스케줄링, 선점, 축출

📘 Scheduling, Preemption & Eviction (쿠버네티스 스케줄링 영역 전체 요약)

쿠버네티스가 *"새 파드를 어디에 둘 것인가?"*, 그리고 *"문제가 생겼을 때 어떤 파드를 내보낼 것인가?"*를 결정하는

스케줄링·우선순위·축출 관련 주요 기능들을 정리한 카테고리다.

아래 항목들은 서로 다른 목적을 가지고 있지만,

클러스터의 안정성·성능·자원 효율을 위해 하나의 큰 흐름을 이룬다.

⚙️ 스케줄러 성능 튜닝 (Scheduler Performance Tuning)

스케줄러의 계산 비용을 줄이고, 더 빠르고 효율적으로 "어디에 배치할지" 결정하도록 만드는 영역.

- ⏩ Scheduling Queue 최적화
- 🧱 Scoring 단계 튜닝
- 🧩 Filter/Score plugin 조절
- 📉 대규모 클러스터에서 병목 줄이기

📦 리소스 빈 패킹 (Bin Packing)

"리소스를 빈틈 없이 채워 넣는 전략".

- 🔥 **MostAllocated**

  → "거의 꽉 찬 노드"를 선호해 **리소스 분산을 방지**
- 📊 **RequestedToCapacityRatio**

→ CPU/MEM별 비율 기반 점수로 “얼마나 꽉 찼는가” 계산

- ⚖️ Custom weights

  → CPU:3, MEM:1 이런 식으로 리소스 중요도 조절 가능

- 🧠 목적: **리소스를 더 효율적으로 사용하면서 노드 수를 줄임**

🎚️ 파드 우선순위(Priority) & 선점(Preemption)

"누가 더 중요한가?"

우선순위에 따라 **낮은 파드를 밀어내고(high priority)** 자리를 확보해주는 기능.

- 🧭 **PriorityClass**

  → 숫자가 높을수록 더 중요한 파드

- 🗑️ **Preemption Logic**

  → 자리가 없으면 낮은 파드 축출 후 자리 확보

- 🏷️ **nominatedNodeName**

  → “여기 자리 비워둘게” 표시

- ⚠️ Preemption side-effects

  → 너무 높은 우선순위 설정 실수 시 대혼란 발생 가능

🧯 노드-압박 축출(Node-pressure Eviction)

스케줄러가 아닌 **kubelet**이 실행하는 "긴급 보호 메커니즘".

노드가 위험 상태일 때:

- 💾 메모리 부족
- 📦 디스크 부족
- 🔢 PID 고갈

kubelet이 직접 파드를 종료해 **노드를 죽지 않게 지키는 시스템**.

주요 요소

- 🚨 **Eviction Signals**
  - memory.available
  - nodefs.available
  - imagefs.inodesFree
  - pid.available
- 🚦 **Soft / Hard eviction thresholds**

  → Soft: 일정 시간 초과 시 축출

  → Hard: 즉시 축출(0초)
- 🔍 Node Conditions

  → MemoryPressure / DiskPressure 등
- 🧱 Eviction 순서
  1. 요청량 초과 BestEffort/Burstable
  2. 이후 우선순위(Priority) 낮은 것부터

> "서버가 죽기 직전이면, kubelet이 즉시 가방을 비워서라도 기계를 살린다".

🚨 API를 이용한 축출 (API-initiated Eviction)

운영자·컨트롤러가 **의도적으로 파드를 '정상 종료' 시키는 공식 API**.

- 📤 Eviction 객체 생성
- 🧹 terminationGracePeriodSeconds 준수
- 🛡️ PodDisruptionBudget(PDB) 준수
- 🧾 응답 코드
  - 200 OK → 축출 성공
  - 429 Too Many Requests → PDB 때문에 불가
  - 500 Internal Server Error → PDB 충돌 등 문제

📚 전체 관계 구조

스케줄러 (cluster-wide)
 ├─ 파드 우선순위 + 선점 → 자리 배치 판단
 └─ 빈 패킹 → 자원 효율 최적화

kubelet (node-local)
 ├─ 노드 압박 축출 → 노드 보호
 └─ OOM Killer 조정 → 컨테이너 생존 조절

사용자/컨트롤러
 └─ Eviction API → 정상 종료 기반의 정책적 축출