Contents

[10.9] Fallimenti dell'Accoppiamento dei Sistemi

1. Definizione Operativa: Uno stato in cui sistemi strettamente accoppiati (dove un fallimento in uno colpisce immediatamente e inevitabilmente l'altro) mancano della resilienza per gestire i disturbi correlati alla sicurezza, portando a fallimenti a cascata.

2. Metrica Principale e Algoritmo:

- **Metrica:** Impatto del Fallimento dell'Accoppiamento (CFI). Formula: `CFI = (Downtime_Sistema_Dipende / Downtime_Sistema_Primario) * Peso_Criticità_Dati`.
- **Pseudocodice:**

  python

```
def calculate_cfi(primary_system_incident):
    # Ottieni il downtime del sistema primario
    primary_downtime = primary_system_incident.downtime_minutes

    # Ottieni il sistema accoppiato più colpito e il suo downtime dalle relazioni CMDB
    coupled_systems = get_directly_dependent_systems(primary_system_incident.affected_syst
    max_cfi = 0

    for system in coupled_systems:
        # Trova incidenti per questo sistema accoppiato durante l'incidente primario
        coupled_incident = find_related_incident(system, primary_system_incident.time)
        if coupled_incident:
            coupled_downtime = coupled_incident.downtime_minutes
            # Ottieni il peso della criticità da CMDB (ad es. 1 per dev, 5 per prod critic
            criticality = get_system_criticality(system)

            # Calcola CFI per questo sistema accoppiato
            if primary_downtime > 0:
                cfi = (coupled_downtime / primary_downtime) * criticality
            else:
                cfi = coupled_downtime * criticality
            if cfi > max_cfi:
                max_cfi = cfi
    return max_cfi
```

- **Soglia di Avviso:** `CFI > 1,0` (Il fallimento di un sistema causa un'interruzione *più* grave in un sistema dipendente, soprattutto se critico).

3. Fonti Dati Digitali (Input dell'Algoritmo):

- **CMDB:** Per le relazioni `depends_on` e i rating `system_criticality`.
- **Piattaforma di Gestione degli Incidenti:** Per il `system_downtime` per incidente.

4. Protocollo di Audit Umano-Umano: Durante le revisioni architettoniche, sfida specificamente i team sull'accoppiamento dei sistemi. "Se questo database va giù per 10 minuti, cosa si interrompe? Si interrompe immediatamente o c'è un buffer? Mostrami la modalità di guasto." Questo espone l'accoppiamento stretto.

5. Azioni di Mitigazione Consigliate:

- **Mitigazione Tecnica/Digitale:** Introduci accoppiamento lasco dove possibile: usa code (ad es. Kafka, SQS) per il buffer tra i sistemi, implementa ripetizioni con backoff esponenziale e progetta per il degrado elegante.
- **Mitigazione Umana/Organizzativa:** Addestra architetti e ingegneri sulle implicazioni di sicurezza e resilienza dell'accoppiamento stretto rispetto a quello lasco.
- **Mitigazione dei Processi:** Rendi obbligatorio che i test di business continuity e disaster recovery includano specificamente scenari in cui un sistema strettamente accoppiato è compromesso per testare la resilienza delle sue dipendenze.