DEVOPS

5. forduló

<epam>

A kategória támogatója: EPAM

Ismertető a feladathoz

Felhasznált idő: 00:00/10:00 **Elért pontszám:** 0/7

1. feladat 0/1 pont

Amennyiben a Canary release-t DNS routing policy használatával szeretnénk implementálni, melyik AWS Route53 policy erre az ideális?

Válasz

- ○ Failover routing policy
- ○ Latency routing policy
- ◉ Weighted routing policy
  Ez a válasz helyes, de nem jelölted meg.
- ○ Multivalue answer routing policy

Magyarázat

Az ideális választás a Weighted routing policy, mivel finoman szabályozható, hogy mekkora mennyiségű forgalmat tereljen a policy. Ezt javasolja az AWS dokumentáció is erre a célra. https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy-weighted.html

2. feladat 0/1 pont

Melyik állítás ***NEM igaz*** az adott load balancer-ről, ahol az Application Load Balancer-t az ALB és a Network Load Balancer-t az NLB jelöli?

Válasz

- ○ Az ALB az Open Systems Interconnection (OSI) model 7-es rétegében működik
- ○ Az NLB támogatja az Elastic IP hozzárendelést

- [x] Az NLB támogatja a request path based routing-ot
  Ez a válasz helyes, de nem jelölted meg.
- [ ] Az ALB támogatja a user authentication-t

Magyarázat

Az ALB az OSI 7-es rétegében, míg az NLB a 4-es rétegében működik, az ALB az amelyik a path based routing-ot támogatja.
https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

https://docs.aws.amazon.com/elasticloadbalancing/latest/network/introduction.html

3. feladat 0/1 pont

Mire szolgál a field-level encryption?

Válasz

- [ ] Az RDS-ben tárolt adatokat a mezők szintjén titkosítani lehessen
- [x] CloudFront használta esetén a felhasználók már feltöltésnél titkosíthatják az érzékeny adataikat a POST requestekben
  Ez a válasz helyes, de nem jelölted meg.
- [ ] AWS Serverless Aurora esetén bekapcsoltható extra biztonságot jelentő opció az adatbázisban tárolt mezők titkosítására
- [ ] AWS Aurora esetén bekapcsoltható extra biztonságot jelentő opció az adatbázisban tárolt mezők titkosítására

Magyarázat

CloudFront használata esetén a HTTPS tunnel belselyében az érzékeny adatok titkosítva kerülnek továbbításra:
https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/field-level-encryption.html

4. feladat 0/1 pont

AWS cloudban egy Terraformmal létrehozott infrastruktúra esetén mindenképpen szeretnél törölni és újra létrehozni egy létező resourcot, függetlenül attól, hogy adott esetben erre nem feltétlen lenne szükség. Erre a **terraform taint** parancs használható. Hogy működik, mi a hatása ennek a parancsnak?

Válasz

- [ ] Egyből létrehozza az új resource-ot, majd a régit törli
- [ ] Egyből törli a resource-ot
- [x] Megjelöli a resource-ot a state fájlban, a következő terraform apply-nál törli és újra létrehozza
  Ez a válasz helyes, de nem jelölted meg.
- [ ] Létrehozza újra a resource-ot, a régi törléséről nekünk kell gondoskodni, hogy legyen kontrollunk az esetleges adatvesztés felett.

Magyarázat

A taint csak a state fájlban jelöli meg a resource és a következő apply ciklus fogja törölni, majd újra létrehozni.

5. feladat 0/1 pont

Egy publikus cloudban futtatott alkalmazásunkat szeretnénk tanúsítvánnyal ellátni. Mire tudnánk használni ebben az esetben a CRL-t?

Válasz

- ○ A CRL a kiállított tanúsítványok sorszámát tartalmazza, így ellenőrizhető, hogy az adott tanúsítvány tényleg legitim személy által lett kiállítva.
- ● A CRL a visszavont/érvénytelen tanúsítványok listája, aminek segítségével visszavonható egy tanúsítvány, még annak lejárati ideje előtt.
  Ez a válasz helyes, de nem jelölted meg.
- ○ A tanúsítványoknál nem létezik ilyen.
- ○ Ez a köztes aláíró tanúsítvány, amely szükséges, hogy a böngésző megbízzon a kiállított tanúsítványban függetlenül attől, hogy a gyökér tanúsítványt megbízhatónak ítéli-e meg vagy sem.

Magyarázat

A CRL egy publikusan elérhető lista, amely tartalmazza azon tanúsítványok azonosítóit, amelyeket a kiállító szerv visszavont/érvénytelenített, így biztosítható, hogy invalid tanúsítványok akkor se legyenek használhatóak, ha azoknak az érvényessége még nem járt le vagy az aláírása egyébként valid.

6. feladat 0/1 pont

A használt CI/CD pipeline során nagyon sokszor találkozunk megbízhatatlan (flaky) tesztekkel, amelyek bizonyos esetekben megfelelően lefutnak, máskor pedig hibát jeleznek.

Milyen problémá(k)ra utalhatnak a rendszerünkben az ilyen típusú tesztek?

Válaszok

- ☑ Több különböző környezetet használunk véletlenszerűen és ezek konfigurációja eltérő lehet
  Ez a válasz helyes, de nem jelölted meg.
- ☑ Tesztek valamilyen módon függenek egymástól (pl. egyik teszt adatokat használ fel másik tesztből), így amikor nem jó sorrendben, nem jó időzítéssel futnak a tesztek, akkor hiba keletkezik.
  Ez a válasz helyes, de nem jelölted meg.
- ☑ A tesztelt kód nem determinisztikus, így ugyan azon bemenő paraméterekre futtatásonként különböző eredményt adhat.
  Ez a válasz helyes, de nem jelölted meg.
- ☐ Való életben sem tudjuk minden esetben kontrollálni a környezeteket, így jó is, hogy nem törekszünk mindig determinisztikus működésre. Azon kívül, hogy újraindítjuk a hibás eredményt adó tesztet, nem szükséges a problémával foglalkozni.

Magyarázat

A megbízhatatlan tesztek sokszor tudnak gondot okozni, növelik a fejlesztési időt, frusztrációt kelthet a kollégákban, extra költségeket generálhatnak, így érdemes megvizsgálni, mi áll a háttérben és javítani a tesztek hibáit.

7. feladat 0/1 pont

A CI/CD környeztetben egy git repository változását, különböző eseményeit szeretnéd figyelni. Milyen lehetőségied vannak ennek automatikus megvalósítására?

Válaszok

- [x] Néhány percenként ellenőrizzük (poll) a repository állapotát, és amennyiben változás van, futtatjuk a kívánt taskot.
  Ez a válasz helyes, de nem jelölted meg.
- [x] Webhook segtségével a repository tud adott esemény bekövetkeztekor egy jelzést generálni pl. a Jenkins felé, amire különböző jobok indíthatóak.
  Ez a válasz helyes, de nem jelölted meg.
- [ ] Automatikusan nem megoldható, git változás esetén szükséges az adott job kézi elindítása Jenkins szerveren.
- [ ] A gépünkön lokális git commit futtatása után indíthatunk egy helyi scriptet, ami el tudja indítani a Jenkins szerveren a szüséges jobokat.

Magyarázat

CI/CD rendszer és Git repository integrálására alapvetően a pollingot vagy a webhookokat tudjuk használni.

Polling esetén azonban az esetek nagy részében feleslegesen indítjuk a kérést pazarolva ezzel a rendelkezésre álló erőforrásokat, illetve a lekérdezések fix intervalluma miatt valamennyi késleltetés lesz az esemény bekövetkezte és annak észlelése között.