Arhitekture CC

Veljko Petrović
Januar, 2023

Arhitekture Cloud Computing

Načini organizacije resursa

Osnovne arhitekture CC

Arhitektura distribucije posla

- Workload distribution architecture
- Ovo je bazična inkarnacija onoga što već neko vreme radimo
- Replikacija resursa i load balancer ispred svega toga

Arhitektura kombinacije resursa

- Resource pooling architecture
- Još jedna fundamentalna stvar koju smo obradili
- Ovde je situacija kada želimo da kombinujemo više resursa u jednu celinu i onda uzimamo koliko treba
- Ovo je jako bitan element CC arhitekture

Arhitektura kombinacije resursa

- Možemo da kombinujemo više stvari
  - Fizičke servere
  - Virtuelne servere
  - Resurse za smeštanje podataka
  - Resurse za prenos podataka
  - CPU/Memoriju
- Šta kombinujemo zavisi od mehanizama koje koristimo

Arhitektura dinamičke skalabilnosti

- Dynamic scalability architecture
- Još jedna stvar koju smo pokrili: ovo je kada se konfiguracija našeg sistema menja kada se promene uslovi
- Ovde arhitektura zahteva sve što ima arhitektura distribucije posla plus još mehanzime za merenje uslova i skaliranje resursa

Arhitektura dinamičke skalabilnosti

- Skaliranje može doći u mnogo oblika
  - Dinamičko horizontalno skaliranje
    - Kada dodajemo još repliciranih resursa da bi olakšali posao
  - Dinamičko vertikalno skaliranje
    - Kada dodajemo još resursa *postojećim instancama* automatski
  - Dinamička relokacija
    - Kada same podatke pomeramo sa jednih resursa na druge da bi modifikovali karakteristiku performansi.

Arihtektura elastičnog kapaciteta resursa

- Ovo je povezano sa prethodnim, samo što se modifikuje sistem skaliranja tako da se resursi ne samo automatski alociraju nego se i automatski oslobađaju.
- Skaliranje, dakle, radi i na gore i na dole.
- Ovo je ono što smo istraživali ponajviše kroz to kako AWS to radi, uprkos tome valja napomenuti: to je danas manje-više fundamentalna osobina svih provajdera ove usluge, nije jedinstvena za AWS.

Arhitektura balansa opterećenja servisa

- Ovo je arhitektura koja nastaje kompozicijom i modifikacijom ranije pomenjenih elementarnih arhitektonskih obrazaca
- Specifično nastaje kada se arhitektura rasporeda rada kombinuje sa dinamičkom skalabilnošću ili elastičnom arhitekturom i modifikuje da još više odgovara servisu koji je dominantno baziran u oblaku.
- Sepcifična modifkacija ulkjučuje i redudantne računarske resurse čime se ne modifikuju performanse no robustnost sistema.

Arhitektura proboja

- Cloud bursting architecture
- Ovo je posebna varijanta skaliranja namenjena da radi u hibridnom režimu
- Hibridni režim u ovom kontekstu znači da se većinu vremena radi na računarima koje su lokalna instalacija nekakve institucije, ali u slučajevima iznimnog opterećenja se skaliranje vrši na oblak, tj. sistem *probije* na oblak.
- Ovo je dobar način da se ojača lokalna instalacija koja dobro radi i ne treba joj migracija na oblak.

Arhitektura proboja

- Proboj može da se radi i na razne druge načine
- U zadnje vreme je popularna tehnika izmeštanja radnih stanica sa teškim opterećenjem na oblak, kada je u pitanju recimo AI ili razvoj računarskih igara
- Video igre već neko vreme pokušavaju da implementiraju arhitekturu proboja

Arhitektura elastične provizije diskova

- Ovo je izuzetno specijalizovana arhitektura gde se količina alociranih resursa za čuvanje podataka varira u skladu sa potrebom.
- To znači da se alocira još memorije kada treba ili, paralelno, se nema potrebe, se oslobađa prostor.
- Ovo naravno samo radi ako ima arhitektura kombinacije resursa u pozadini koja pruža te resurse
- Uglavnom služi da minimizuje troškove koji su obično naplaćeni u odnosu na angažovan kapacitet, ne na korišćen.

Arhitektura skladištenja podataka sa redudantnošću

- Ovo je varijacija na elastično skaliranje postavnjeno nad arhitekturu elastične provizije diskova
- Efektivno: želimo da implementiramo nešto vrlo kao RAID samo nad individualnim resursima u oblaku, a ne na individualnim diskovima
- To znači da je ovo mehanizam automarizovane replikacije podataka na više resursa ne bi li se povećala robusnost
- Ovo može biti *simetrično* gde se replicira po identičnim instancama ali naročito zanimljivo jeste ako ovo uradimo *asimetrično* odnosno repliciramo stvari po instancama sa različitim karakteristikama

Arhitektura skladištenja podataka sa redudantnošću

- Primer ovoga (koji bi sam po sebi bio neka vrsta napredne arhitekture) je situacija gde imamo tri moguća načina skladištenja podataka:
  - Skladište A - Brzo ali jako nepouzdano i skupo
  - Skladište B - Sporije ali razumno pouzdano i razumne cene
  - Skladište C - Izuzetno sporo ali jako pouzdano i jeftino
- Onda na skladište A stavimo stvari sa kojima se trenutno radi, ali se ne uzdamo u njih i brzo ih vraćamo na skladište B gde je autoritativna kopija podataka, dok je skladište C rezervna kopija svega.

- A, B, i C, mogu odgovarati memorijskom kešu, SSD-u, i arhiviranju na magnetnim trakama.

Napredne arhitekture

Rezultat kompozicije i ekstenzije

Arhitektura klasterovanja kroz hipervizor

- Ovo je jako često korišćena arhitektura danas, naročito kod servisa koji pružaju CC usluge.
- Ideja je da se koristi kombinacija elastičnosti i objedinjenja resursa da bi se kreirao glatko skalirajući broj virtuelnih mašina
- Arhitekrura uzme određeni broj fizičkih servera (pre ili kasnije sve mora da počiva na nekakvom fizičkom računaru koji nešto radi) i nad svakim podigne hipervizor

Arhitektura klasterovanja kroz hipervizor

- Svaki od računara nad kojima je podignut hipervizor može da na sebi ima ma koju virtuelnu mašinu
- To znači da smo u velikoj meri objedinili resurse svih fizičkih računara u jedno i možemo da ih delimo na virtuelne mašine
- Sledeći korak jeste da se provizionisanje virtulenih mašina napravi tako da je robusno, to jest, neosetljivo na otkaze fizičke opreme.

Arhitektura klasterovanja kroz hipervizor

- Postoji uvek menadžment sistem koji kontroliše ovaj sistem hipervizora
- Deo tog sistema je podsistem za nadzor koji prati tkzv. ‘otkucaj srca’ signal koji funkcionalne virtuelne mašine i, ključno, hipervizori redovno šalju
- Izostanak signala je indikator otkaza
- U slučaju otkaza dinamički se alocira neki drugi hipervizor na nekom drugom fizičkom računaru

- Ali ovo ništa neće pomoći ako 10 od naših 150 fizičkih računara pokreću sve VM instance: neophodno je pametno distribuirati i opterećenje fizičkih resursa.

Arhitektura Virtualnog Balansiranja Instanci

- Ovo je arhitektonska ekstenzija i modifikacija balansiranja opterećenja koja se fokusira na balansiranje opterećenja nad fizičkim instancama koje preko hipervizora predstavljaju domaćine virtuelnim instancama
- Kreiranjem novih virtuelnih instanci ili vertikalnim skaliranjem tih instanci i usmeravanjem zahteva intelegentno je moguće ravno izbalansirati opterećenje nad svim virtuelnim instancama.

Arhitektura Virtualnog Balansiranja Instanci

- Ovo se radi tako što mehanizma za upravljanje uzima u obzir opterećenost fizičkih instanci (što je moguće skupljati kroz telemetriju) kada pravi nove instance
- Takođe bitno je operacija rebalansa gde se rasterećuju opterećeni fizički serveri tako što se neke njihove virtuelne mašine migriraju na druga mesta.

Arhitektura realokacije servisa bez prekida

- Ovo je ključan element za robusnost baš kao i za virtualno balansiranje instanci
- Ideja je da se virtuelna mašina može premestiti iz jednog hipervizora na drugi dinamički tokom rada sistema na način koji ne prekida pružanje usluge
- Naravno ovo samo radi ako servis nije doživeo otkaz pre migracije (mada videti sledeću arhitekturu)

Arhitektura realokacije servisa bez prekida

- Tehnika se svodi na fundamentalnu osobinu virtuelnih mašina: ako repliciramo disk i VM konfiguraciju replicirali smo mašinu do na restart
- Ako repliciramo i memoriju onda možemo mašinu replicirati maltene do na instrukciju
- Tipično se ne ide tako daleko: umesto napravi se nova instanca koja je pokrenuta kao i prva i dovedena u stanje da može da obrađuje zahteve
- Onda se zahtevi preusmere na nju a prva se onda bezbedno terminira: sa tačke gledišta korisnika ništa se nije desilo.

Arhitektura realokacije servisa bez prekida

- Relokacija radi još bolje kada su svi podaci već na nekakvom (repliciranom) sistemu za skladištenje podataka
- Onda je moguće koristiti, npr. mehanizam sa copy-on-write semantikom ili štagod slično za gotovo trenutno kopiranje
- Sa druge strane ako se koriste tehnike virtualizacije sa direktnim I/O pristupom (više o tome kasnije) onda relokacija nije moguća: VM-ovi su vezani za svoj hardver.

- Naravno, uvek je moguće zaustaviti instancu i pokrenuti je drugde, ali prava relokacija neće raditi tako glatko.

Arhitektura sa nultim vremenom otkaza

- Ovo je arhitektonski šablon koji služi da, u slučaju otkaza, kranji korisnik ne primeti da je do njega došlo
- U pitanju je kombinacija balansiranja i replikacije sa relokacijom
- Ako već imamo balansiranje i virtuelno i između fizičkih instanci, onda nema problema: sve je već replicirano i balansirano.

Arhitektura sa nultim vremenom otkaza

- Ali šta ako imamo samo jedan jedini server koji nije horizontalno skaliran, zar to onda nije SPOF?
- Ne ako imamo server koji je kopija prvog (realokacija) i koji se izvršava na drugom fizičkom računaru (balansiranje) i prema kome u slučaju detekcije otkaza (klasterovanje hipervizorom) će balanaser (balansiranje) usmeriti zahteve.
- Ovo znači da u slučaju da se vidi otkaz, posao prelazi na identičan već-spreman sistem koji se istog trenutka i sam replicira u slučaju da on otkaže.

Arhitektura balansa nad oblacima

- Kao što imamo balans nad virtuelnim instancama i balans nad fizičkim instancama možemo imati balans i nad različitim oblacima.
- Ovo ima nekoliko svrha:
  - Obrada ogromnog broja zahteva
  - Geografska efikasnost
  - Ekstremna robusnost
- Ovi oblaci mogu biti drugi provajderi ili nezavisni regioni postojećih provajdera

Arhitektura rezervisanja resursa

- Arhitektura rezervisanja resursa služi da se izbegne problem gde tokom rada sistema dođe do konflikta oko resursa što može nastati ili zato što jednostavno nema više slobodnih sistemskih resursa (jednostavno nijedan fizički računar na raspolaganju nema dovoljno memorije, recimo) ili zato što se radi sa ne-objedinjenim resursima koji se ‘pozajmljuju’ za ekskluzivnu upotrebu i više entiteta se ‘bore’ oko ograničenog broja resursa ili neki entitet resurs ne vraća na vreme.
- Ova struktura se može rešiti kroz mehanizam alokacije unapred.

Arhitektura rezervisanja resursa

- Alokacija resursa unapred samo znači da se neki deo resursa ili neki poseban resurs za neko vreme ili za stalno dodeli nekim servisima koji se izvršavaju u okruženju na oblaku.
- Ovo je nešto što se javlja i kod oblak rešenja koje sami pravite i kada koristite usluge provajdera CC
- Razlika je u što je u prvom slučaju to više akt balansiranja ograničenim hardverskim resursima i lepo se kombinuje sa arhitekturama proboja
- U drugom slučaju, to je više ekonomska odluka: pitanje nije toliko da li će nestati resursa koliko je pitanje koliko ste voljni da platite za garantovan pristup dodatnim resursima i koliko vam takvih resursa treba.

Arhitektura dinamičke detekcije i prevencije otkaza

- Mi smo se bavili detekcijom i prevencijom otkaza ranije kada smo pričali o arhitekturi sa nultim vremenom otkaza
- Ali to je ograničen, specijalizovan slučaj: ovo može biti znatno kompleksnije
- Možemo da imamo mnogo više softiciran sistem koji detektuje različite forme otkaza i preuzima adekvatne metode da se nešto povodom toga odradi
- Ovo može biti sve od logovanja pa do automatskih operacija oporavka.

Arhitektura provizionisanja ‘golog metala’

- ‘Go metal’ je žargonski termin koji se koristi za fizičke instance kojima upravlja korisnik sa onoliko kontrole koliko tipično ima nad fizičkim računarom
- Ovo znači da umesto da se alocira virtuelna mašina se umesto alocira fizički računar
- Takav računar se može kontrolisati preko posebnog ROM-baziranog menadžment rešenja koje omogućava da se na takvu mašinu postavi bilo koji željeni operativni sistem.
- Što bi želeli ovako nešto?

Arhitektura brzog provizionisanja

- Veoma čest je scenario gde je neophodno izuzetno brzo reagovati na željene promene u sistemu
- Znate iz ličnog iskustva koliko zahteva instalacija i podešavanja računarskog sistema
- Kada bi vam dao prazan laptop i tražio da rekonstruišete okruženje u kome većinu vremena radite, koliko bi vam trebalo vremena?

Arhitektura brzog provizionisanja

- Ovo je neprihvatljivo za veliki broj računara stoga se proces automatizuje
- Šabloni
- Slike
- Skripte za konfiguraciju
- Sistemi za menadžment softverom/paketima
- **Kontejnerizacija**

Arhitektura za menadžment opterećenja skladišta podataka

- Ovo je visoko specijalizovana arhitektura koja radi balansiranje nad fizičkim/virtualnim instancama ali fokusirana na specifično mehanizme za skladištenje podataka.
- Ovde nije samo far fer raspodele fajlova što se tiče prostora nego i raspodele *zahteva.*
- To znači da se oportunistički kopiraju fajlovi ili delovi fajlova ili LUNovi ili štagod da je jedinica transfera na više ili manje mesta u zavisnosti od obrazca zahteva.
- Primer vrlo robusnog rešenja je IPFS.

Specijalizovane arhitekture

Arhitekture posebne namene

Arhitektura direktnog pristupa U/I

- Ovo je arhitektura koja omogućava da se direktno pristupa ulazno-izlaznim mehanizmima računara koji se koristi da bude domaćin virtuelne mašine.
- Performanse su bitne, ali mnogo bitnije je to što ovo omogućava da se prosledi specijalizovan hardver koji je teže virtualizovati.
- Za današnje mašine ovo je u velikom broju slučajeva GPU.

Arhitektura dinamičke normalizacije podataka

- Redudantnost podataka je jako moćna *kada nam treba.*
- Kada nam ne treba predstavlja čisto traćenje novca
- Stoga, ovakva arhitektura se može integrisati u ceo mehanizam skladištenja podataka da radi deduplikaciju.
- Deduplikacija može na nivou podataka u bazi, fajlova, ili čak samih blokova.
- Određeni sistemi skladištenja podataka (oni adresabilni sadržajem) rade implicitnu deduplikaciju kao recimo IPFS.

Arihtektura elastičnog kapaciteta mreže

- Ako možemo dinamički da alociramo računarske resurse da pokrijemo neplanirano veliki zahtev prema računarskim resursima, što ne isto sa mrežom?
- Ovo obuhvata alociranje mrežnih linkova kada zafali protoka
- Kompozitovano sa arhitekturama proboja može da se koristi kao lukava zaštita od (D)DOS napada.