HealthCheck Kubernetes

1. **Healthcheck là gì?**
    - Đảm bảo app luôn "khỏe" khi chạy trên K8s.
    - Tự động restart hoặc ngừng nhận traffic khi app lỗi.

2. **Các loại Probe**
    - **Liveness Probe:** Kiểm tra app còn sống không.→ Fail thì K8s tự restart container.
    - **Readiness Probe:** Kiểm tra app đã sẵn sàng nhận traffic chưa (ví dụ: đã kết nối DB).→ Fail thì pod bị loại khỏi Service endpoint, không nhận traffic.
    - **Startup Probe:** Kiểm tra app đã khởi động xong chưa (dùng cho app khởi động lâu).→ Fail thì K8s restart container.

3. **Cách hoạt động**
    - K8s gọi các endpoint probe liên tục.
    - Nếu liveness fail → restart container.
    - Nếu readiness fail → ngừng nhận traffic.
    - Nếu startup fail → restart container khi khởi động.

4. **Các cơ chế probe**
    - **httpGet:** Gửi HTTP GET tới endpoint trong container.
    - **tcpSocket:** Kiểm tra kết nối TCP tới port của container.
    - **exec:** Chạy lệnh trong container, thành công nếu trả về mã 0.
    - **grpc:** Gọi gRPC health check (K8s 1.24+).

5. **Lưu ý khi dùng exec probe**

- Exec probe tạo process mới mỗi lần check, có thể gây tốn CPU nếu pod density cao hoặc period thấp.
- Nên dùng http/tcp probe nếu có thể.

6. **Probe outcome**
   - **Success:** Container "khỏe".
   - **Failure:** Container "lỗi", K8s sẽ xử lý theo loại probe.
   - **Unknown:** Không xác định, K8s sẽ kiểm tra lại.

7. **Cấu hình mẫu trong Deployment**
   Api:

```
[HttpGet("liveness")]
public IActionResult Liveness()
{
    return Ok("Alive");
}

[HttpGet("readiness")]
public IActionResult Readiness()
{
    // Kiểm tra kết nối database,...
    var connStr = _config.GetConnectionString("DefaultConnection");
    try
    {
        using var conn = new NpgsqlConnection(connStr);
        conn.Open();
        using var cmd = new NpgsqlCommand("SELECT 1", conn);
        cmd.ExecuteScalar();
    }
    catch (Exception ex)
    {
        return StatusCode(503, $"DB not ready: {ex.Message}");
    }
```

```
    return Ok("Ready");
}

[HttpGet("healthz")]
public IActionResult Healthz()
{
    return Ok("Healthy");
}

[HttpGet("kill")]
public IActionResult Kill()
{
    Environment.Exit(1);
    return StatusCode(500, "App killed");
}
```

HttpGet:

```
 livenessProbe:
   httpGet:
     path: /liveness
     port: 80
   initialDelaySeconds: 5
   periodSeconds: 10
   failureThreshold: 1
 readinessProbe:
   httpGet:
     path: /readiness
     port: 80
   initialDelaySeconds: 5
   periodSeconds: 10
   failureThreshold: 1
   successThreshold: 3
 startupProbe:
   httpGet:
     path: /healthz
     port: 80
```

```
  failureThreshold: 30
  periodSeconds: 10
```

- **livenessProbe:**
  - **initialDelaySeconds:** Đợi 5 giây sau khi container start mới bắt đầu check.
  - **periodSeconds:** Mỗi 10 giây sẽ check lại một lần.
  - **failureThreshold:** Nếu fail 1 lần liên tiếp thì coi như container "chết" và sẽ bị restart.
- **readinessProbe:**
  - **initialDelaySeconds:** Đợi 5 giây sau khi container start mới bắt đầu check.
  - **periodSeconds:** Mỗi 10 giây sẽ check lại một lần.
  - **failureThreshold:** Nếu fail 1 lần liên tiếp thì pod bị loại khỏi Service endpoint (ngừng nhận traffic).
  - **successThreshold:** Phải liên tục thành công 3 lần thì mới được đánh dấu là "Ready" (bắt đầu nhận traffic).
- **startupProbe:**
  - **failureThreshold:** Nếu fail liên tục 30 lần thì container bị restart (tối đa 30 × 10 = 300 giây để khởi động).
  - **periodSeconds:** Mỗi 10 giây sẽ check lại một lần.

TcpSocket:

```
livenessProbe:
  tcpSocket:
    port: 8042
  initialDelaySeconds: 5
  periodSeconds: 10
  failureThreshold: 3
```

> Dùng cho các app lắng nghe TCP

Đoạn cấu hình này là **livenessProbe** dùng **tcpSocket** để kiểm tra container Orthanc có còn sống không.

- **tcpSocket:** K8s sẽ thử kết nối TCP tới port 8042 trong container.
- Nếu kết nối thành công → container "khỏe".
- Nếu kết nối thất bại liên tiếp 3 lần (**failureThreshold: 3**) → K8s sẽ tự động restart container Orthanc.

Exec:

```
livenessProbe:
  exec:
    command:
      - redis-cli
      - -a
      - redis123
      - ping
  initialDelaySeconds: 5
  periodSeconds: 10
  failureThreshold: 3
```

> Dùng cho các app có CLI kiểm tra trạng thái / custom script

Đoạn cấu hình này là **livenessProbe** dùng kiểu **exec** để kiểm tra container Redis có còn sống không.

- **exec:** K8s sẽ chạy lệnh `redis-cli -a redis123 ping` bên trong container.
- Nếu lệnh trả về `"PONG"` (exit code 0) → container "khỏe".
- Nếu lệnh trả về lỗi hoặc không phản hồi liên tiếp 3 lần (**failureThreshold: 3**) → K8s sẽ tự động restart container Redis.

References

- Kubernetes: **Configure Liveness, Readiness and Startup Probes**
- Viblo: **Cấu hình Healthcheck trên Kubernetes**
- Kubernetes: **Pod Lifecycle**