Disaster Recovery projektu SQL Server z wykorzystaniem AWS

Grzegorz Prasek

May 20, 2025

Cel projektu

Celem projektu było przygotowanie kompletnego rozwiazania Disaster Recovery (DR) dla systemu przechowujacego dane w bazie Microsoft SQL Server. System został uruchomiony na instancji EC2 w regionie N. Virginia (USA), a jego architektura i konfiguracja odpowiadaja realnemu scenariuszowi odzyskiwania danych po awarii. Projekt oparty został na wcześniej utworzonym środowisku bazodanowym i rozszerzony o mechanizmy automatycznego backupu oraz przenoszenia danych do innego regionu (Oregon), zgodnie z wymogami zadania DR.

W ramach realizacji wykonano pełna konfiguracje serwera produkcyjnego, przygotowano zapasowy serwer DR w innej lokalizacji geograficznej oraz wdrożono automatyczny mechanizm wykonywania codziennych kopii zapasowych bazy danych i ich przesyłania do zapasowego środowiska. Dodatkowo zrealizowano integracje z usługa AWS Lambda, która po każdym wykonanym backupie zapisuje informacje do dziennika zdarzeń CloudWatch Logs — umożliwiajac monitorowanie i audyt operacji.

Projekt uwzglednia również parametry RTO i RPO (opisane także w pliku README) oraz ich praktyczne uzasadnienie. Całość została zaprojektowana z myśla o niskim koszcie operacyjnym oraz kompatybilności z budżetem środowiska Learners Lab.

Repozytorium projektu

Wszystkie pliki źródłowe, skrypty, kod funkcji Lambda oraz instrukcje konfiguracyjne znajduja sie w repozytorium GitHub: `https://github.com/Grzesiek1212/EC2-chmurki.git`

Plik `README.md` zawiera również opis założeń dotyczacych parametrów RTO i RPO oraz instrukcje krok po kroku umożliwiajace samodzielne uruchomienie projektu.

Etapy realizacji

Etap 1: Uruchomienie głównego serwera (N. Virginia)

- Utworzono instancje EC2 z Ubuntu w regionie `us-east-1`.
- Zainstalowano Microsoft SQL Server oraz narzedzia sqlcmd.

- Utworzono baze danych `poziomka`.
- Wykonano pierwszy backup bazy danych:

```
/opt/mssql-tools/bin/sqlcmd -S localhost -U sa -P '*********' \
 -Q "BACKUP DATABASE poziomka TO DISK = '/var/opt/mssql/db_backups/poziomka.bak'"
```

Etap 2: Konfiguracja dostepu miedzy serwerami

- Wygenerowano pare kluczy SSH (.pem).
- Skonfigurowano zdalny dostep pomiedzy instancjami EC2 za pomoca SCP.
- Rozwiazano problemy z uprawnieniami plików przy użyciu `chmod` i `icacls`.

Etap 3: Utworzenie serwera zapasowego (DR) w Oregonie

- Utworzono nowa instancje EC2 z Ubuntu w regionie `us-west-2`.
- Zainstalowano SQL Server, mssql-tools oraz wymagane biblioteki.
- Utworzono katalog `/var/opt/mssql/backups` z odpowiednimi uprawnieniami.

Etap 4: Transfer i przywrócenie backupu

- Backup przesyłany z Virginii do Oregonu:

```
scp -i ~/dr-key.pem poziomka-latest.bak ubuntu@IP_DR:/var/opt/mssql/backups/
```

- Odtworzenie bazy na serwerze DR:

```
RESTORE DATABASE poziomka
FROM DISK = '/var/opt/mssql/backups/poziomka.bak'
WITH MOVE 'poziomka' TO '/var/opt/mssql/data/poziomka.mdf',
     MOVE 'poziomka_log' TO '/var/opt/mssql/data/poziomka_log.ldf',
     REPLACE;
```

Etap 5: Automatyzacja: codzienny backup i wysyłka

- Stworzono skrypt `backup_poziomka.sh`, który:
  (a) O godzinie 2:00 wykonuje backup lokalny.
  (b) O 3:00 przesyła backup na serwer DR (Oregon).
  (c) Nastepnie wywołuje funkcje Lambda do logowania operacji.
- Skrypt został dodany do `crontab`:

```
0 2 * * * /home/ubuntu/backup_poziomka.sh >> /home/ubuntu/backup_log.txt 2>&1
```

Etap 6: Serverless: funkcja Lambda do logowania

- Stworzono funkcje AWS Lambda o nazwie `LogBackupEvent` w .NET 8.
- Funkcja odbiera parametry `region` i `message` i zapisuje log do CloudWatch:

```
aws lambda invoke \
  --function-name LogBackupEvent \
  --region us-east-1 \
  --payload '{"queryStringParameters": {"region": "Virginia", "message": "Backup zak
  /tmp/lambda-output.json
```

- Przypisano instancji EC2 odpowiednia role IAM (`LabInstanceProfile`), aby miała uprawnienie do wywoływania funkcji Lambda.
- Logi funkcji można śledzić w CloudWatch Logs: `/aws/lambda/LogBackupEvent`.

Disaster Recovery

System tworzy automatyczne kopie zapasowe bazy danych poziomka przy pomocy sqlcmd, a skrypt backup poziomka.sh uruchamiany jest cyklicznie przez cron. Kopie zapisywane s na serwerze, a dodatkowo replikowane do innego regionu (S3).

- **RTO (Recovery Time Objective)**: 30 minut Czas potrzebny na odtworzenie aplikacji oraz przywrócenie bazy danych z backupu to około 30 minut, ponieważ wystarczy uruchomić gotowa instancje EC2 i odtworzyć .bak.
- **RPO (Recovery Point Objective)**:1 dzień Backup wykonywany raz dziennie, wiec maksymalna strata danych to 24 godziny. Jest wykonywany Codzienie o godzinie 2 w nocy na obecnym serwerze i o godzinie 3 w nocy jest wykonwany backup na baze stojaca na innym serwerze w innym regione.

Opis architektury

Architektura projektu opiera sie na dwóch instancjach EC2 działajacych w różnych regionach AWS — serwer główny w regionie N. Virginia oraz serwer zapasowy (DR) w Oregonie. Na serwerze głównym znajduje sie baza danych Microsoft SQL Server, która codziennie o 2:00 jest backupowana, a o 3:00 backup jest przesyłany na serwer DR. Dodatkowo, po każdej operacji backupu uruchamiana jest funkcja AWS Lambda, która rejestruje zdarzenie w CloudWatch Logs. Lambda uruchamiana jest bezpośrednio ze skryptu bash działajacego na serwerze EC2, a dostep umożliwia przypisana rola IAM. Architektura wykorzystuje usługi EC2, Lambda, IAM, SCP oraz CloudWatch.

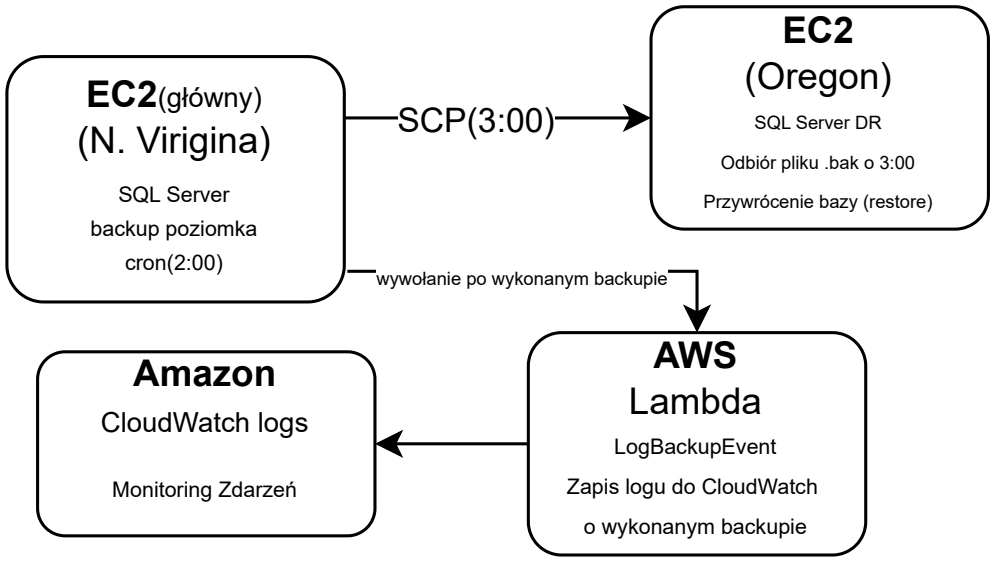

Figure 1: Diagram architektury systemu Disaster Recovery z wykorzystaniem AWS EC2, Lambda i CloudWatch.

Podsumowanie

Projekt Disaster Recovery został w pełni zrealizowany:

- Codzienne backupy bazy danych sa wykonywane i przesyłane do drugiego regionu.
- Logowanie przebiegu backupu odbywa sie automatycznie dzieki funkcji AWS Lambda.
- System gotowy jest do odtworzenia danych w przypadku awarii, zgodnie z założonymi parametrami RTO i RPO.
- Projekt wykorzystuje wiele usług AWS: EC2, SSM, Lambda, IAM oraz CloudWatch.