Розробка інформаційної інтелектуальної системи для управління серверами в хмарі

ст. гр. 1КСУА-15мн
Григорцевич Ю.О.

Метою дослідження є розробка методу та системи управління серверами в «хмарі», що поліпшить середній коефіцієнт використання IT-ресурсів.

Для досягнення поставленої мети розв’язано наступні задачі:

- аналіз існуючих підходів та методів управління серверами в «хмарі»;
- визначити основні переваги та недоліки використання хмарних технологій;
- обґрунтування підходу до розробки інтелектуальної інформаційної системи управління серверами.

Об’єкт дослідження. Процес генерації, передавання, оброблення, зберігання та утилізації логів.

Предмет дослідження. Методи та засоби управління серверами в «хмарі».

Наукова новизна. Запропоновано вдосконалення інформаційної системи керування серверами, що на відміну від існуючих систем використовує аналіз логів та генетичні алгоритми, що дає змогу підвищити середній коефіцієнт використання IT-ресурсів та розширити функціональні можливості системи.

Також було запропоновано математичні моделі та відповідні методи планування і диспетчерування навантаження і розподілу ресурсів фізичних серверів між додатками за доцільними критеріями ефективності за умови виконання ресурсних, технологічних та інших актуальних обмежень.

Огляд програмних засобів управління конфігураціями

Швидкий розвиток віртуалізації разом зі збільшенням потужності серверів, які відповідають промисловим стандартам, а також доступність «хмарних» обчислень привели до значного зростання числа серверів, що потребують управління, як всередині, так і поза організацією.

У цей момент засоби управління конфігураціями і вступають в гру. У багатьох випадках, ми управляємо групами однакових серверів, на яких запущені однакові додатки і сервіси. Вони розміщуються на системах віртуалізації всередині організації, або ж запускаються як «хмарні» і гостьові в віддалених ЦОД. Тому можливість змусити їх усіх виконати задачі поставлені системним адміністратором не може бути знецінена. Це єдиний шлях управляти зростаючими інфраструктурами.

Puppet, Chef, Ansible і Salt були задумані щоб спростити настройку та обслуговування десятків, сотень і навіть тисяч серверів. Це не означає, що маленькі компанії не отримають вигоди від цих інструментів, так як автоматизація зазвичай робить життя простіше в інфраструктурі будь-якого розміру

Огляд програмних засобів управління конфігураціями

Таблиця 1 - Результати тестування в тестовому дата центрі

	Доступність	Сумісність	Управління	Масштабо-ваність	Продуктив-ність	Вартість	Підсу-мок
Вага	20%	20%	20%	20%	10%	10%	
AnsibleWorks Ansible 1.3	9	7	8	8	9	9	8,2 Дуже добре
Enterprise Chef 11.4	9	8	7	9	8	9	8,3 Дуже добре
Puppet Enterprise 3.0	9	9	9	9	9	9	9 Відмінно
SaltStack Enterprise 0.17.0	9	8	9	9	9	9	8,8 Дуже добре

Модель віртуального хостингу

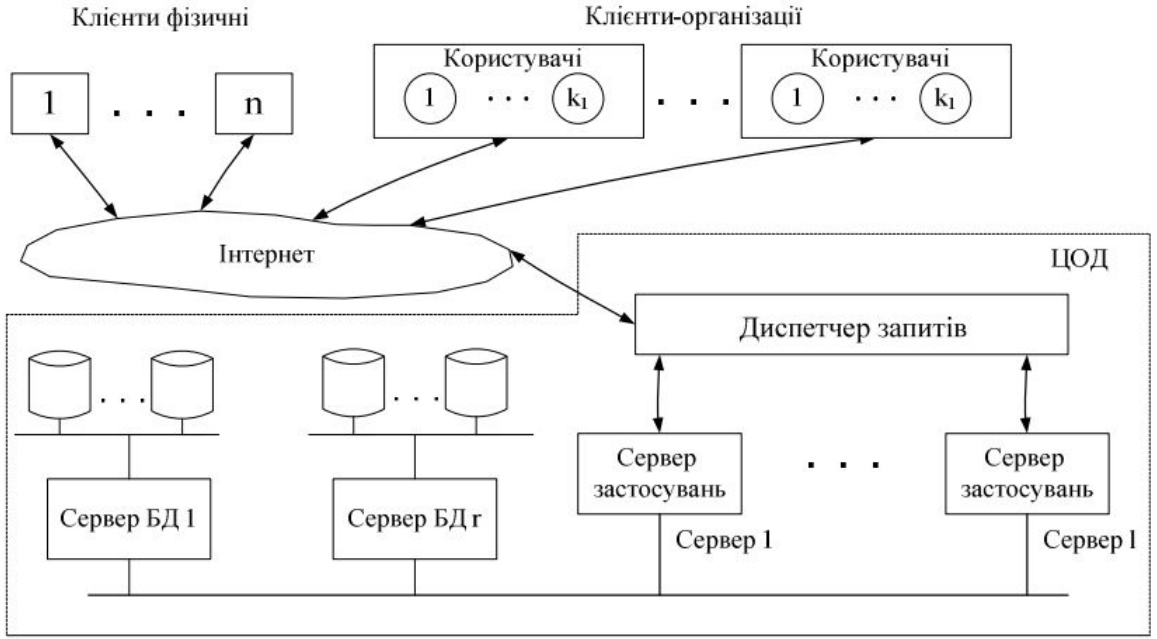


Рисунок 1 – Схема доступу до ресурсів при віртуальному хостингу

Показники та параметри системи

а) $N=\{N1, ... , Nn\}$ – множина серверів, кількість яких дорівнює n;

б) $A=\{A1, ... , Am\}$– множина застосувань, кількість яких дорівнює m;

в) кожний сервер Ni, i=1, ... , n, має чотири параметри, які разом характеризують його потужність:

- Ωi – процесорна місткість сервера Ni;
- Гі– місткість оперативної пам’яті сервера Ni;
- Фі – місткість жорстких дисків сервера Ni;
- Λі – місткість каналів сервера Ni;

г) кожне застосування Aj, j=1, ... , m, має вимоги до:

- ωj– процесорної місткості серверів, на яких розташовані екземпляри застосування Aj;
- γj– до оперативної пам’яті серверів, на яких розташовані екземпляри застосування Aj;
- φj– до пам’яті жорстких дисків серверів, на яких розташовані екземпляри застосування Aj;
- λj– до місткості каналів серверів, на яких розташовані екземпляри застосування Aj.

Показники та параметри системи

Особливість характеристик серверів і вимог застосувань полягає в тому, що вони поділяються на два класи:

- залежні від навантаження – Ωi, ωj;
- незалежні від навантаження – Гі, γj, Фі, φj;

д) вектор C={C1, ... , Cm}, де

$$C = \begin{cases} 1, \text{ якщо СУ може розгортати і згортати застосування Aj;} \\ 0, \text{ в супротивному випадку;} \end{cases}$$

е) wj– важливість застосування j.

Математична модель планування розподілу ресурсів

Задачу планування розподілу ресурсів доцільно розглядати як задачу визначення матриці X=xji, яка задає прив'язку екземплярів застосувань до серверів.

Оптимальне розміщення повинно враховувати такі обмеження:

- сумарні вимоги, незалежні від навантаження, визначені для сервера екземплярів застосувань не перевищують незалежні від навантаження параметри сервера;
- вимоги застосувань, залежні від завантаження, визначають на підставі потенціалу усіх користувачів і повинні бути забезпечені сумарними можливостям залежних від завантаження параметрів тих серверів, на яких розміщуються екземпляри цього застосування;

Як критерії можна використовувати:

- максимальну сумарну важливість застосувань, які отримали ресурси;
- рівномірне завантаження серверів;
- мінімальна сумарна вартість реалізації операцій переходу від попереднього розміщення до нового.

Розв’язання задач оптимального розміщення за допомогою генетичного алгоритму

Крок 0. Визначаємо i=0. Утворення випадкової початкової популяції Pk(0)={p1(0),...,pk(0)} з урахуванням певного відсіювання у перевірці обмежень.

Крок 1. Оцінюємо кожний ланцюжок популяції щодо виконання обмежень відповідної задачі. Для тих ланцюжків pq(0), для яких обмеження виконуються, робиться оцінка f(pq(0)) цільової функції відповідної задачі і вибір найкращого рішення.

Крок 2. Відбираємо представників для нової популяції (ланцюжків з найкращим значенням функції f(pq(0)).

Крок 3. l=l+1. Утворюємо нову популяцію застосуванням оператора рекомбінації.

Крок 4. Поліпшуємо популяцію застосуванням оператора мутації.

Крок 5. Оцінюємо кожний ланцюжок популяції щодо виконання обмежень і функцію f(pq(l)), якщо обмеження виконується.

Крок 6. Якщо рішення ліпше від попереднього, то завершуємо виконання алгоритму, в іншому випадку повертаємося на крок 3. Вибираємо найліпше рішення в утвореній популяції Pk(l).

Розв’язання задач оптимального розміщення за допомогою евристичного алгоритму

Крок 1. Пошук застосування, яке має найменший коефіцієнт ρ.

Крок 2. Пошук сервера з найбільшим коефіцієнтом P.

Крок 3. Якщо для вибраних сервера і застосування не виконуються обмеження, то повертаємося на крок 2 для пошуку наступного сервера.

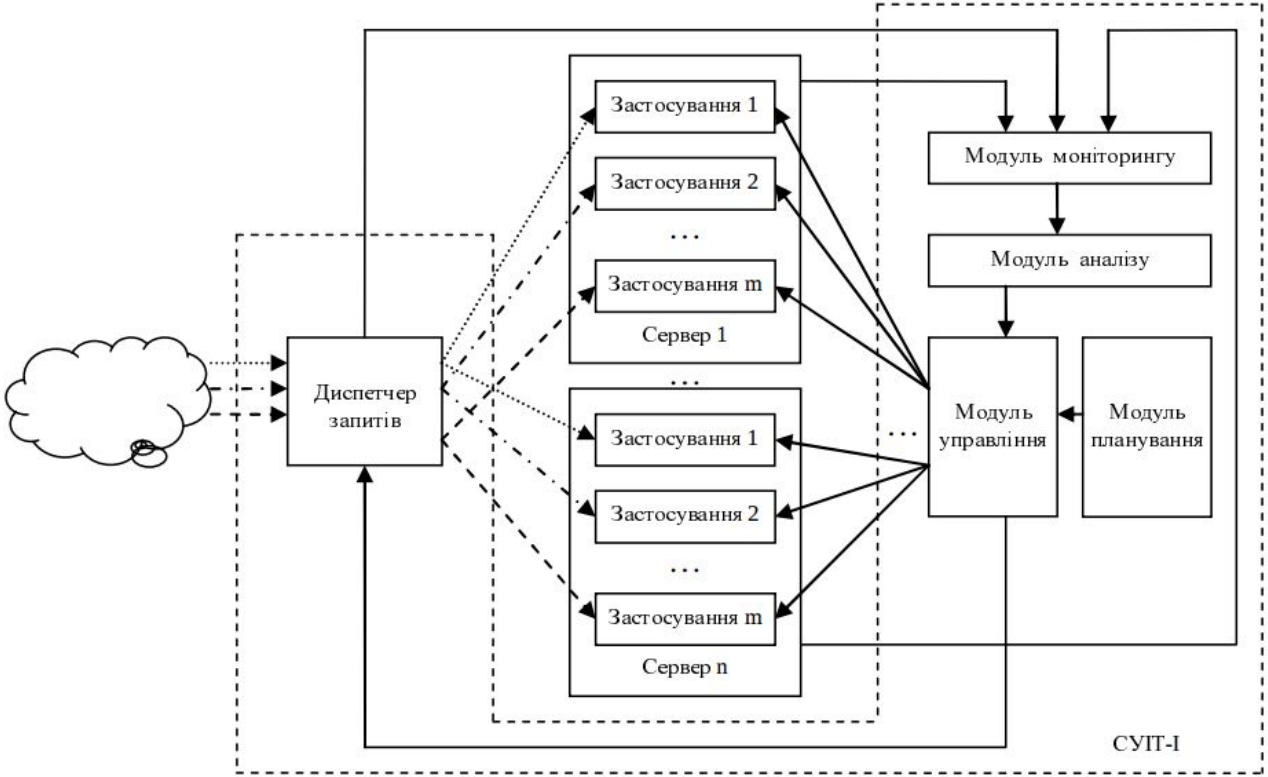
Крок 4. Якщо для вибраних сервера і застосування виконуються обмеження, причому залежні від навантаження вимоги повністю задоволені, то переходимо на крок 5. Якщо виконуються обмеження, але сервер не може повністю задовольнити залежні від навантаження вимоги застосування, то переходимо на крок 6.

Крок 5. Застосування вилучається зі списку застосувань, коригується коефіцієнт P сервера і він вбудовується в список серверів відповідно до нового значення коефіцієнта. Перехід до кроку 7.

Крок 6. Ресурси сервера, що залишилися, застосовують і сервер вилучається зі списку серверів. Переходимо до кроку 2.

Крок 7. Якщо список застосувань порожній, то кінець, інакше переходимо до кроку 1.

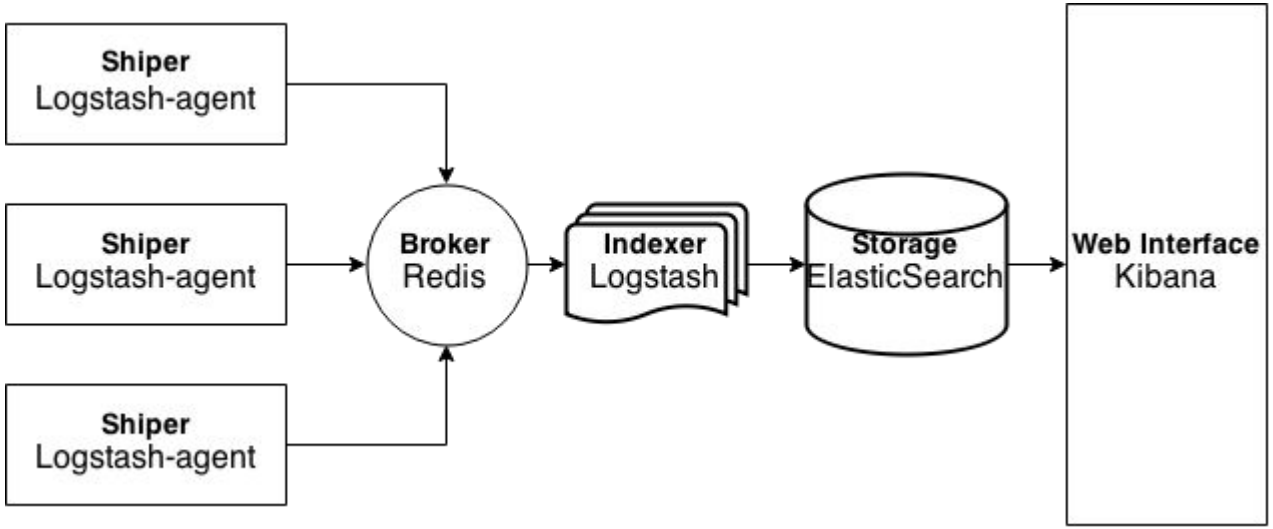
Система управління серверами в “хмарі”



Призначення складових системи управління

- **модуль моніторингу**: збір інформації, контроль стану серверів і застосувань, інформування елементів СУ вищого рівня;
- **модуль аналізу**: виконує опрацювання статистичної інформації.
- **модуль планування**: коли розраховує нове розміщення екземплярів застосувань з урахуванням можливості серверів і вимог застосувань до залежних від навантаження і незалежних від навантаження ресурсів;
- **модуль управління**: впроваджує нове розміщення застосувань у систему з урахуванням попереднього плану розміщення застосувань (розгортає і згортає екземпляри застосувань на серверах), формує правила вибору диспетчером екземплярів застосувань для спрямування запитів користувачів з урахуванням плану розміщення і поточного навантаження серверів;
- **диспетчер запитів**: спрямовує запити користувачів екземплярам застосувань на підставі правил, одержаних від модуля управління.

Система обробки лог-файлів



Висновки

Було запропоновано підхід до планування завантаження і використання ресурсів ЦОД в умовах віртуального хостингу. Представлені моделі і методи, які використані при створенні СУ. Від попередніх технологій управління навантаженням і використанням ресурсів його відрізняють надання адміністраторам можливості використовувати декілька критеріїв, врахування декількох типів ресурсів, гнучкість у переході до нових планів розміщення екземплярів застосувань.

Запропоновані методи були досліджені стосовно одержання розв’язків близьких до оптимального і часу роботи. Обчислення виконувалися для одного кластера, але з різними варіантами кількостей фізичних серверів і застосувань. У цьому разі вимоги застосувань і характеристики фізичних серверів змінювалися. Вхідні дані генерувалися випадково. Результати продемонстрували працездатність обох методів.

Практичний результат полягає в тому, що покращився загальний коефіцієнт використання ІТ ресурсів, який склав 1,11 (тобто 11% економія на витрати з підтриманню існуючої іт інфраструкутури).

Дякую за увагу!