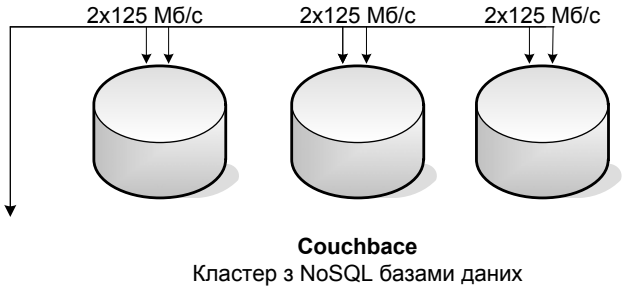

Рис. 2. Кластер з NoSQL базами даних

Запропонована структура серверної частини мережі для підтримки он-лайн гри наведена на рис.3.

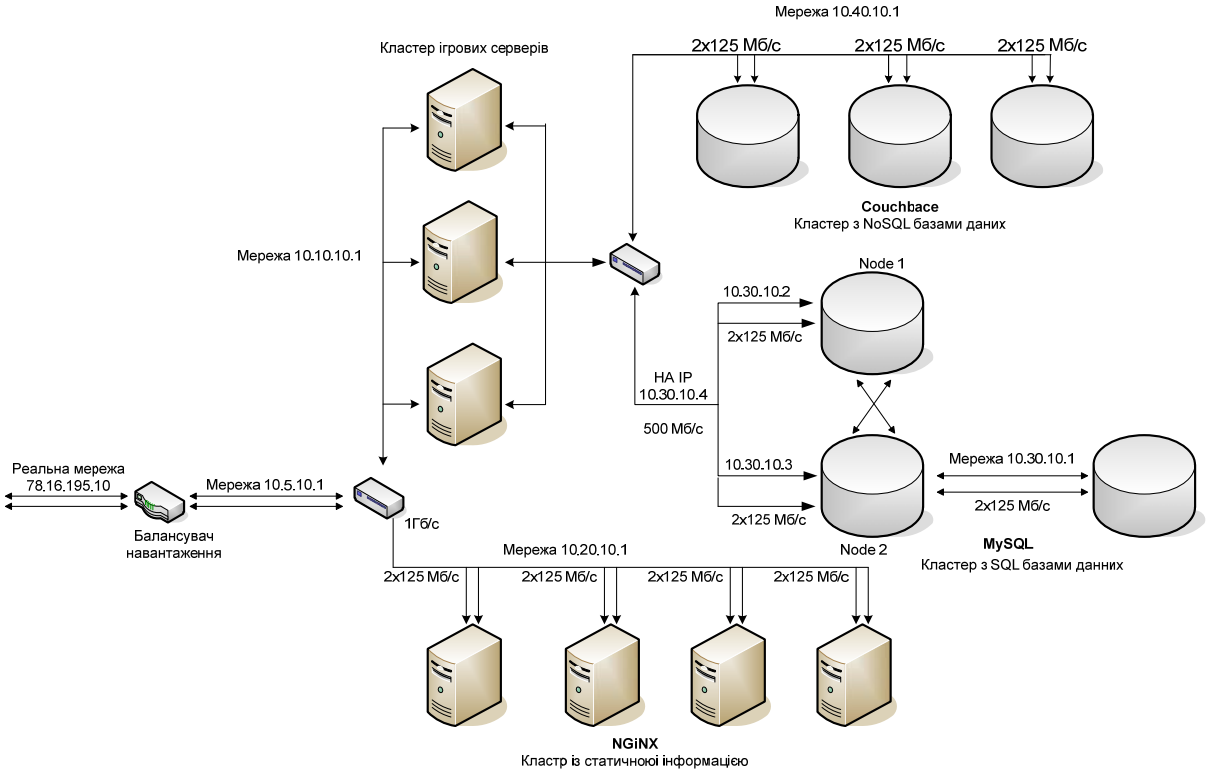

Рис. 3. Структура серверної частини мережі для підтримки он-лайн гри

Структура серверної частини мережі складається з таких основних компонентів. Балансувач навантаження здійснює балансування запитів з мережі інтернет та на основі хедерів розкидає запити на різні сервера. Маршрутизатори створюють таблиці маршрутизації та роутинг пакетів в мережі. Кластер зі статичною інформацією − кластер серверів, який відповідає за віддачу статичної інформації користувачу: картинок, флеш-об'єктів тощо. Дана група серверів має низькі характеристики: невеликий об'єм ОЗП, слабкий процесор. Дані сервера не мають великого навантаження, оскільки використовується легкий веб-сервер Nginx. В кластері ігрових серверів розташовується серверна частина гри. В якості ігрового сервера використовується Tomcat сервер для обробки java. Даним серверам необхідна велика розрахункова потужність, оскільки пам'ять, яка необхідна для роботи Tomcat, залежить від кількості користувачів на сервері. NoSQL кластер з реалізованим бонд-інтерфейсом використовується для збереження тимчасової інформації під час роботи гри. Основною відмінністю Couchbase серверів є те, що всі дані зберігаються в ОЗП, що не викликає навантаження на диск, та забезпечується висока швидкість доступу та обробки даних в пам'яті. Кластер SQL баз даних використовується для збереження інформації користувача. Для функціонування даного кластеру необхідні потужні сервера, оскільки зберігається велика кількість інформації, яка з кожною секундою змінюється, отже, необхідна енергонезалежна, високошвидкісна, та високо доступна пам'ять. Для цих цілей використовуються або дорогі SSD накопичувачі, або рейд-масиви з HDD дисків. Також для швидкої обробки всіх запитів необхідні потужні процесори та велика кількість ОЗП для збереження кешу та черг.

Робота мережі полягає у наступному. Користувач завантажує гру з соціальної мережі, тим самими

формує GET запит до ігрового кластеру наступними приблизними параметрами: get loading swf. Даний запит приходить на балансувач навантаження [4].

Балансувач навантаження, отримавши запит, перенаправляє його на один з серверів з ігрового кластеру, який у свою чергу відкриває сокет. Одночасно балансувач навантаження перенаправляє запит на кластер зі статичною інформацією і в користувача починається завантаження статичної інформації, тобто зовнішнього вигляду гри. В основному, користувач отримує лише один swf файл, який вже ініціалізує завантаження всього іншого контенту, який необхідно конкретному користувачу.

Отримавши запит від користувача, ігровий сервер надсилає соціальній мережі запит на отримання даних користувача, переважно, у відповідь очікується лише унікальний ідентифікатор користувача у соціальній мережі. Отримавши у відповідь унікальний ідентифікатор, ігровий сервер формує select запит до бази даних, з неї він отримує дані про користувача, такі як його рівень в грі, ігровий баланс, кількість різних ітемів. Отримавши дані з бази даних, ігровий сервер відкриває користувачу доступ до гри. Оскільки під час гри динамічно змінюється ігровий баланс в обидві сторони, кількість ітемів, кожен раз записувати це в базу даних немає сенсу, оскільки вона відкаже через високе навантаження на процесор, велику кількість дискових операцій. Для цих цілей використовується Couchbase кластер. Ігровий сервер під час гри користувача динамічно, при зміні будь яких параметрів, генерує json об'єкти з даними і відсилає їх до Couchbase кластеру. Протягом всього сеансу користувача ці дані змінюються і синхронізуються з даними в Couchbase кластері, тим самим знімається навантаження з бази даних.

Couchbase сервер реалізує підтримку двонаправленої "master" − "master" реплікації. На двох серверах будуть ідентичні дані, оскільки кожен з серверів буде виступати як в ролі мастер-сервера, так і в ролі слейв-сервера. Тобто, перший сервер окрім того, що буде приймати дані від користувачів, буде писати лог-файли, з яких буде брати інформацію другий сервер, і в свою чергу буде зчитувати дані з лог-файлів другого сервера. Таким чином, на обох серверах завжди будуть ідентичні дані.

Коли користувач закінчує гру, ще деякий час на сервері зберігається його сесія. Коли цей час закінчується, ігровий сервер розриває сесію і з Couchbase серверу отримує кінцеві дані про користувача, після цього формує запит до бази даних з оновленням даних.

Висновки

При використанні Couchbase серверу звертання до бази даних SQL відбувається лише декілька разів, а не при кожній зміні параметрів користувача. Під час встановлення з'єднання ігровий сервер вичитує дані з ігрової SQL бази та записує ці дані в Couchbase кластер, з якими надалі і працює. Швидкість оброки запитів в оперативній пам'яті на порядок вища, ніж при роботі з дисковими масивами. При закритті сесії користувача, ігровий сервер вичитує дані з Couchbase кластеру і записує до ігрової SQL бази. Отже, запропоноване використання Couchbase кластеру дозволило розвантажити кластер з базою даних, а також суттєво підвищить швидкість роботи гри і оновлення даних гравця.

Крім того, запронована структура дозволила вирішити дві основні проблеми: невисоку пропускна здатність мережевих інтерфейсів та велику завантаженість кластеру баз даних. Перша проблема вирішується об'єднанням двох або більше інтерфейсів в віртуальний інтерфейс – бонд. Таким чином, немає необхідності виділення нової IP адреси та переконфігурування гри. Два Ethernet інтерфейси об'єднуються в один віртуальний. Пропускна здатність збільшується вдвічі, і також збільшується відмовостійкість, оскільки при відмові одного з інтерфейсів, IP адреса автоматично присвоюється реальному інтерфейсу, і таким чином сервер не вилітає з робочої мережі. Друга проблема вирішується шляхом введення двунаправленої "master" − "master" реплікації баз даних та використання бонд – інтерфейсів, які дають можливість будувати надійні мережі з великим ростом навантаження.

Література

1. Дэвид В. Брандмауэры Cisco Secure PIX / В. Дэвид, М. Чепмен, Энди Фокс – М.: Вильямс, 2003. – 384 с.

2. Павлов П. Планирование архитектуры на основе Couchbase Server [електронний ресурс] / Павлов П. // Режим доступу до журн.: http://webperformance.ru/2013/04/29/couchbase-server

3. Ільніцький А.Ю. Основи захисту інформації від несанкціонованого доступу: Методичні рекомендації / А.Ю. Ільніцький, В.В. Шорошев, І.Л. Близнюк – К.: НАВСУ, 2001. – 160 с.

4. Голубєв В.О. Програмно-технічні засоби захисту інформації від комп'ютерних злочинів / Під ред. О.П. Снігерьова. – Запоріжжя, 2003. – 144 с.

References

1. David W. Firewalls Cisco Secure PIX / W. David, M. Chapman, Andy Fox − M. Williams, 2003. – 384 p.

2. Pavlov P.Planning architecture based on Couchbase server [E resource] / P. Pavlov // Mode to access journal.: http://webperformance.ru/2013/04/29/couchbase-server

3. Ilnitskiy A. Fundamentals of protecting information from unauthorized access: Methodical recommendations/ A. Ilnitskigo, V. Shoroshev, I. Bliznyuk − K.: NAVSU, 2001. - 160 p.

4. Golubev V. Software and technical data protection against computer crimes / Ed. O. Sniherova. - Zaporozhye, 2003. − 144 p.

Рецензія/Peer review : 11.10.2014 р. Надрукована/Printed :24.10.2014 р.