УДК 004.75

АВАРІЙНЕ ВІДНОВЛЕННЯ У ХМАРНИХ МЕРЕЖАХ. ПРОБЛЕМАТИКА WARM СЦЕНАРІЮ АВАРІЙНОГО ВІДНОВЛЕННЯ

DOI 10 36994 / 2788- 5518- 2022-02-04-17

Демидов А.С. *Відкритий міжнародний університет розвитку людини «Україна»,Київ,Україна. anton89d@gmail.com*

Даценко І.П. *Національний Університет Міністерства Оборони України, Київ, Україна. docik_ivan@ukr.net*

Анотація*: Стаття призначена аналізу сучасного стану хмарних систем та сценаріїв аварійного відновлення у хмарах. Проведено аналіз сценарію аварійного відновлення за warm моделлю на прикладі хмарного сервісу розгорнутого у хмарі AWS. Визначені основні проблеми, які приводять до збільшення RTO (recovery time objective). Запропоновані рекомендації щодо зменшення затримки відновлення сервісу після відмови.*

Ключові слова*: хмара, хмарні провайдери, AWS, RTO, аварійне відновлення*

DISASTER RECOVERY IN CLOUD. PROBLEMS OF WARM DISASTER RECOVERY SCENARIO

Anton Demydov*. Open International University of Human Development "Ukraine", Kyiv, Ukraine. anton89d@gmail.com*

Ivan Datsenko. *National Defense University of Ukraine, Kyiv, Ukraine**.** docik_ivan@ukr.net*

Abstract: *Cloud computing is the current routine of using Information Technologies all over the world. This concept is evolving, large cloud computing providers adding more and more functionality, becoming one of the most popular and fastest-growing technologies in the IT field. Disaster recovery is a constant problem in the information technology sphere, but the problem is much broader for cloud providers because they must continue to provide service to customers even during disasters. An accident is an unforeseen event during the system's lifetime. This can be caused by natural disasters, hardware or software failures, human error, or sabotage. This can lead to serious financial losses from a business point of view. For a business, the main purpose of having a disaster scenario is business continuity, which means restoring online services after a failure. The disaster recovery scenario should cover multiple levels of restoring – the data layer, the system layer, and the program layer. There are 3 main approaches of disaster recovery scenarios. Hot model – physical mirroring of infrastructure. Warm model – partial mirroring. Cold model – backups are usually stored outside of the cloud computing provider. A warm model is preferable for medium and small businesses because of its price and complexity. The paper compares different disaster recovery scenarios, their advantages, and disadvantages, and specifies possible problems and high-level solutions for the warm model. Reviewed a real case service deployed to Amazon Web Services provider. Highlighted the need of using Infrastructure as a code to have the ability to change infrastructure as quickly as possible and provided recommendations for decreasing RTO and RPO by using additional load balancers, DNS failover, and PostgreSQL bi-directional replication.*

Key words*: Disaster Recovery, Cloud, AWS, RTO, Warm mo*del

Вступ

Сьогодні попит на послуги провайдерів хмарних обчислювань зростає з великою швидкістю. За оцінкою Fortune ринок досяг значення 480.04 міліардів доларів у 2020 році[1], хоча у 2020 році був на рівні 222 міліардів доларів. З дослідження Gartner хмарні обчислення є однією з десяти найбільших інновацій у сфері інформаційних технологій. [2]

Попередні дослідження стверджують, що хмарні обчислення є формою конвергенції двох основних тенденцій в інформаційних технологіях[3]:

- IT – ефективність, за якою підприємства можуть використовувати сучасні комп'ютерні системи за допомогою високомасштабованих апаратних і програмних ресурсів
- Бізнес – гнучкість, за якою IT використовуються як конкурентна перевага завдляки швидкому розгортанню та співпраці з клієнтами в реальному часі.

Технологія хмарних обчислень має низку особливостей, які роблять її можливим вирішенням різноманітних проблем. Наведенм деякі з цих характеристик [4]:

- Масштабованість
- Віртуалізація
- Надійність
- Універсальність
- Еластичність ресурсів
- Вимірюваність послуг на вимогу
- Економічність
- Керованість
- Аварійне відновлення

Національний інститут стандартів та технологій США дає 5[5] основних характерстик які описують провайдерів хмарного обчислення:

1. Самостійне обслуговування (On-demand self-service).
2. Користувач має можливість отримувати ресурси за запитом
3. Швидкісний мережевий доступ
4. Об'єднання ресурсів в пул
5. Швидкодіюча еластичність - швидке отримання та звільнення ресурсів
6. Врахованість послуг

Однією з найбільш важливих тем у хмарних мережах є аварії та аварійне відновлення після. Аварії яки були спричинені людьскою помилкою або природним явищем можуть привести до дорогих збоїв та частковї або повної зупинки бізнесу. На рис.1 ми можемо побачити залежність суми збитків та часу якщо порівнювати традиційни дата центри та провайдерів хмарних обчислювань.

Рис. 1. Орієнтована швидкість відновлювання хмарного сервіса в порівнянні з традиційним дата центром

Хоча провайдери хмарних обчислень наголошують у своїх угодах про надання послуг (SLA) на доволі високих рівнях доступності сервісів - аварії бувать і у них. Більш того, бізнес потребує мати

сценарій відновлення сервісу у випадку аварії. Можно виділити наступні сценарії які повинні тестуватися:

1. Втрата даних і відновлення резервної копії. Речі, які слід враховувати:
    a. Час, який займе відновлення.
    b. Непередбачувані проблеми під час відновлення.
    c. Потенційні способи покращення часу відновлення.
2. Невдале резервне копіювання. Необхідно ретельно перевіряти резервні копії і проводити тестування розгортання з них.
3. Збої в роботі мережі.
4. Збої обладнання.
5. Тестування безпеки.

Існує декілька підходів для розробки сценарію аварійного відновлення. Усі вони базуються на патерні використання системи та стратегіях резервного копіювання та резервування. Наприклад, резервування використовує статегію роботи паралельної інфраструктури, яка працює у іншому регіоні.

У таблиці ми можемо бачити орієнтований час та швидкість аварійного відновлення та його можливі сценарії.

Таблиця 1.
Моделі аварійного відновлення

Модель	Час синхронізації	Час відновлення	Характеристики резервного копіювання
Гаряча (Hot)	Секунди/Хвилини	Хвилини	Фізичне відзеркалення інфраструктури
Тепла (Warm)	Години	1-24 години	Обмежене фізичне/віртуальне відзеркалення
Холодна (Cold)	Дні	Більше 24 годин	Резервне копіювання поза провайдером хмраного обчислення

Основним завдання розробки сценарію аварійного відновлення – мінімізування проміжку часу від збою до відновлення сервісу – RTO (recovery time objective). Затримки відновлення після відмови складаються з [2]:

- Часу, необхідного на налаштування обладнання
- Час ініціалізації операційної системи
- Час ініціалізації програмного продукту
- Час відновлення стану даних
- Мережеві затримки (IP, DNS)

Показниками ефективного сценарію є мінімальність впливу на нормальну роботу сервіса, географічне розділення та безпековий компонент.

Малі та середні бізнеси, які можуть собі дозволити невеликий час простою у разі аварії віддають перевагу Warm моделі аварійного відновлення.

У статті розглядається реальний кейс використання Warm моделі аварійного відновлення у хмарі Amazon Web Services. Сервіс використовує різноманітні можливості хмари. Основною базою даних є RDS PostgresSQL. Метою дослідження є надання рекомендацій щодо зменшення часу відновлення системи після аварії у іншому географічному регіоні.

Виконання досліджень

Дослідження проводилося на реальному програмному продукті, розгорнутому в хмарі AWS у географічному регіоні us-east-1 (N. Virginia). Піддослідний програмний продукт має мікросервісну архітектуру та використовує наступні хмарні сервіси:

- Elastic Container Service – сервіс для оркестрації програмних продуктів з мікросервісною архітектурою
- Elastic Container Registry – сервіс для зберігання контейнерів
- AWS RDS – сервіс баз даних. Даний програмний продукт використовує Postgres
- AWS Dynamo DB – глобальна NoSQL база даних
- AWS EC2 – сервіс віртуальних машин
- AWS S3 –об'єктно-орієнтований сервіс для зберігання даних
- AWS Lambda – безсерверний, керований подіями обчислювальний сервіс, який дозволяє виконувати код практично для необмеженого типу програми або сервісу без надання серверів та можливості їх безпосереднього обслуговування
- AWS API Gateway – сервіс, призначений для створення, публікації, обслуговування, моніторингу та забезпечення безпеки API у будь-яких масштабах
- Як ми бачимо на рис. 2, усі запити від клієнтів сервіса надходять на Application Load Balancer котрий розподіляю навантаження та виконує функціїї SSL Termination. Основний програмний продукт розгорнуто за допомогою AWS Elastic container service котрий виконує функцію оркестрації контейнерів. Дані зберігаються у декількох базах даних - AWS Dynamo DB та AWS RDS Postgres. Також, є декілька API сервісів, які використовують AWS Lambda та AWS API Gateway. У якості брокера повідомлень використовується RabbitMQ, який розгорнутий за допомогою auto-scaling групи на віртуальних машинах EC2. Додамо, що вся інфраструктура розгорнута за допомогою моделі Infrastructure as a Code та програмного продукта Terraform. Він дозволяє декларативно курувати інфраструктурою, що дає можливість її менеджменту у будь який час, змінюючи лише декілька змінних. Наприклад, ось як декларується Application Load Balancer:

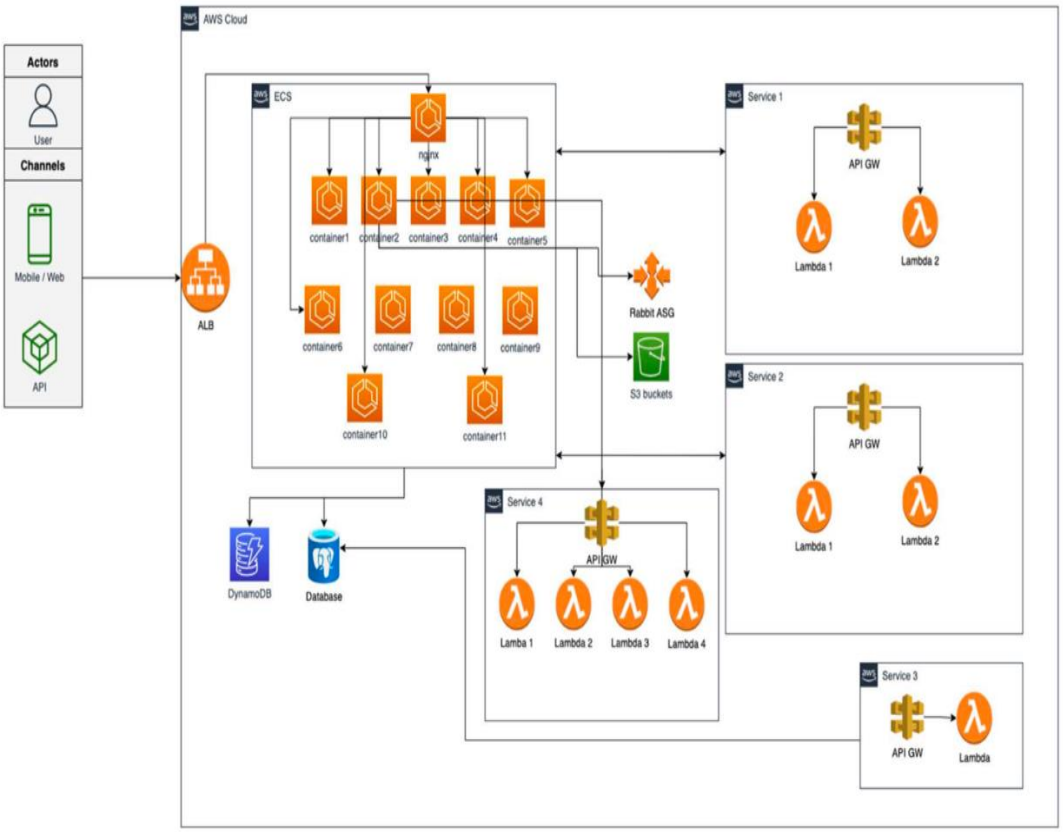

Рис. 2. Інфраструктурна діаграма

```
resource "aws_alb" "vcc-proxy-alb" {
  name               = "${local.env}-${local.application}" # Naming our load balancer
  load_balancer_type = "application"
  subnets            = data.aws_subnets.public.ids
  # Referencing the security group
  security_groups = [aws_security_group.vcc-proxy-alb-sg.id]
}
resource "aws_lb_target_group" "vcc-proxy-alb-tg" {
  name        = "tg-${local.env}-vcc-proxy"
  port        = 80
  protocol    = "HTTP"
  target_type = "ip"
  vpc_id      = data.aws_vpc.vpc.id
  health_check {
    matcher = "200,301,302"
    path    = "/api/health-check/"
  }
  depends_on = [aws_alb.vcc-proxy-alb]
}
resource "aws_lb_listener" "alb-listener" {
  load_balancer_arn = aws_alb.vcc-proxy-alb.arn # Referencing our load balancer
  port              = "443"
  protocol          = "HTTPS"
  ssl_policy        = "ELBSecurityPolicy-2016-08"
  certificate_arn   = local.cert_arn
  default_action {
    type = "fixed-response"
    fixed_response {
      content_type = "text/plain"
      message_body = "hello"
      status_code  = "200"
    }
  }
}
resource "aws_lb_listener" "alb-listener-redirect" {
  load_balancer_arn = aws_alb.vcc-proxy-alb.arn
  port              = "80"
  protocol          = "HTTP"

  default_action {
    type = "redirect"

    redirect {
      port        = "443"
      protocol    = "HTTPS"
      status_code = "HTTP_301"
    }
  }
}


resource "aws_lb_listener_rule" "redirect-to-vcc-proxy" {
  listener_arn = aws_lb_listener.alb-listener.arn
  action {
    type             = "forward"
    target_group_arn = aws_lb_target_group.vcc-proxy-alb-tg.arn
  }
  condition {
    host_header {
      values = ["${local.proxy_domain}.com"]
    }
```

```
  }
}
```

Щодо можливих змін – усе конфігурується у файлі variables.tf, що дає можливість визначити скільки ресурсів має той чи інший географічний регіон, вносити зміни у версіях тощо.

Базуючись на можливих сценаріях аварійного відновлення та вимогах бізнесу для даної системи була обрана Warm модель сценарію аварійного відновлення. У іншому географічному регіоні була розгорнута така сама архітектура, за допомогою коду terraform. Для зменшення витрат бізнесу інфраструктура у іншому регіоні має вдвічі менше ресурсів, тобто кількість працюючих контейнерів, розмір бази даних.

Для резервування бази даних був використаний механізм read-replica, який дозволяє мати репліку бази даних у іншому регіоні, яка доступна тільки на читання. (рис 3)

Зазначимо, що версії програмних продуктів мають бути оновлені до тих самих версій, які є у основного стеку. Це може бути досягнено за допомогою CI/CD принципа та постійного моніторингу. У випадку аварії, інженери, які відповідають за інфраструктуру повинні отримати дозвіл на ініціювання аварійного сценарію. Якщо такий дозвіл є, то вони повинні виконати наступні кроки:

- Ініціювання репліки бази даних як основної
- Заміна записів DNS для усіх сервісів, які використовуються у даній системі
- Моніторинг журналів логування резервної системи та збільшення кількості ресурсів за умов великого навантаження та правильної роботи.
- При обранні WARM сценарію аварійного відновлення ми можемо
- мати доволі швидке переключення DNS записів, але для деяких
- клієнтів це може стати проблемою, якщо вони використовують власні
- DNS сервера з великим часом кешування. Тому, TTL DNS записів має
- бути доволі малим. Але, основною проблемою є затримка реплікації
- бази даних між географічними регіонами (рис. 4).
- При моделюванні аварії у даній конкретній системі ми можемо стверджувати, що у випадку аварійного сценарію час на відновлення програмного стеку у іншому географічному регіоні може коливатися від 45 хвилин до декількох годин. Зведемо дані у таблицю:

Таблиця 2.
Час ініціювання аварійного стеку

Сервіс	Час на зміну	Коментарі
DNS записи	1-15 хвилин	Залежить від DNS TTL та кількості DNS записів
Read-Replica Promotion	15-120 хвилин	Залежить від об’єму бази даних та її конфігурації

Це означає, що при аварії є шанс втрачання певної кількості даних. Також, швидкість переключення репліки бази даних до мастера займає значний час.

Для вирішення цих проблем є декілька опцій:

- Використання між регіональних балансувальників нагрузки, які здатні переключати трафік на інший регіон автоматично, або спеціалізованих сервісів, таких як AWS Global Accelerator.
- Використання master-master реплікації бази даних, або bi-directional реплікацію.
- Використання глобальних баз даних.
- Використання контейнеризації для зменшення часу ініціалізації.

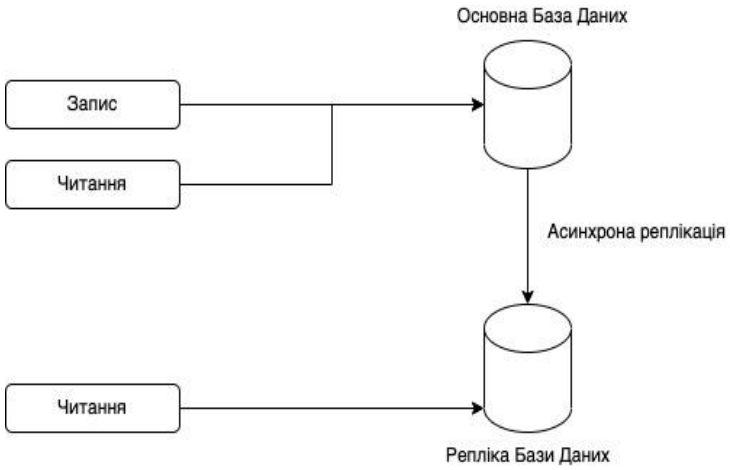

Рис. 3. Реплікація бази даних у інший географічний регіон

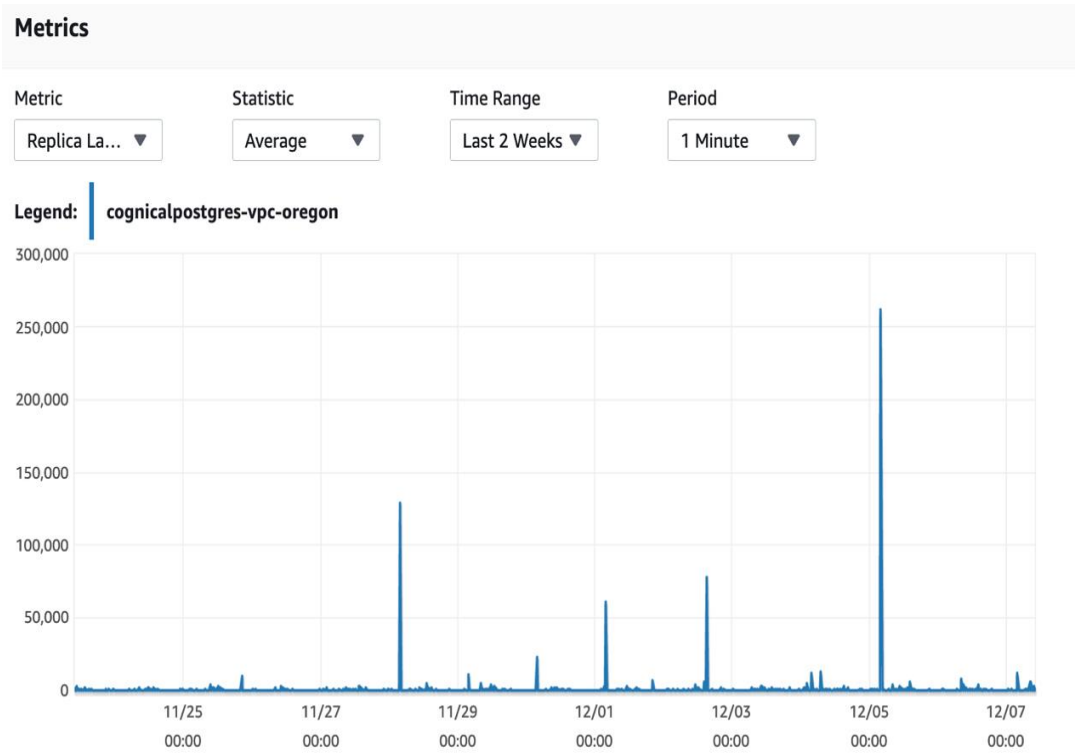

Рис. 4. Затримка реплікації бази даних у інший географічний регіон

Висновки

У статті розглянуті основні сценарії аварійного відновлення у хмарних мережах.

Проведені дослідження проблематики Warm моделі аварійного відновлення на основі працюючого сервіса, розгорнутого у хмарі AWS.

Основними проблемами, які збільшують час відновлення сервісу після аварії є швидкість переключення DNS записів після аварії і швидкість переключення мастера бази даних у інший географічний регіон.

Запропоновані методи оптимізації RTO – використання bi-directional реплікації бази даних, що дозволить мати фактично мастер-мастер реплікацію. Це означає, що буде збережений час, який зараз витрачається на переключення. Також, запропоновано використання між-регіональних балансувальників навантаження, які дозволять зменшити до 0 час переключення DNS записів.

Література

1. Електронний ресурс. Fortubebusinessinsights. Cloud computing Market size, Analysis. URL: https://www.fortunebusinessinsights.com/cloud-computing-market-102697
2. Електронний ресурс. Gartner, “Gartner identifies the top 10 strategic technologies for 2011”, URL: http://www.gartner.com/it/page.jsp?id=1454221
3. Associated Risks of Cloud Computing for SMEs. Open International Journal of Informatics (OIJI), (2012) ,1, pp.37-45.
4. Introduction to Cloud Computing, White Paper,Dialogic Corporation, 2010. R. Buyya, J. Broberg, and A. Goscinski, “Cloud Computing Principles and Paradigms”, John Wiley & Sons, 2011.
5. Таненбаум Э.. Современные операционные системы. Питер, 2018. 553 ст.

References

1. Fortubebusinessinsights. Cloud computing Market size, Analysis. URL: https://www.fortunebusinessinsights.com/cloud-computing-market-102697
2. Gartner, “Gartner identifies the top 10 strategic technologies for 2011”, URL: http://www.gartner.com/it/page.jsp?id=1454221
3. Associated Risks of Cloud Computing for SMEs. Open International Journal of Informatics (OIJI), (2012) 1, pp.37-45.
4. Introduction to Cloud Computing, White Paper,Dialogic Corporation, 2010. R. Buyya, J. Broberg, and A. Goscinski, “Cloud Computing Principles and Paradigms”, John Wiley & Sons, 2011
5. Tanenbaum E. Sovremennye operacionnye sistemy. [Modern operating systems]. Piter, 2018. 553 s.