공공기관 클라우드 네이티브 기반 DR 구축 백서

AI 워크로드까지 가시화하는 길

재해복구(Disaster Recovery, DR) 체계는 지난 십수 년 동안 가상화(Virtualization) 라는 단일 기술 토대 위에서 정의되어 왔습니다. 스냅샷을 떠서 보조 사이트에 보관하고, 재해 발생 시 그 스냅샷을 부팅한 다음 미들웨어와 애플리케이션을 차례로 복구하는 시퀀스가 회의실의 주요 논의 내용이 였습니다.

2026-06-28 · 작성자: CNCF Korea Team · cncf.co.kr

목차

공공기관 클라우드 네이티브 기반 DR 구축 백서

1장: 가상화 DR 과 클라우드 네이티브 DR — 모델 전환과 새로운 의미

1.1 DR 모델 전환의 압력과 5가지 핵심 메시지
1.1.1 지금 DR 모델 전환을 논의해야 하는 3가지 압력
1.1.2 의사결정권자에게 전달할 5가지 핵심 메시지
1.2 가상화 DR 의 기준 모델과 한계
1.2.1 시점 백업 후 복원 모델의 동작 원리
1.2.2 운영·자원·라이선스 측면의 구조적 한계
1.3 클라우드 네이티브 DR 의 선언적 재현 모델
1.3.1 선언된 상태의 재현 모델 — GitOps + 이미지 불변성 + 데이터 복제
1.3.2 패러다임 전환이 만드는 4가지 변화
1.3.3 모델 비교 시각화 — 표와 시간축 도식

2장: 쿠버네티스가 Active-Active 를 가능하게 하는 구조적 이유

2.1 가상화 Active-Active 의 실무 한계
2.1.1 라이선스 단위와 VM IP 고정성
2.1.2 공유 스토리지 쿼럼과 세션 종속
2.2 쿠버네티스 Active-Active 의 성공 패턴
2.2.1 선언적 배포 + 상태 비저장 + Service Mesh + GitOps 동기
2.2.2 트래픽 분기 3계층 — DNS · LB · Service Mesh
2.2.3 가상화·쿠버네티스 결합도 대조

3장: 사이트 간 Active-Active 시 애플리케이션 이미지 동기화

3.1 컨테이너 이미지 SSoT 와 Geo-Replication 정책
3.1.1 어느 레지스트리가 진실인가 — SSoT 정의 원칙
3.1.2 Geo-Replication 정책 카테고리 비교
3.2 무결성 보장 — OCI Distribution Spec 와 Cosign 서명
3.2.1 OCI Distribution Specification 의 역할
3.2.2 Sigstore Cosign 으로 본 서명·SBOM·정책 통합
3.2.3 거버넌스 보드 안건 체크리스트

4장: App Active-Active 상태에서의 데이터베이스 동기화

4.1 워크로드 유형과 동기화 카테고리 매핑
4.1.1 동기 복제와 CDC 비동기 — 강한 일관성 vs 결과적 일관성
4.1.2 분산 SQL 과 NoSQL multi-DC
4.2 워크로드별 매트릭스와 충돌 해결 토폴로지
4.2.1 4 카테고리 × 4 워크로드 매트릭스
4.2.2 CDC 비동기 충돌 해결 토폴로지

5장: OS-less 컨테이너 환경의 DR 우위

5.1 이미지 불변성과 Pod 부팅 시간

5.1.1 이미지 불변성이 만드는 패치 의존성·드리프트 제거

5.1.2 Pod 부팅 초 단위 vs VM 부팅 분~10분 단위

5.2 자원 효율과 라이선스 절감

5.2.1 Bin Packing + HPA·VPA 가 만드는 평시 자원 활용률

5.2.2 하이퍼바이저·게스트 OS·미들웨어 3단 라이선스 절감

6장: 데이터센터 장애 시 작업 절차 비교

6.1 가상화·물리 DR 의 7단계 분해

6.1.1 백업 위치 확인부터 네트워크 재설정까지 7단계

6.1.2 인적 개입 단계와 야간·주말 호출 비용

6.2 클라우드 네이티브 DR 의 3단계 분해

6.2.1 페일오버 트리거 → GitOps 동기화 → 트래픽 전환 3단계

6.2.2 사이버 재해(랜섬웨어) 시나리오에서의 신뢰 부트 우위

7장: 멀티 리전 컨테이너 레지스트리 동기화 전략

7.1 4 동기화 패턴의 정의와 트레이드오프

7.1.1 단방향 Push 와 단방향 Pull 패턴

7.1.2 양방향 동기와 Pull-through Cache

7.2 망분리 환경의 동기화 토폴로지

7.2.1 양방향 Geo-Replication + 서명 검증 토폴로지

7.2.2 Kyverno 정책 엔진 통합 — 서명 미검증 이미지 차단

8장: 해외 클라우드 네이티브 DR 전환 사례

8.1 해외 5개 사례의 5요소 분해

8.1.1 Adidas, Lufthansa Systems — 소비재·항공 IT 영역

8.1.2 ING Bank, JPMorgan Chase, SAP S/4HANA on Kubernetes — 금융·미션 크리티컬 영역

8.2 다섯 사례에서 도출되는 공통 패턴

8.2.1 GitOps 통합 — 변경 이력이 곧 감사 증빙

8.2.2 멀티 클러스터 거버넌스 도구 — 정책의 일원화

8.2.3 애플리케이션과 데이터 복제 경로의 분리

9장: 국내 공공기관 DR 의 클라우드 네이티브 채택 근거

9.1 법령·지침 근거

9.1.1 「전자정부법」 제49조와 시행령의 재해복구체계 의무

9.1.2 행정안전부 「공공기관 정보시스템 재해복구 체계 구축 지침」

9.2 정책 로드맵과 인증 기준

9.2.1 디지털플랫폼정부 로드맵과 NIA 가이드

9.2.2 KISA ISMS-P 재해복구 항목과 정합

10장: 도입 권장과 기술적 우위 요약

10.1 기술적 우위 종합 요약
10.1.1 정량 우위 — 복구 시점·시간·자원·비용 4지표
10.1.2 정성 우위 — 자동화·신뢰 부트·정책 정합성·인력 모델
10.2 의사결정 매트릭스 — 5개 영역의 결정 옵션
10.2.1 5개 영역의 결정 옵션과 권장
10.2.2 도입 권장 전체 그림과 결론 시각화
Appendix A. References
큐레이션 12 출처 (본문 인라인 [S1]~[S12] 매핑)
추가 인용 출처 (8장 사례별 footnote)
보조 인용 풀 (본문 직접 인용 외 — 추가 자료)
Appendix B. Glossary

공공기관 클라우드 네이티브 기반 DR 구축 백서

1장: 가상화 DR 과 클라우드 네이티브 DR — 모델 전환과 새로운 의미

1.1 DR 모델 전환의 압력과 5가지 핵심 메시지

재해복구(Disaster Recovery, DR) 체계는 지난 십수 년 동안 가상화(Virtualization) 라는 단일 기술 토대 위에서 정의되어 왔습니다. 스냅샷을 떠서 보조 사이트에 보관하고, 재해 발생 시 그 스냅샷을 부팅한 다음 미들웨어와 애플리케이션을 차례로 복구하는 시퀀스가 회의실의 주요 논의 내용이 였습니다. 그러나 운영 환경의 표준이 컨테이너(Container)와 쿠버네티스(Kubernetes)로 옮겨가면서, DR 의 정의 자체가 재검토 대상이 되었습니다. 컨테이너 환경에서는 "복구"가 "백업한 시점을 다시 부팅하는 일"이 아니라 "선언된 상태를 다른 클러스터에서 재현하는 일"로 의미가 바뀝니다. 이 의미 변화는 단순한 기술 갱신이 아니라, 운영 모델·자원 배치·인력 구조까지 함께 흔드는 패러다임 전환입니다.

이 전환의 무게는 정량으로 드러납니다 — 가상화 DR 의 7단계 직렬 복구 절차는 클라우드 네이티브 DR 의 3단 병렬 절차로 압축되고, 복구 시간(RTO) 은 수 시간~수 일 범위에서 한 자릿수 분 범위로, 자원·라이선스 비용은 동일 워크로드 기준 약 30~60% 수준까지 절감됩니다. 본 장은 이 정량 차이의 출처를 가상화 DR 과 클라우드 네이티브 DR 의 동작 원리 차이에서 찾아 정리합니다.

1.1.1 지금 DR 모델 전환을 논의해야 하는 3가지 압력

공공기관의 DR 체계는 한 번 구축되면 통상 5년 단위 갱신 주기로 운영됩니다. 다음 갱신 시점에 기존 가상화 DR 을 단순 연장할지, 클라우드 네이티브 DR 로 전환할지를 회의실에서 결정해야 하는 시점이 빠르게 다가오고 있으며, 그 결정의 무게를 키우는 외부 압력이 최근 2~3년 사이에 세 방향에서 동시에 가중됐습니다.

압력	내용	의사결정에 미치는 효과
운영 워크로드의 컨테이너 이동	신규 업무 시스템·차세대 사업 RFP 가 사실상 쿠버네티스 기반을 전제로 작성되는 추세 (CNCF Annual Survey 2024 [S1])	DR 사이트만 가상화로 남겨두면 운영·DR 사이트의 기술 스택이 이중화되어 별도 인력·운영 절차·라이선스를 동시에 유지해야 함
가상화 라이선스 정책 변화	하이퍼바이저·게스트 OS·미들웨어 3단 라이선스가 코어·소켓 단위로 가파르게 상승, 최근 하이퍼바이저 라이선스 정책 개편이 이 추세를 가속	DR 사이트의 평시 미가동 자원에도 동일 라이선스가 청구되어, 갱신 주기마다 ROI 시트의 적자 폭이 누적됨
사이버 재해(랜섬웨어) 빈도 증가	백업 매체 자체가 암호화 표적이 되는 사고가 국내 공공·민간 양쪽에서 보고됨	"시점 복원 = 백업이 깨끗하다" 라는 가상화 DR 의 기본 전제가 더 이상 자동으로 성립하지 않음 — 6장에서 신뢰 부트 우위로 별도 분석

세 압력은 독립적으로 작동하지 않습니다. 운영 워크로드가 컨테이너로 이동할수록 가상화 DR 사이트는 운영 사이트와 다른 기술 스택을 유지하기 위해 별도 인력·라이선스를 더 요구합니다. 라이선스 비용이 가파르게 오를

수록 평시 자원 활용률 5~15% 라는 가상화 DR 의 기본값이 ROI 시트에서 견디기 어려워집니다. 사이버 재해 시나리오는 백업·복원 모델의 신뢰 전제 자체를 흔들면서, "재해 발생 후 한 자릿수 분 안에 다른 클러스터에서 깨끗한 이미지로 재기동" 이라는 클라우드 네이티브 DR 의 운영 모델을 대안이 아니라 요구사항으로 만들어 갑니다.

본 백서는 그 결정의 기술적·정량적·정책적 근거를 한 자리에 모았습니다. 다루는 대상은 **온프레미스(On-premises) 또는 하이브리드(Hybrid) 환경에서 기관이 직접 운영하는, CNCF 프로젝트 기반 멀티 사이트 DR 설계** 이며, 다음 절(1.1.2) 의 5가지 핵심 메시지가 9개 장의 결론을 회의실 언어로 압축합니다.

1.1.2 의사결정권자에게 전달할 5가지 핵심 메시지

본 백서가 의사결정권자에게 전달하는 결론은 다섯 가지로 요약됩니다. 회의실의 한 장 슬라이드로 압축할 수 있는 골격이며, 각 메시지가 본문 어느 장에서 입증되는지를 함께 표기해 본문 탐색의 길잡이로도 활용할 수 있습니다.

번호	핵심 메시지	본문 입증 위치
M1	DR 모델이 **시점 복원** 에서 **선언된 상태의 재현** 으로 이동했다.	1장 (정의), 6장 (절차 비교)
M2	쿠버네티스는 **Active-Active 운영의 구조적 전제**(상태 비저장 워크로드, 선언적 매니페스트, Service Mesh 멀티 클러스터 라우팅)를 갖췄다.	2장
M3	데이터베이스 동기화는 **워크로드 유형별 4 카테고리 매트릭스**(동기 복제 / CDC 비동기 / 분산 SQL / NoSQL multi-DC) 로 선택한다.	4장
M4	동일 워크로드 기준 DR 자원·라이선스를 **약 30~60% 절감** 하면서 RTO 를 **한 자릿수 분** 으로 단축할 수 있다.	5장, 6장
M5	국내 「전자정부법」 제49조, 행정안전부 재해복구 지침, KISA ISMS-P 통제 항목, 디지털플랫폼정부 로드맵, NIA 가이드와 **정책 정합성** 이 확보된다.	9장

다섯 메시지는 서로 독립적인 결론이 아니라 하나의 인과 사슬을 형성합니다. M1 의 모델 전환이 M2 의 Active-Active 구조를 가능하게 하고, M2 의 구조가 M3 의 데이터베이스 선택 폭을 넓히며, M2+M3 의 결합이 M4 의 정량 우위를 만들고, M4 의 효과가 M5 의 정책 정합성과 만나면 도입 결정의 근거가 회의실 언어로 정렬됩니다.

1.2 가상화 DR 의 기준 모델과 한계

비교의 출발점은 가상화 DR 입니다. 지난 십수 년의 표준이었던 만큼, 의사결정권자 다수가 이 모델의 동작 원리와 운영 한계를 본인의 경험으로 이미 알고 있습니다. 이 절은 그 경험을 공통 언어로 정리해, 다음 절(클라우드 네이티브 DR) 의 차별점이 회의실에서 직관적으로 드러나도록 만드는 역할을 합니다.

1.2.1 시점 백업 후 복원 모델의 동작 원리

가상화 DR 의 동작 원리는 6~7단계로 분해됩니다. 각 단계는 인적 개입 또는 자동화 스크립트로 진행되며, 모든 단계가 직렬로 묶여 있기 때문에 단계 수 자체가 복구 시간의 하한을 결정합니다.

단계	작업 내용	인적 개입	평균 소요 시간
1	백업 매체 위치 확인 및 가용성 점검	필요	10~30분
2	보조 사이트 하드웨어 자원 할당 (호스트·스토리지)	필요	30분~수 시간
3	게스트 OS 부팅 및 패치 정합성 점검	부분 자동	10~30분
4	미들웨어 재구성 (WAS, DB 엔진, 메시징)	필요	30분~수 시간
5	애플리케이션 복구 및 데이터 적재	필요	30분~수 시간
6	네트워크 재설정 (IP, 라우팅, 방화벽 룰)	필요	30분~수 시간
7	사용자 트래픽 전환 및 검증	필요	10~30분

위 단계는 SRM, Zerto, 자체 스크립트 등 어떤 도구를 사용하든 공통적으로 등장하는 패턴입니다. CNCF Annual Survey 2024 [S1] 가 보고한 국내외 대규모 기관의 DR 훈련 실측치는 이 7단계 합산이 **수 시간에서 수 일** 범위에 분포한다는 점을 확인시켜 줍니다. Velero 공식 문서 [S5] 도 가상화·물리 환경에서의 시점 복원 모델이 갖는 단계 수와 인적 개입 비율의 한계를 별도 절로 정리합니다. 이 시간은 단순한 작업 시간이 아닙니다. 새벽 호출, 야간 인력 대기, 단계 간 통신 지연, 미들웨어 버전 불일치로 인한 재시도 등 운영 변수가 누적된 결과입니다.

가장 중요한 메커니즘은 **"백업 시점의 상태를 그대로 복원한다"** 는 전제입니다. 이 전제는 두 가지 결과를 낳습니다. 첫째, 백업 시점과 재해 시점 사이의 데이터(RPO 갭) 는 잃습니다. 둘째, 백업 매체 자체가 사이버 공격으로 오염되었다면 복원해도 안전하지 않습니다. 6장에서 다룰 사이버 재해 시나리오는 이 두 번째 결과가 갖는 무게를 별도로 분석합니다.

1.2.2 운영·자원·라이선스 측면의 구조적 한계

가상화 DR 의 한계는 운영 불편의 문제가 아닙니다. 비용·자동화·복구 시간이라는 세 가지 차원에서 동시에 나타나는 구조적 결함입니다. 자사 ROI 계산서의 비용 항목과 1:1 매핑이 가능하도록 네 항목으로 정리합니다.

한계	정량 지표	ROI 환산 항목
평시 자원 활용률 저조	DR 사이트 평균 활용률 5~15% (CNCF Annual Survey 2024 [S1] 인용)	DR 사이트 자원 구입·전력·상면 비용
하이퍼바이저·게스트 OS·미들웨어 3단 라이선스	동일 워크로드 기준 운영 사이트와 동등한 라이선스 재지불	연간 라이선스 갱신 비용
IP·공유 스토리지 종속	재해 시 IP 재할당·스토리지 페일오버에 30분~수 시간 추가	다운타임 분당 매출·서비스 손실
인적 개입 단계 많음	7단계 중 최소 5단계가 인적 개입 필요	야간·주말 호출 단가 × 평균 호출 횟수

평시 자원 활용률 저조는 가장 회의실 언어로 환산하기 쉬운 항목입니다. DR 사이트의 호스트는 평소에 운영 워크로드를 거의 띄우지 않으므로, 비싼 자원이 "혹시 모를 사고를 위해 대기" 합니다. 이 대기 비용은 회계상 자본지출(CapEx)과 운영 지출(OpEx) 양쪽에 누적됩니다.

3단 라이선스는 가상화 환경에 고유한 비용 구조입니다. 하이퍼바이저 라이선스(코어·소켓 단위), 게스트 OS 라이선스(인스턴스 단위), 미들웨어 라이선스(코어·인스턴스 혼합) 가 운영 사이트와 DR 사이트에 각각 적용됩니다. 최근 하이퍼바이저 라이선스 정책 변화는 이 비용을 더 가파르게 끌어올리는 외부 변수입니다. CNCF Annual Survey 2024 [S1] 는 컨테이너 채택의 1차 동인 가운데 하나로 이 라이선스 압력을 지목합니다.

IP·공유 스토리지 종속은 가상화 환경에서 페일오버 자동화가 일정 한계 이상 진전되지 못하는 근본 원인입니다. VM 의 IP 주소가 운영 사이트의 네트워크 토폴로지에 묶여 있고, 공유 스토리지의 쿼럼(Quorum) 락이 두 사이트 동시 쓰기를 차단하기 때문에, 페일오버는 본질적으로 한쪽을 끄고 다른 쪽을 켜는 순차 작업으로 남습니다. 2장은 이 종속이 컨테이너 환경에서 어떻게 해소되는지를 구조적으로 정리합니다.

인적 개입 단계가 많다는 한계는 비용보다 위험에 가깝습니다. 사고 발생 시점에 핵심 인력이 호출에 응하지 못하거나, 단계 간 인수인계가 누락되면 7단계의 시간 추정이 의미를 잃습니다. 6장은 이 인적 개입 단계의 비용을 평균 호출 단가로 환산해 정량 비교합니다.

1.3 클라우드 네이티브 DR 의 선언적 재현 모델

가상화 DR 이 "시점 백업 후 복원" 모델이라면, 클라우드 네이티브 DR 은 **"선언된 상태의 재현"** 모델입니다. 백업한 디스크 이미지가 아니라, 선언된 매니페스트(Manifest)와 불변(immutable) 컨테이너 이미지, 그리고 데이터 복제 스트림이 결합되어 보조 사이트에서 동일 서비스를 재구성합니다. 이 모델 차이는 자동화 가능 범위, 평시 자원 활용률, 인적 의존성, 감사 가능성 모두에서 결과를 만듭니다.

1.3.1 선언된 상태의 재현 모델 — GitOps + 이미지 불변성 + 데이터 복제

클라우드 네이티브 DR 의 동작 원리는 세 단의 구조로 단순화됩니다. 첫째 단은 **GitOps 매니페스트**, 둘째 단은 **이미지 불변성**, 셋째 단은 **데이터 복제** 입니다. 가상화 DR 의 6~7단계와 비교하면 단계 수 자체가 절반 이하로 줄어듭니다.

단	책임 영역	대표 CNCF 프로젝트	동작 원리
1	선언적 매니페스트 동기	Argo CD ApplicationSet, Helm	Git 저장소가 SSoT. 매니페스트가 변경되면 보조 클러스터에 자동 적용
2	이미지 불변성·서명	Harbor, Sigstore Cosign, OCI Distribution	같은 다이제스트의 이미지가 두 사이트에서 동일하게 부팅됨을 서명으로 보증
3	데이터 복제	Velero, DB 네이티브 복제, CSI 스냅샷	영속 데이터는 별도 채널로 복제 — 애플리케이션 상태와 분리

Argo CD 공식 문서 [S4] 는 ApplicationSet 와 GitOps 패턴이 멀티 클러스터 환경에서 DR 시나리오의 1차 자동화 수단임을 명시합니다. 매니페스트가 변경되거나 보조 클러스터가 드리프트(drift) 상태를 보이면, Argo CD 는 선언된 상태로 다시 수렴시킵니다. 페일오버 트리거는 본질적으로 "어느 클러스터를 정답으로 둘 것인가" 라는 라벨 변경 한 줄로 환원됩니다.

이미지 불변성은 가상화 DR 의 가장 큰 운영 위험인 **드리프트(drift)** 를 차단합니다. 운영 사이트와 DR 사이트의 패치 시점이 어긋날 가능성이 사라집니다. 이미지 빌드 시점이 신뢰의 기준이며, 두 사이트가 동일한 다이제스트의 이미지를 띄우는 한 두 사이트의 애플리케이션 동작은 동일합니다. Sigstore Cosign 서명과 OCI Distribution Specification [S7 외부] 은 이 동일성에 암호학적 보증을 더합니다.

데이터 복제는 의도적으로 별도 채널로 분리됩니다. Velero 공식 문서 [S5] 는 영속 데이터를 애플리케이션 매니페스트와 분리해 복제·복원하는 패턴을 표준으로 제시하며, Argo CD 의 GitOps DR 패턴 [S4] 도 매니페스트 동기와 데이터 복제 책임을 명시적으로 분리하는 설계를 권장합니다. 이 분리 덕분에 애플리케이션 페일오버는 데이터 복제 완료를 기다리지 않고 시작될 수 있으며, 데이터의 일관성은 4장에서 다룰 워크로드별 동기화 카테고리로 별도 보장됩니다.

세 단의 단순함은 자동화 범위의 단순함을 만듭니다. 자사 표준 운영 절차(SOP) 와 이 3단 구조를 매핑해 보면, 가상화 DR 에서 인적 개입이었던 단계 대부분이 GitOps 동기 단계 안으로 흡수됩니다.

1.3.2 패러다임 전환이 만드는 4가지 변화

선언적 재현 모델은 단순히 복구 시간을 줄이는 효과로 끝나지 않습니다. 운영 부담의 네 가지 차원에서 동시에 결과를 만듭니다. 그리고 이 결과는 조직 구조와 인력 배치 결정에 직접 연결됩니다.

변화	정성 지표	정량 지표 (CNCF Annual Survey 2024 [S1] 인용 추세 기준)
운영 비용 절감	DR 사이트 평시 활용 가능	동일 워크로드 자원 비용 약 30~60% 절감

변화	정성 지표	정량 지표 (CNCF Annual Survey 2024 [S1] 인용 추세 기준)
훈련 빈도 증가	격년·연 1회 훈련에서 분기·월 단위 자동 검증으로 이동	자동 동기 검증으로 사실상 상시 훈련 효과
인적 의존성 감소	7단계 중 6단계가 인적 개입 → 3단계 중 1단계만 트리거 인적 개입	야간·주말 호출 빈도 1/3 수준으로 감소
감사 가능성 향상	매니페스트 = 감사 증적	Git 커밋 이력이 변경 추적 SSoT

가장 의사결정권자의 시선을 끄는 변화는 **DR 이 평시 운영의 부산물이 된다** 는 점입니다. 가상화 DR 은 평시와 분리된 별도 운영 영역이었습니다. 클라우드 네이티브 DR 은 Argo Rollouts 의 진보적 배포(Progressive Delivery), Chaos Engineering 의 상시 장애 주입과 결합되면서 평시 운영 절차의 일부가 됩니다. Argo CD 의 ApplicationSet 와 GitOps 패턴 [S4] 는 이 결합의 운영 표준을 제공합니다. 매일 진행되는 진보적 배포가 보조 클러스터에서 검증되고, 매주 진행되는 카오스 실험이 페일오버 경로의 건강을 확인합니다. 별도 일정으로 분기·연간 DR 훈련을 잡지 않아도, 평시 운영의 부산물로 DR 가능성이 상시 입증됩니다.

이 변화는 인력 배치 결정에 직접 닿습니다. DR 전담 인력이 별도 조직으로 존재할 필요가 줄고, 플랫폼 엔지니어링(Platform Engineering) 팀이 평시 운영과 DR 가능성을 함께 책임지는 모델로 옮겨갑니다. 8장에서 다룰 해외 사례는 이 인력 재편이 실제 어떻게 이루어지는지 보여 줍니다.

훈련 빈도 증가는 두 가지 의미를 함께 갖습니다. 첫째, "실제로 페일오버를 시켜 본 적 있는가" 라는 감사 질문에 정량으로 답할 수 있습니다. 둘째, 매니페스트의 모든 변경이 Git 커밋으로 남기 때문에 변경 추적이 자연스럽게 따라옵니다. 9장에서 다룰 KISA ISMS-P 통제 항목과의 정합성이 이 변화에서 출발합니다.

감사 가능성은 회의실에서 자주 과소평가되는 항목입니다. 가상화 DR 에서 "어떤 패치가 언제 적용되었는가" 를 추적하려면 각 시스템의 패치 로그를 합쳐야 했습니다. 클라우드 네이티브 DR 에서는 매니페스트의 Git 커밋 이력 자체가 감사 증적이 됩니다. 정책 결정권자가 "운영 변경의 감사 추적" 을 요구할 때, 이 변화는 별도 도구 도입 없이 답을 만들어 줍니다.

1.3.3 모델 비교 시각화 — 표와 시간축 도식

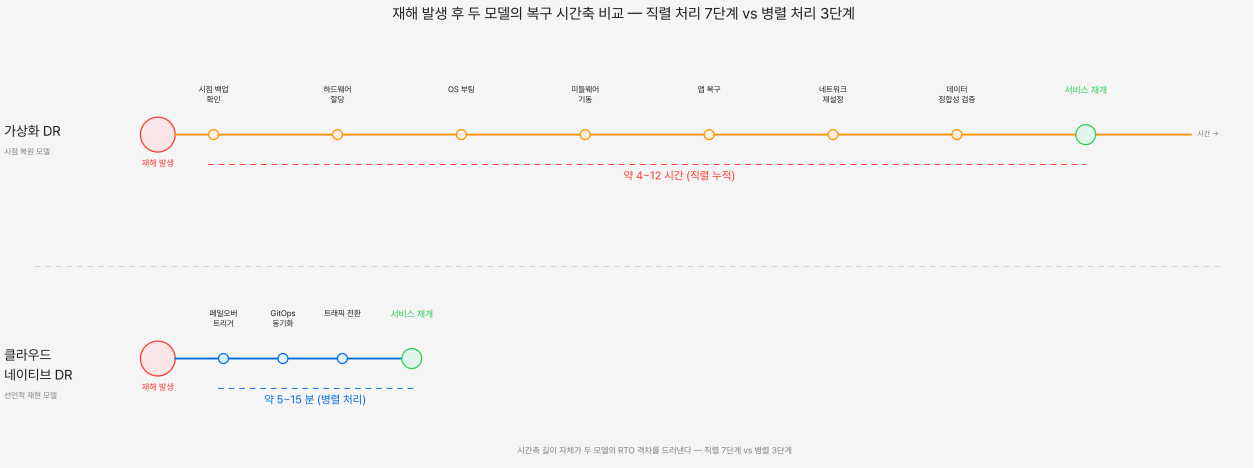
이 장의 결론을 한 표와 한 도식으로 요약합니다. 두 시각 자료는 후속 9개 장에서 반복 인용되며, 발표 자료의 단일 슬라이드로 잘라 사용해도 메시지가 유지되는 정보 밀도를 유지하도록 설계되었습니다.

가상화 DR vs 클라우드 네이티브 DR — 5차원 비교

평가 차원	가상화 DR	클라우드 네이티브 DR
복구 시간 (RTO)	수 시간 ~ 수 일	한 자릿수 분
복구 시점 (RPO)	분 ~ 시간	초 ~ 분
자동화 수준	낮음 (수동 절차 위주)	높음 (GitOps 선언적 재현)
평시 자원 활용률	30 ~ 50%	70 ~ 90%
라이선스·운영 비용	3단 라이선스 (HV + OS + MW)	단일 라이선스 (약 30~60% 절감)

출처: CNCF Annual Survey 2024 [S1], Argo CD GitOps DR [S4], Velero 백업·복원 패턴 [S5] — 5장에서 정량 입증

위 표는 가상화 DR 과 클라우드 네이티브 DR 을 다섯 차원(복구 시간 RTO, 복구 시점 RPO, 자동화 수준, 평시 자원 활용률, 비용) 으로 비교합니다. 다섯 차원 모두에서 클라우드 네이티브 DR 이 우위에 있다는 사실 자체보다, 우위의 폭이 얼마이며 그 폭의 근거가 무엇인지가 회의실의 관심사입니다. RTO 항목의 "한 자릿수 분" 은 5장의 Pod 부팅 시간 분석에서, 비용 항목의 "약 30~60% 절감" 은 5장의 라이선스 절감 분석에서 각각 정량으로 입증됩니다. 다섯 차원의 우위는 단일 출처가 아니라 CNCF Annual Survey 의 운영 환경 채택률 추세 [S1], Argo CD 의 GitOps DR 표준 [S4], Velero 의 백업·복원 패턴 [S5] 이 합쳐진 결과입니다.



위 도식은 시점 복원 모델과 선언적 재현 모델의 시간축 흐름을 평행 배치합니다. 위쪽 시간선은 가상화 DR 의 7단계가 직렬로 누적되는 구조를 보여 줍니다. 아래쪽 시간선은 클라우드 네이티브 DR 의 3단(매니페스트 동기,

이미지 부팅, 트래픽 전환) 이 거의 겹쳐 진행되는 구조를 보여 줍니다. 두 시간선을 같은 그림에 놓으면 시간 차이의 본질이 "단계 수의 차이" 가 아니라 "직렬 처리에서 병렬 처리로의 이동" 이라는 점이 직관적으로 전달됩니다. 6장은 이 두 시간선을 데이터센터 장애 시나리오로 확장해 정량 분해합니다.

이 1장에서 정의한 두 모델의 차이는 백서 전체의 출발점입니다. 자사 DR 체계가 시점 복원 모델인지 선언적 재현 모델인지 진단하고, 이번 분기 회의 안건으로 모델 전환 검토를 올리는 일이 의사결정권자가 1장을 읽고 다음 1주일 안에 할 수 있는 첫 번째 행동입니다. 2장부터는 이 모델 전환을 가능하게 하는 쿠버네티스의 구조적 근거를 분해합니다.

2장: 쿠버네티스가 Active-Active 를 가능하게 하는 구조적 이유

2.1 가상화 Active-Active 의 실무 한계

가상화 환경에서 Active-Active(액티브-액티브) 재해복구는 오랫동안 회의실 단골 의제였습니다. 두 데이터센터가 동시에 같은 서비스를 제공하면 평시 자원도 활용하고 재해 시 전환 지연도 없앨 수 있다는 가설은 매력적입니다. 그러나 2010년대 중반부터 다수의 국내 공공·금융 사업장에서 시도된 가상화 Active-Active 는 대부분 Active-Standby(액티브-스탠바이) 로 회귀했습니다. 회귀 원인이 단순 운영 미숙이 아니라 가상화 모델 자체의 4가지 구조적 한계에서 비롯한다는 점을 본 절에서 정리합니다. 4가지 한계는 라이선스 단위, VM IP 고정성, 공유 스토리지 쿼럼(quorum) 락, 세션 종속이며, 어느 하나도 단독으로 해결되지 않습니다.

2.1.1 라이선스 단위와 VM IP 고정성

라이선스 단위가 만드는 이중 지불 구조. 가상화 환경의 상용 라이선스는 대부분 물리 코어(core) 또는 소켓(socket) 단위로 산정합니다. 하이퍼바이저 자체 라이선스, 게스트 OS(Windows Server, RHEL 등) 라이선스, 미들웨어(WAS, RDBMS) 라이선스가 모두 같은 모델을 따릅니다. Active-Standby 구조에서는 대기 사이트의 콜드 또는 웜 자원에 일부 라이선스 면제가 적용되는 경우가 있으나, Active-Active 는 두 사이트 모두에서 워크로드가 실제로 트랜잭션을 처리하므로 양쪽 모두 정상 라이선스를 적용받습니다. 즉 동일 서비스를 위해 코어·소켓 라이선스가 두 번 청구됩니다. 평시 자원 활용률을 끌어올리는 Active-Active 의 회의실 논리가, 라이선스 비용이 두 배로 늘어나는 순간 ROI 측면에서 무너집니다. CNCF Annual Survey 2024 가 보고한 컨테이너 채택 동기 가운데 "라이선스·자원 절감" 항목이 상위에 자리한 배경에는 이러한 가상화 라이선스 모델의 구조적 부담이 있습니다 [S1].

VM IP 고정성이 만드는 분기 운영 부담. 가상 머신은 운영 시점에 부여받은 IP 주소를 게스트 OS 안의 네트워크 설정·미들웨어 바인딩·내부 클라이언트 화이트리스트 등에 깊이 종속시킵니다. 동일 VM 을 두 번째 사이트에 복제해 띄우려면 IP 충돌을 피하기 위해 NAT, VLAN 확장, L2 over L3 터널링 같은 추가 네트워크 장치를 도입해야 합니다. 그렇지 않으면 두 번째 사이트의 같은 VM 이 다른 IP 로 떠야 하고, 그 순간 게스트 OS 안의 설정·미들웨어 바인딩·클라이언트 화이트리스트가 모두 갱신 대상이 됩니다. 두 사이트가 동일 IP 대역을 공유하는 가상화 Active-Active 모델은 결국 광역 L2 네트워크라는 부가 인프라를 전제하며, 광역 L2 는 운영 단순성을 떨어뜨리고 장애 도메인을 두 사이트로 확장하는 부작용을 동반합니다. 가상화 Active-Active 가 회의실 슬라이드에 비해 실제 안착률이 낮은 1차 원인이 여기에 있습니다.

두 부담의 결합 효과. 라이선스 이중 지불과 광역 L2 운영 부담은 독립 사건이 아닙니다. 라이선스 비용을 정당화하려면 평시 두 사이트 모두에서 실제 트랜잭션을 처리해야 하고, 평시 트랜잭션을 처리하려면 두 사이트가 동일 IP 로 외부에 노출되어야 하며, 동일 IP 노출은 광역 L2 또는 글로벌 부하분산기를 전제로 합니다. 어느 한쪽을 포기하면 Active-Active 의 평시 가치가 함께 소실됩니다. 쿠버네티스는 이 결합을 두 측면에서 풀어냅니다. 첫째, 컨테이너 이미지는 라이선스가 부착되는 단위가 아니며 런타임을 이미지에 포함하면 게스트 OS 라이선스가 사라집니다 [S1]. 둘째, Service(서비스) 리소스가 IP 를 가상화하여 Pod(파드) IP 가 매니페스트에 명시된

외부 약속이 되지 않습니다 [S2]. 라이선스와 네트워크 종속이 동시에 풀리는 구조이며, 이 점이 2.2 절에서 다룰 쿠버네티스 Active-Active 성공 패턴의 전제가 됩니다.

가상화 라이선스 이중 지불과 광역 L2 운영 부담은 자사 가상화 라이선스 계약서의 코어·소켓 산정 조항을 첨부 자료로 두면 즉시 정량 환산이 가능합니다. 의사결정권자는 회의 안건에 "현 라이선스 모델로 Active-Active 를 평시 운영할 때 추가 청구되는 코어·소켓 수" 한 줄을 올리는 것만으로 본 항의 결론을 자사 환경에 적용할 수 있습니다.

2.1.2 공유 스토리지 쿼럼과 세션 종속

공유 스토리지 쿼럼 락의 본질. 가상화 환경의 고가용성은 일반적으로 공유 스토리지(SAN, NAS, vSAN) 위에 동작합니다. 동일 가상 디스크를 두 사이트의 하이퍼바이저가 동시에 마운트하면 파일시스템 일관성을 유지하기 위해 분산 락 매니저(distributed lock manager)가 개입합니다. 두 사이트가 같은 가상 디스크에 동시 쓰기를 시도하면 락 매니저는 쿼럼(quorum) 합의를 통해 한쪽의 쓰기만 허용합니다. 즉 두 사이트에 같은 VM 을 동시 가동해도 데이터 무결성을 위해 한쪽은 사실상 읽기 전용 상태로 강등되거나 일정 주기마다 락을 교대해야 합니다. 진정한 Active-Active 가 아니라 운영 부담만 늘어난 Active-Standby 변형으로 귀결됩니다. 사이트 간 거리가 늘어날수록 락 합의 지연도 함께 늘어나 응답시간 SLA 가 무너지는 사례가 흔합니다.

스트레치 클러스터의 한계. 공유 스토리지 쿼럼 문제를 해결하기 위해 벤더들은 스트레치 클러스터(stretch cluster) 와 동기 미러링을 결합한 솔루션을 제시합니다. 두 사이트가 같은 가상 디스크를 동기 복제로 미러링하고 락을 분산 합의로 관리하는 방식입니다. 그러나 동기 미러링은 광역 회선 대역폭과 1ms 이하 왕복 지연 같은 물리적 조건을 요구합니다. 국내 공공기관의 주센터·재해복구센터 간 거리가 수십~수백 km 에 이르는 일반적인 토폴로지에서는 동기 미러링의 지연 비용이 데이터베이스 응답시간을 두 자릿수 밀리초 이상 증가시키는 사례가 보고됩니다. 또한 회선 단절 시 쿼럼 합의 자체가 멈춰 두 사이트 모두 쓰기 불가 상태로 빠지는 split-brain 보호 모드가 작동합니다. Active-Active 의 평시 가치가 회선 사고 한 번에 양쪽 모두 정지되는 부작용으로 상쇄되는 구조입니다.

세션 종속이 만드는 트래픽 분기 한계. 가상화 환경의 상용 미들웨어(WAS, 세션 클러스터, 라이선스 서버)는 대부분 세션 어피니티(session affinity)를 가정하고 설계됐습니다. 같은 사용자의 요청이 같은 인스턴스로 라우팅되는 전제 위에서 메모리 캐시, 인증 토큰, 트랜잭션 상태가 유지됩니다. 두 사이트로 트래픽을 분기하면 사용자 요청이 매번 다른 사이트로 라우팅될 수 있고, 그 순간 세션이 유실되거나 두 사이트가 서로 다른 세션 상태를 들고 있는 상황이 발생합니다. 이를 방지하려면 세션 클러스터 자체를 두 사이트 간 동기 공유로 구성해야 하며, 다시 광역 회선 지연과 쿼럼 합의 비용을 짊어집니다. 가상화 시절의 Active-Active 시도가 실제 운영에 안착하지 못한 두 번째 원인이 이 세션 종속이며, 첫 번째 원인인 공유 스토리지 쿼럼과 결합하면 4가지 종속(라이선스·IP·스토리지·세션)이 서로를 강화하는 구조가 드러납니다.

왜 쿠버네티스에서는 이 문제가 풀리는가. 쿠버네티스 환경의 워크로드는 공유 스토리지 위가 아니라 Pod 단위로 분리된 컴퓨트 위에 동작합니다. 상태가 있는 워크로드는 StatefulSet(스테이트풀셋) 으로 분리되어 별도 일관성 모델(4장에서 다룸)을 따르고, 상태가 없는 워크로드는 단순히 두 클러스터에 동시 실행됩니다 [S2]. 세션은 메모리 캐시가 아니라 외부 세션 저장소(Redis, JWT 토큰 등)에 위임되어 사이트 간 종속이 사라집니다. 공유 스토리지 쿼럼 자체가 워크로드 실행 경로에서 빠지는 구조이며, 이 점이 2.2 절에서 다룰 쿠버네티스 Active-Active 성공 패턴의 두 번째 전제입니다.

가상화 Active-Active 의 실패는 단일 기술의 미숙이 아니라 라이선스·IP·스토리지·세션의 4중 종속이라는 모델 자체의 한계에서 비롯합니다. 자사가 과거 가상화 Active-Active 를 시도했다가 회귀한 경험이 있다면, 회귀 사유 보고서를 본 절의 4가지 종속과 대조해 보십시오. 4가지 중 어느 하나도 단독으로 해결되지 않았다는 점이 확인되면, 다음 절(2.2)에서 정리하는 쿠버네티스 모델이 같은 시도를 반복하지 않아도 되는 이유를 보여줍니다.

2.2 쿠버네티스 Active-Active 의 성공 패턴

쿠버네티스가 Active-Active 를 가능하게 하는 것은 단일 기능이 아니라 4가지 구조의 결합입니다. 선언적 배포(Declarative Deployment), 상태 비저장(Stateless) 워크로드 분리, Service Mesh(서비스 메시) 멀티 클러스터(Multi-cluster) 라우팅, GitOps 동기가 한 묶음으로 작동해야 두 사이트가 동일 서비스를 동시에 제공하는 구조가 안정화됩니다. 어느 하나가 빠지면 2.1 절에서 정리한 4중 종속(라이선스·IP·스토리지·세션) 중 하나가 다시 살아나면서 가상화 시절의 회귀 경로로 돌아갑니다.

2.2.1 선언적 배포 + 상태 비저장 + Service Mesh + GitOps 동기

구조 1 — 선언적 배포(Declarative Deployment). 쿠버네티스 워크로드는 명령형(imperative) 절차가 아니라 선언형(declarative) 매니페스트로 정의됩니다. Deployment(디플로이먼트), Service, ConfigMap(컨피그맵) 같은 리소스의 원하는 상태(desired state)를 YAML 매니페스트에 기술하면, 컨트롤 플레인(control plane)의 컨트롤러 루프가 실제 상태(actual state)를 원하는 상태와 일치시키는 조정(reconcile) 동작을 반복합니다 [S2]. 동일 매니페스트를 두 클러스터에 적용하면 두 클러스터 모두 동일한 desired state 를 보유하게 됩니다. Active-Active 의 정의가 "두 사이트가 같은 서비스를 동시 제공" 이라면, 같은 desired state 를 들고 있는 두 클러스터는 그 정의를 충족하는 가장 단순한 방법입니다. 가상화 시절의 "복제 후 부팅" 모델은 시점 백업에 의존했지만, 선언적 배포는 매니페스트라는 단일 SSoT(Source of Truth)에서 두 클러스터가 각자 자기 상태를 재현합니다. 매니페스트가 곧 운영 명세이자 DR 명세입니다.

구조 2 — 상태 비저장(Stateless) 워크로드 분리. 쿠버네티스 모범 사례는 워크로드를 상태 비저장(Stateless) 과 상태 저장(Stateful) 으로 명시 분리할 것을 요구합니다 [S2]. 상태 비저장 워크로드는 Deployment 로 정의되어 임의 노드에 임의 개수로 복제 가능하고 임의 시점에 종료·재생성 가능합니다. 상태 저장 워크로드는 StatefulSet 으로 분리되어 안정적 네트워크 식별자와 영속 볼륨(persistent volume)을 가집니다. Active-Active 의 핵심은 상태 비저장 워크로드를 두 클러스터에 동시 실행하고, 상태 저장 워크로드는 4장(데이터베이스 동기화) 에서 다루는 별도 일관성 모델로 분리하는 것입니다. 가상화 시절에는 VM 안에 애플리케이션 인스턴스와 데이터베이스 인스턴스가 한 게스트 OS 안에 묶여 있어 분리가 불가능했지만, 컨테이너 환경에서는 Deployment/StatefulSet 의 명시 분리가 강제됩니다. 이 분리 자체가 Active-Active 의 전제 조건이며, 분리 없이는 데이터베이스 쿼럼이 다시 애플리케이션 실행 경로에 끼어들게 됩니다.

구조 3 — Service Mesh 멀티 클러스터 라우팅. Istio(이스티오) 같은 Service Mesh 는 멀티 클러스터(Multi-cluster) 토폴로지를 공식 지원합니다. Istio 공식 docs 는 Multi-Primary, Primary-Remote, External Control Plane 세 가지 설치 모델을 제시하며, 두 클러스터가 동일 메시 안에서 서로의 서비스를 직접 발견·호출할 수 있게 합니다 [S3]. Multi-Primary 모델은 두 클러스터가 각자 독립 컨트롤 플레인을 가지면서도 워크로드 디스커버리·트래픽 라우팅·mTLS 인증을 통합 운영합니다. 가상화 시절의 광역 L2 가 만든 IP 종속이 Service Mesh 에서는 사라집니다 — 워크로드는 IP 가 아니라 서비스 이름으로 호출되며, 메시가 라우팅·재시도·서킷 브레이커를 책임집니다. 두 클러스터의 트래픽 분기는 매니페스트의 VirtualService(버추얼서비

스) 가중치 한 줄로 50/50, 90/10, 카나리 5/95 같은 비율 분기를 즉시 실현합니다. 가상화 환경에서는 글로벌 부하분산기에 별도 정책을 입력해야 가능했던 작업이, 메시에서는 매니페스트 변경 한 줄로 끝납니다.

구조 4 — GitOps 동기. Argo CD(아르고 CD) 같은 GitOps 컨트롤러는 Git 저장소에 보관된 매니페스트를 SSoT 로 삼아 두 클러스터의 실제 상태를 지속 동기화합니다 [S4]. ApplicationSet 컨트롤러는 단일 정의로 다수 클러스터에 동일 매니페스트를 자동 배포·갱신합니다. 매니페스트 변경이 Git 커밋으로 기록되므로 두 클러스터 사이의 상태 차이는 항상 Git 이력으로 추적 가능하며, 어느 한쪽이 표류(drift)하면 컨트롤러가 자동 복구합니다. Active-Active 운영에서 두 사이트가 시간이 지남에 따라 미세 차이를 누적하는 문제(가상화 환경의 "사이트 간 환경 차이" 사고로 흔히 나타나던)가 GitOps 에서는 표류 탐지로 즉시 가시화됩니다. 가상화 시절의 DR 훈련이 "복구 매뉴얼대로 잘 동작하는지 확인" 이었다면, GitOps 환경의 DR 훈련은 "두 클러스터의 desired state 가 Git 과 일치하는지 자동 확인" 으로 평시 운영의 일부가 됩니다. DR 이 별도 영역에서 평시 운영의 부산물로 이동하는 1차 메커니즘이 GitOps 입니다.

4가지 구조의 결합 효과. 4가지 구조 중 어느 하나가 빠지면 Active-Active 는 다시 가상화 시절의 함정에 빠집니다. 선언적 배포가 없으면 두 클러스터가 명령형 절차로 갈라져 표류를 누적합니다. 상태 비저장 분리가 없으면 데이터베이스 쿼럼이 애플리케이션 실행 경로에 끼어들어 가상화 시절의 공유 스토리지 쿼럼 문제를 반복합니다. Service Mesh 멀티 클러스터 라우팅이 없으면 IP 종속과 트래픽 분기 부담이 광역 L2 시절과 같습니다. GitOps 동기가 없으면 두 클러스터의 매니페스트 차이가 운영자 수작업으로 관리되어 사고 시 root cause 가 미궁에 빠집니다. 4가지가 한 묶음으로 동시 도입되어야 Active-Active 가 평시 운영의 부산물 위치로 이동합니다. 자사가 이미 도입한 구조와 미도입 구조를 분리해 보면 다음 도입 우선순위가 자연히 드러납니다 — 컨테이너 플랫폼은 있으나 Service Mesh 가 없다면 멀티 클러스터 라우팅이 다음 PoC 후보이며, 컨테이너 플랫폼·메시는 있으나 GitOps 가 없다면 ApplicationSet 도입이 다음 분기 안건입니다.

Submariner(서브마리너) 등 보조 프로젝트. Multi-Primary Istio 설치는 두 클러스터 사이의 Pod-to-Pod 네트워크 도달성을 전제합니다. CNCF 의 Submariner 프로젝트는 두 클러스터의 Pod CIDR(파드 CIDR) 을 안전한 터널로 연결하여 같은 메시 안의 워크로드가 서로 직접 통신할 수 있게 합니다. 별도 게이트웨이 컴포넌트가 클러스터 간 라우팅·DNS·보안을 책임지며, 메시는 그 위에서 서비스 디스커버리만 담당합니다.
Submariner 의 존재가 4가지 구조 외에 추가 도입 결정 항목임을 인지하면, 멀티 클러스터 네트워크 토폴로지 설계 시 누락 없이 평가할 수 있습니다.

2.2.2 트래픽 분기 3계층 — DNS · LB · Service Mesh

두 사이트의 Active-Active 트래픽을 어느 계층에서 분기할지는 워크로드 특성에 따라 다릅니다. DNS, 부하분산기(LB), Service Mesh 세 계층의 트레이드오프가 운영 안정성을 좌우합니다. 본 항은 3계층의 동작 원리와 전환 지연·세션 정합·운영 부담의 차이를 정리합니다.

계층 1 — DNS 기반 분기. DNS 가중치(weighted DNS)와 GSLB(Global Server Load Balancing) 가 두 사이트의 공개 IP 를 비율로 응답하는 방식입니다. 구현이 간단하고 기존 DNS 인프라를 재사용할 수 있습니다. 단점은 DNS TTL(Time To Live) 동안 캐시된 응답이 즉시 갱신되지 않는다는 점입니다. TTL 60초로 설정해도 일부 리졸버는 캐시를 더 오래 유지하며, 모바일 사업자 네트워크처럼 캐시 갱신이 지연되는 환경에서는 전환에 분 단위 시간이 걸립니다. Active-Active 운영의 평시 분기는 DNS 가중치로 충분하지만, 한쪽 사이트 장애 시 즉시 전환에는 한계가 있습니다. 세션 정합 측면에서도 클라이언트가 매 DNS 조회마다 다른 사이트로 라우팅될 수 있어 세션 어피니티가 약합니다 — 외부 세션 저장소가 없으면 곤란합니다. 운영 부담은 가장 낮으나 분기 정밀도가 가장 떨어집니다.

계층 2 — 부하분산기(LB) 기반 분기. 단일 글로벌 부하분산기(또는 BGP Anycast)가 두 사이트 백엔드에 트래픽을 분배하는 방식입니다. 헬스체크 기반 자동 전환이 가능하고 분기 비율을 즉시 변경할 수 있습니다. 가상화 시대에 가장 흔히 채택된 모델이며 컨테이너 환경에서도 그대로 활용 가능합니다. 단점은 글로벌 부하분산기 자체가 단일 실패점이 될 수 있다는 점입니다. 이를 회피하려면 부하분산기를 다중화하거나 BGP Anycast 로 분산해야 하며, 다시 네트워크 인프라 복잡도가 올라갑니다. 세션 어피니티는 cookie 기반 sticky session 으로 보장 가능하나, 사이트 장애 시 sticky 가 깨지면서 세션이 유실되는 위험이 있습니다. 운영 부담은 중간 수준이며, 기존 부하분산기 운영팀의 역량이 그대로 활용됩니다.

계층 3 — Service Mesh 기반 분기. Istio Multi-Primary 같은 멀티 클러스터 메시가 VirtualService 가중치로 트래픽을 분기하는 방식입니다 [S3]. 헬스체크는 메시의 사이드카(sidecar) 가 자동 수행하고, 분기 비율은 매니페스트 한 줄 변경으로 즉시 적용됩니다. 카나리 배포, A/B 테스트, 점진 전환(progressive rollout) 같은 고급 패턴을 메시 차원에서 표준화할 수 있습니다. 세션 어피니티는 메시의 consistent hashing 규칙으로 표현되며, 사이트 장애 시 메시가 정상 사이트로 자동 재라우팅합니다. 단점은 메시 자체의 운영 부담입니다 — 사이드카 자원 오버헤드, mTLS 인증서 관리, 멀티 클러스터 네트워크 토폴로지 설계가 모두 신규 역량을 요구합니다. Argo CD ApplicationSet 와 결합하면 매니페스트 SSoT 운영이 단순화되지만, 메시 자체 도입 학습 곡선이 가장 깊습니다 [S4].

3계층 선택의 의사결정 축. 세 계층은 배타가 아니라 누적입니다 — DNS 위에 LB, LB 위에 메시가 올라가는 다단 구조가 일반적입니다. 의사결정 축은 (a) 전환 지연 허용치, (b) 세션 정합 요구사항, (c) 메시 운영 역량 세 가지입니다. 분 단위 전환 허용 + 약한 세션 정합 + 운영팀 신규 역량 회피 → DNS 가중치 단독. 초 단위 전환 필요 + 강한 세션 정합 + 기존 부하분산기 역량 활용 → LB 다중화 + sticky session. 카나리·점진 전환 등 고급 패턴 필요 + 멀티 클러스터 운영 표준화 + 메시 역량 확보 → Service Mesh + ApplicationSet 결합. 자사 워크로드를 트래픽 패턴별로 분류하면 세 모델 중 하나에 자연히 매핑됩니다. 분기 계층 선택은 한 번 정하면 변경 비용이 크므로 워크로드별로 분리 결정하는 것이 안전합니다.

트래픽 분기 3계층 비교 — DNS · LB · Service Mesh

Active-Active 트래픽 분기 계층 선택의 전환 지연 · 세션 정합 · 운영 부담 트레이드오프

계층 / 메커니즘	전환 지연	세션 정합	운영 부담
DNS TTL 기반 (가중치 / GSLB)	분 ~ 시간 리졸버 캐시 갱신 지연 의존	약함 캐시 변동으로 어피니티 불안정	낮음 기존 DNS 인프라 재사용
부하분산기 (LB) Layer 4 분배 (BGP Anycast / VIP)	초 단위 헬스체크 기반 자동 전환	중간 Cookie sticky session 장애 시 어피니티 깨짐	중간 기존 LB 운영팀 역량 그대로 활용
Service Mesh L7 라우팅 (Istio Multi-Primary)	초 미만 매니페스트 한 줄 변경 즉시 적용	강함 mTLS + consistent hash 장애 시 자동 재라우팅	높음 Sidecar 자원·mTLS 인증서·메시 운영

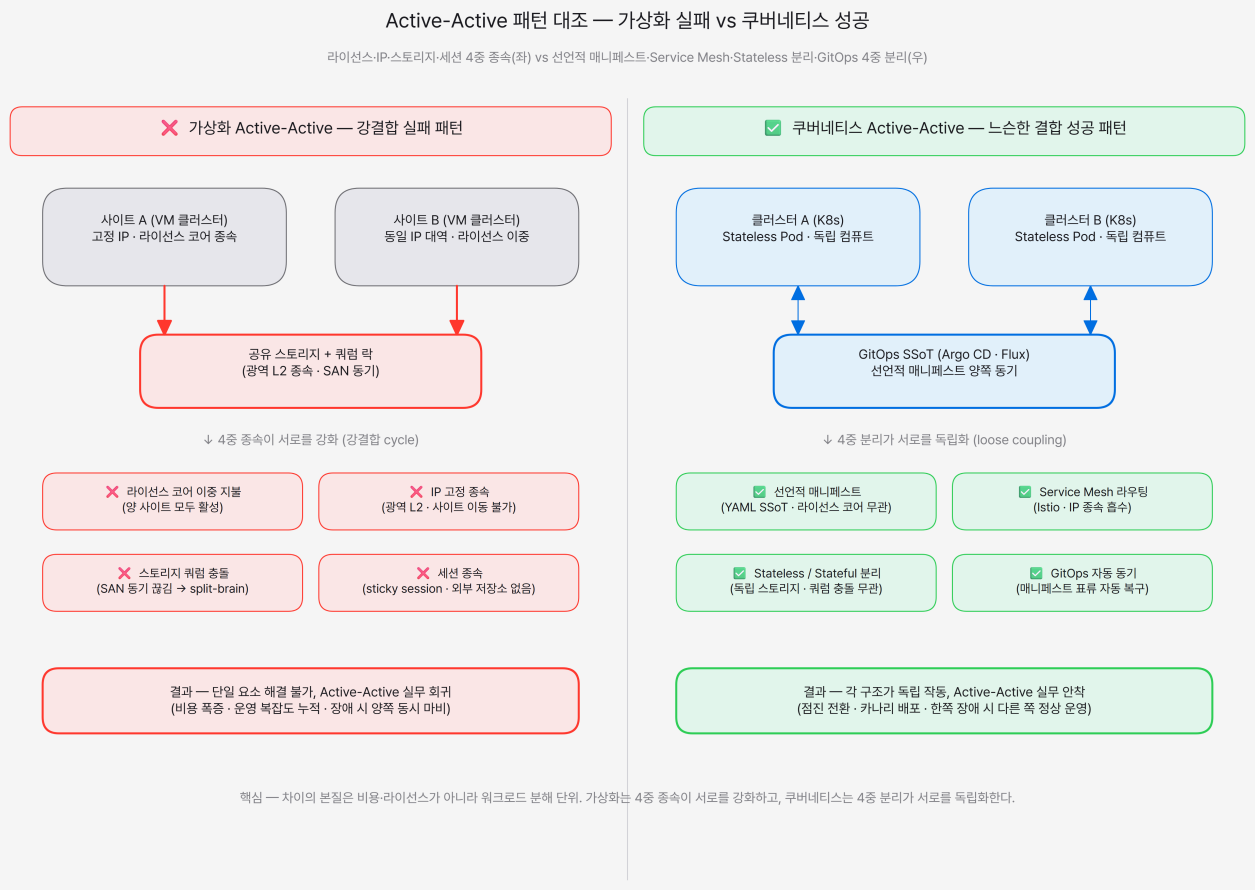
주: 세 계층은 배타가 아니라 누적 — DNS 위에 LB, LB 위에 Service Mesh 가 올라가는 다단 구조가 일반적이다. 워크로드별로 분리 결정한다.

(도식 설명: DNS / LB / Service Mesh 3계층의 트래픽 분기 경로와 트레이드오프 — 전환 지연 / 세션 정합 / 운영 부담 / 추천 워크로드 4개 축 비교표 + 다단 누적 토폴로지 다이어그램. 가상화 시절의 광역 L2 모델이 다

단 누적의 최하위에 놓이는 옵션 외(out-of-scope)로 표시되어, 3계층 선택이 광역 L2 의존 없이 가능하다는 점을 시각화한다.)

2.2.3 가상화·쿠버네티스 결합도 대조

가상화 Active-Active 와 쿠버네티스 Active-Active 의 본질적 차이는 비용·회선 같은 단일 요소가 아니라 워크로드 분해 단위 자체에 있습니다. 같은 그림 위에 두 패턴을 평행 배치하면 그 차이가 결합도(coupling) 의 차이로 정리됩니다.



위 도식은 상단에 가상화 Active-Active 실패 패턴, 하단에 쿠버네티스 Active-Active 성공 패턴을 배치합니다. 상단의 가상화 패턴은 두 사이트가 광역 L2 네트워크와 공유 스토리지 쿼럼으로 강하게 결합되어 있으며, 그 위에 라이선스 이중 지불 표식과 세션 종속 표식이 얹힙니다 — 4가지 종속이 서로를 강화하는 화살표가 결합도를 시각화합니다. 하단의 쿠버네티스 패턴은 두 클러스터가 독립 컴퓨트·독립 스토리지 위에 동작하면서 GitOps 매니페스트(가운데), Service Mesh 라우팅(상부), 상태 비저장/상태 저장 분리(하부) 세 요소로 느슨하게 연결됩니다 [S2][S3][S4]. 도식의 선 굵기가 결합도를 표현하여, 가상화의 강결합과 쿠버네티스의 느슨한 결합(loose coupling) 이 직관적으로 대비됩니다.

가상화 Active-Active 가 실무에 안착하지 못한 이유는 단순히 비용이 비싸서가 아닙니다. 라이선스·IP·스토리지·세션의 4중 종속이 서로를 강화하면서 어느 하나도 단독으로 해결할 수 없는 강결합 구조를 만들었기 때문입니다. 쿠버네티스 Active-Active 가 가능한 이유는 단순히 컨테이너가 가볍기 때문이 아닙니다. 워크로드를 상태 비저장/상태 저장으로 명시 분리하고, 선언적 매니페스트를 단일 SSoT 로 운용하며, Service Mesh 가 IP·라우팅 종속을 흡수하고, GitOps 가 두 클러스터의 표류를 자동 복구하는 4가지 구조가 한 묶음으로 동작하기

때문입니다. CNCF Annual Survey 2024 가 보고한 쿠버네티스 운영 환경 채택률 상승이 단순 기술 유행이 아니라 이러한 구조적 변화의 누적임을 도식이 함께 시사합니다 [S1].

본 장의 결론을 자사 환경에 적용하는 첫 번째 행동은 **자사 워크로드 중 Active-Active 후보 5종을 식별하고, 각각이 4가지 구조(선언적 배포, 상태 비저장 분리, Service Mesh 라우팅, GitOps 동기) 어느 단계까지 도입됐는지를 매트릭스로 작성**하는 일입니다. 매트릭스의 빈칸이 다음 분기 PoC 의 우선순위가 됩니다.

3장: 사이트 간 Active-Active 시 애플리케이션 이미지 동기화

두 사이트가 동시에 같은 서비스를 띄우는 Active-Active 구성에서 가장 먼저 합의해야 할 운영 질문은 회의실 언어로 이렇습니다. "양쪽 사이트가 지금 이 순간 같은 컨테이너 이미지를 띄우고 있다고 어떻게 보증합니까." 가상화 시대에는 이 질문이 잘 등장하지 않았습니다. VM 이미지는 통째로 한쪽 사이트에서 만들어 다른 쪽으로 복제하는 단방향 흐름이 일반적이었고, 두 사이트의 이미지가 동시에 변경될 일은 사실상 없었기 때문입니다. 그러나 컨테이너 환경에서는 CI/CD 파이프라인이 하루에도 수십 차례 새 이미지를 만들고, 두 사이트가 동시 운영되므로 같은 태그의 이미지가 양쪽에서 동시에 필요합니다. 이 장은 그 동기화를 어떻게 설계할지에 대한 의사결정 근거를 정리합니다.

장의 결론은 세 줄로 요약됩니다. 첫째, 각 사이트에 로컬 레지스트리를 두되 어느 레지스트리가 진실의 출처(SSoT)인지 명시적으로 결정합니다. 둘째, OCI Distribution Specification(이하 OCI 표준 사양)을 준수하는 레지스트리만 채택하여 향후 교체 비용을 낮춥니다. 셋째, Sigstore Cosign 서명과 SBOM(Software Bill of Materials, 소프트웨어 자재명세)을 Kyverno 같은 정책 엔진이 운영 시점에 검증하도록 만들어 이미지 무결성을 자동으로 보장합니다. 이 세 결정은 가상화 시대의 백업·복원 모델에서는 존재하지 않던 새로운 의사결정 영역이며, 거버넌스 보드 안건으로 한 번에 정리하는 것이 좋습니다.

3.1 컨테이너 이미지 SSoT 와 Geo-Replication 정책

두 사이트가 같은 태그로 다른 이미지를 띄우는 상황은 평시에는 드러나지 않다가 결제 정산 같은 일관성 결정적 처리에서 사고로 표면화됩니다. 이 위험을 회피하려면 어느 사이트의 레지스트리가 단일 진실 원본(SSoT) 인지 한 줄로 합의해야 하며, 그 합의 위에서 복제 정책의 운영 변수(주기·트리거·범위 필터·충돌 정책) 가 결정됩니다.

3.1.1 어느 레지스트리가 진실인가 — SSoT 정의 원칙

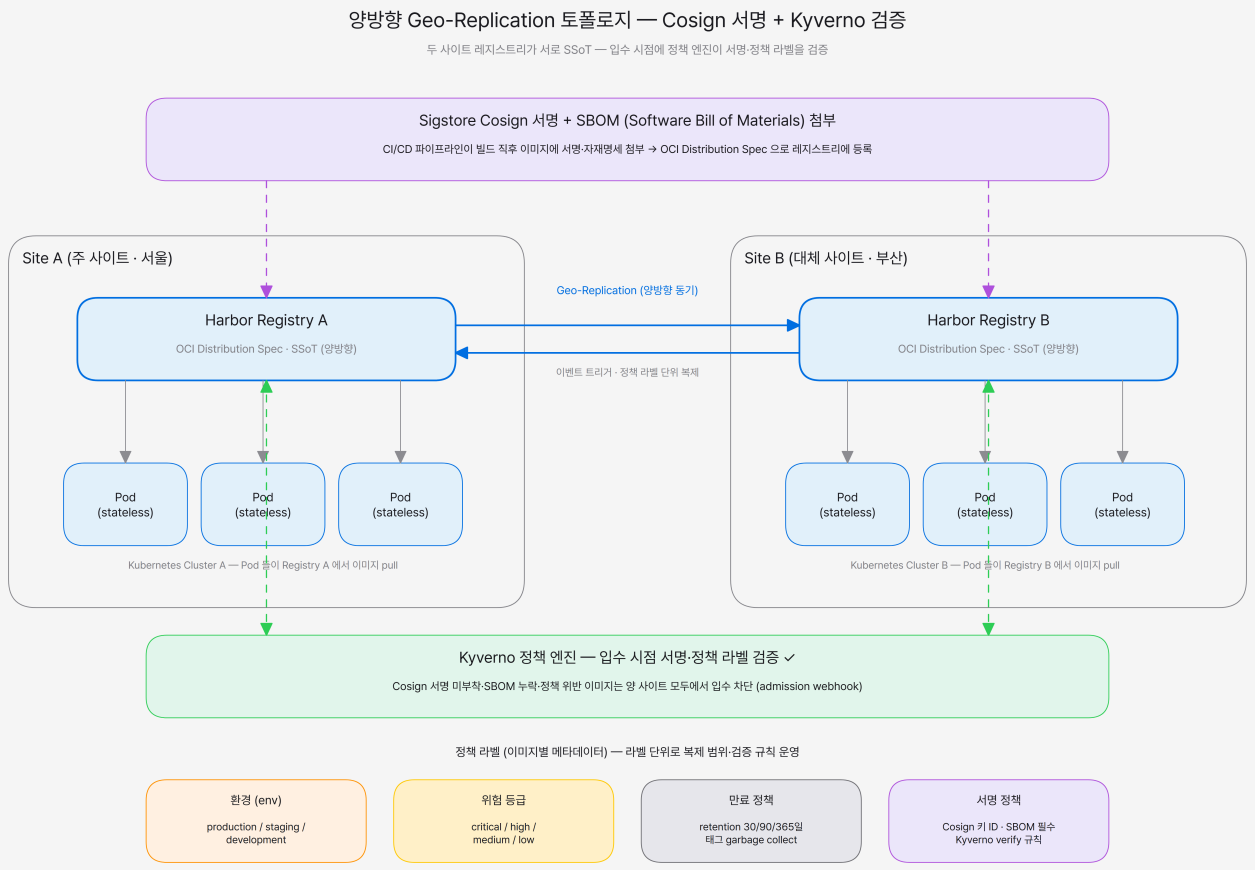
SSoT(Source of Truth, 단일 진실 원본) 가 미정인 상태가 가장 위험합니다. Active-Active 구성에서 SSoT 가 정의되지 않으면 두 사이트가 같은 태그로 다른 이미지를 띄울 위험이 그대로 노출됩니다. 예를 들어 `myapp:v1.4.2` 태그가 서울 사이트에서는 11월 12일 빌드를 가리키고 부산 사이트에서는 11월 13일 빌드를 가리킨다면, 사용자 트래픽이 어느 사이트로 분기되느냐에 따라 서비스 동작이 미묘하게 달라집니다. 이 차이는 평시에는 드러나지 않다가 결제 정산 같은 일관성 결정적 처리에서 사고로 표면화됩니다. SSoT 정의는 곧 "어느 사이트의 레지스트리에 등록된 이미지가 정답이며, 다른 사이트의 레지스트리는 그 정답의 복제본인가" 라는 한 줄 합의를 명문화하는 작업입니다.

SSoT 결정의 첫 번째 선택지는 단방향 패턴입니다. 한쪽 사이트(통상 주 사이트)의 레지스트리를 SSoT 로 두고 다른 사이트는 그 복제본만 받습니다. CI/CD 파이프라인은 SSoT 한 곳에만 push 하고, 부 사이트는 정해진 주기 또는 이벤트 트리거로 pull 합니다. 이 패턴은 운영 모델이 단순하고 권한 모델 설계가 쉽습니다. SSoT 사이트의 거버넌스만 잘 통제하면 부 사이트는 자동 동기화의 결과만 받아 사용합니다. Harbor 공식 문서는 이러한 단방향 복제 흐름이 가장 일반적인 출발 구성이라고 설명합니다 [S6]. 단점은 SSoT 사이트가 단일 장애점이

된다는 점입니다. 평시에는 한쪽이 모든 push 트래픽을 받으므로 가용성·성능 측면의 부담이 한곳에 집중됩니다.

두 번째 선택지는 양방향 패턴입니다. 두 사이트의 레지스트리가 서로 SSoT 이며, 한쪽에 등록된 이미지가 자동으로 반대쪽에도 복제됩니다. Active-Active 운영의 직관과 가장 잘 맞는 모델입니다. 하지만 양방향 패턴은 같은 태그가 양쪽에서 동시에 갱신될 가능성을 운영팀이 명시적으로 차단해야 합니다. 일반적으로는 이름 공간(namespace) 분리, 태그 정책, 정책 엔진의 입수 시점 검증 3가지 장치를 함께 적용합니다. 양방향 패턴이라고 해서 두 사이트가 서로 다른 빌드를 만들어 충돌하는 시나리오를 허용한다는 뜻은 아니며, CI/CD 파이프라인은 여전히 단일 빌드 산출물을 양쪽에 동시에 등록하는 형태로 운영하는 것이 안전합니다. Harbor 의 양방향 정책은 이 시나리오를 정책 단위로 라벨링하여 관리하는 방법을 제공합니다 [S6].

SSoT 결정은 거버넌스 보드의 의결 사항으로 두는 것을 권장합니다. SSoT 변경은 단순 운영 조정이 아니라 권한 모델·접근 통제·감사 추적 범위를 모두 바꾸는 결정입니다. 운영팀 자율로 두면 부 사이트에서 임시 패치 이미지를 등록하는 식의 우회 운영이 자라기 쉽고, 그 결과 양쪽 사이트가 동일하다는 보증이 깨집니다. 거버넌스 보드는 SSoT 변경 절차, 변경 권한자, 변경 이력 보존 기간, 변경 통보 채널 4가지를 같이 정해 둡니다. 이 절차가 있어야 사이버 재해나 회선 단절 시 어느 사이트의 이미지가 정답인지 한 줄로 답할 수 있습니다.



3.1.2 Geo-Replication 정책 카테고리 비교

SSoT 가 정해지면 다음 결정은 복제 정책의 운영 변수입니다. 주요 컨테이너 레지스트리(Harbor, Red Hat Quay, JFrog Artifactory) 가 공통으로 제공하는 Geo-Replication 정책 변수는 카테고리로 묶어 보면 4가지로 정리됩니다 — 복제 주기, 복제 트리거, 복제 범위 필터, 충돌·재시도 정책. 특정 제품을 추천하지 않으며 카테고리 단위 비교에 집중합니다.

복제 주기는 동기화가 일어나는 시간 단위입니다. 실시간 이벤트 기반(push/pull 이 발생할 때마다 즉시 복제), 짧은 주기(분 단위 폴링), 긴 주기(시간·일 단위 배치) 세 가지가 일반적입니다. Active-Active 구성에서는 이벤트 기반이 자연스럽지만, 회선 비용·대역폭 제약이 큰 망분리 환경에서는 정해진 시간대에 배치로 묶어 보내는 모델이 운영적으로 더 안정적인 경우가 있습니다. Harbor 공식 문서는 이벤트 기반과 스케줄 기반을 정책 단위로 혼합 적용할 수 있다고 설명합니다 [S6].

복제 트리거는 어떤 사건이 복제를 시작시키는지 정의합니다. 이미지 push, 매니페스트 변경, 태그 추가, 서명 추가 등이 트리거 후보입니다. 트리거 단위가 곱게 나뉠수록 정책 표현이 유연해지지만, 운영 회의에서 정책을 한 줄로 설명하기 어려워집니다. 권장은 트리거 단위를 "이미지 push 이벤트" 한 가지로 통일하고, 나머지 변경은 push 이벤트 안에 포함되도록 CI/CD 파이프라인을 설계하는 방법입니다.

복제 범위 필터는 어느 이미지를 복제 대상에 포함하고 어느 것을 제외할지 결정하는 규칙입니다. 일반적으로 이름 패턴(`namespace/*`), 태그 패턴(`v[0-9]+.[0-9]+.[0-9]+`), 라벨(label) 기반 필터, 서명 유무 4가지 필터를 조합합니다. 운영 환경 구분(production/staging/development), 위험 등급, 라이선스 분류 같은 메타 데이터를 라벨로 부착해 두면, 정책 변경 시 코드 수정 없이 라벨 규칙만 갱신하면 됩니다. 정책 단위 라벨링은 운영팀이 회의실에서 정책을 인쇄해 한 장으로 검토할 수 있을 만큼 단순해야 하며, 이 단순성이 회선 단절 시 재동기 비용을 결정합니다.

충돌·재시도 정책은 회선 단절·일시 오류 발생 시 동작을 정의합니다. 단방향 패턴에서는 재시도 정책만 정의하면 충분합니다. 양방향 패턴에서는 같은 태그가 양쪽에서 동시에 갱신된 상황을 어떻게 해결할지(타임스탬프 우선, 사이트 우선, 운영팀 수동 결정) 미리 합의해야 합니다. Active-Active 운영의 상식적 권장은 같은 태그의 동시 갱신을 정책 엔진 단계에서 차단하고, CI/CD 가 단일 빌드 산출물을 양쪽에 동시 등록하는 모델로 일원화하는 것입니다 [S6].

컨테이너 레지스트리 Geo-Replication 정책 비교

Harbor / Quay / JFrog Artifactory — 4가지 카테고리로 평가

레지스트리	정책 단위	동기 주기	충돌 정책	무결성	운영 부담
Harbor	프로젝트 단위	즉시 또는 cron	last-write-wins	Cosign 통합	중간 (오픈소스)
Quay (Red Hat Quay)	리포지토리 단위	즉시 또는 예약	last-write-wins	서명 통합	중간
JFrog Artifactory	리포지토리 단위	즉시 또는 cron	push/pull 분리	SBOM 첨부	높음 (엔터프라이즈)

세 레지스트리 모두 OCI 표준 준수 — 본질은 같고 표현이 다름. RFP 시 위 4가지 기준으로 후보를 동일하게 평가 [S6]

세 레지스트리 모두 OCI 표준 사양을 준수하므로, 정책 단위 라벨링과 복제 트리거의 표현 방식은 다를지언정 운영 모델의 본질은 같습니다. RFP 평가 시 한 제품을 영구 고정으로 가정할 필요는 없으며, 위 4가지 카테고리에 자사 요구사항을 채워 두면 향후 교체 시점에 동일 기준으로 후보를 평가할 수 있습니다.

3.2 무결성 보장 — OCI Distribution Spec 와 Cosign 서명

가상화 시대의 신뢰 모델이 "백업 매체 자체를 신뢰한다" 는 가정 위에 서 있었다면, 컨테이너 환경의 신뢰 모델은 "각 이미지가 누가 만들었고 어떻게 변하지 않았는지를 암호 서명으로 증명한다" 는 가정으로 이동합니다. 백업 매체가 랜섬웨어의 1차 표적이 되는 최근 사이버 재해 시나리오에서, 이 신뢰 모델의 차이가 강력한 차별점을 만듭니다.

3.2.1 OCI Distribution Specification 의 역할

OCI Distribution Specification 은 컨테이너 이미지가 레지스트리와 클라이언트 사이에서 어떻게 전송·저장·조회되는지를 정의하는 표준 사양입니다. Open Container Initiative(OCI) 가 관리하며, 매니페스트(manifest) 구조, 블롭(blob) 업·다운로드 절차, 콘텐츠 주소 지정(content-addressable storage) 규칙, 인증·인가 흐름을 표준 사양으로 명시합니다 [S7]. 주요 컨테이너 레지스트리와 클라이언트(docker, podman, crane, oras 등)가 이 사양을 구현하므로, 사양 준수 레지스트리 사이에서 이미지가 호환되어 움직입니다.

이 표준 사양이 멀티 사이트 운영에 주는 효과는 세 가지입니다. 첫째, 호환성입니다. 양쪽 사이트가 서로 다른 레지스트리 제품을 사용하더라도 같은 OCI 사양을 따르면 이미지를 그대로 주고받을 수 있습니다. 한쪽은 Harbor, 다른 쪽은 Quay 같은 혼합 운영도 표준 사양을 매개로 가능해집니다. 둘째, 이식성입니다. 향후 레지스트리 교체 결정 시 데이터 형식 변환 없이 그대로 옮길 수 있으므로 교체 비용이 운영 절차 변경 비용으로 한정됩니다. 셋째, 검증 가능성입니다. 콘텐츠 주소 지정 규칙(이미지의 SHA-256 다이제스트가 곧 식별자)에 따라 같은 다이제스트의 이미지는 어디서 가져오더라도 비트 단위로 동일함이 보장됩니다. 이 비트 단위 동일성이 다음 절의 서명 검증의 전제 조건입니다.

RFP 평가 항목에 OCI 표준 사양 준수 여부를 명시하는 것을 권장합니다. 단순히 "컨테이너 레지스트리 지원" 으로 적으면 사양 미준수 제품도 응찰 가능하지만, "OCI Distribution Specification v1.1 이상 준수" 로 명시하면 향후 교체 비용의 상한이 정해집니다. 이 한 줄이 5년 뒤 마이그레이션 비용을 결정합니다 [S7].

3.2.2 Sigstore Cosign 으로 본 서명·SBOM·정책 통합

Sigstore Cosign(이하 Cosign) 은 컨테이너 이미지에 암호 서명을 첨부하고 검증하는 도구입니다. Sigstore 프로젝트는 Linux Foundation 산하 OpenSSF(Open Source Security Foundation) 이 후원하며, Cosign 은 그 가운데 가장 널리 사용되는 클라이언트 도구입니다. 키 기반 서명, 키 없이(keyless) Fulcio 인증 기관과 OIDC 신원을 결합한 서명, Rekor 투명성 로그(transparency log) 에 서명 이력을 기록하는 흐름을 모두 지원합니다 [S7]. 서명은 이미지 자체와 분리된 OCI 아티팩트로 같은 레지스트리에 저장되며, 따라서 Geo-Replication 정책이 이미지를 복제할 때 서명도 함께 복제됩니다.

운영 시점의 신뢰 흐름은 4단계로 정리됩니다. CI/CD 파이프라인이 이미지를 빌드하면 곧바로 Cosign 으로 서명을 만들고 SBOM(Software Bill of Materials, 소프트웨어 자재명세) 을 함께 첨부합니다. 서명과 SBOM 은 이미지와 함께 SSoT 레지스트리에 등록됩니다. Geo-Replication 정책이 이미지·서명·SBOM 세 묶음을 부 사이트로 복제합니다. 두 사이트의 Kubernetes 가 Pod 를 실제로 띄울 때 입수 시점 검증(admission control) 에서 정책 엔진이 서명을 검증합니다. 검증에 실패하면 Pod 가 기동되지 않습니다.

정책 엔진은 Kyverno 또는 OPA Gatekeeper 가 일반적입니다. Kyverno 는 Kubernetes 네이티브 정책 도구로 YAML 정책을 작성해 서명 검증, SBOM 존재 확인, 라벨 일치 같은 규칙을 강제합니다. OPA Gatekeeper 는 Rego 언어로 더 복잡한 정책을 표현할 수 있습니다. 두 도구 모두 입수 시점 웹훅(admission webhook) 으로 동작하므로 검증 통과 이미지만 클러스터에 진입합니다. 사이버 재해 시나리오에서 침입자가

이미지를 위조해 부 사이트 레지스트리에 밀어 넣더라도, 정책 엔진이 서명 검증에 실패하므로 Pod 가 기동되지 않습니다. 이것이 컨테이너 환경의 신뢰 부트(trusted boot) 가 가상화 시대의 백업·복원 모델보다 강력한 이유입니다 [S7].

ISMS-P 통제 가운데 "정보시스템 변경 통제", "악성 코드 통제", "재해복구 무결성 확인" 항목은 이 4단계 흐름과 직접 매핑됩니다. 감사 시 Rekor 투명성 로그가 변경 추적의 외부 증빙 역할을 하며, 운영팀의 수작업 점검을 정책 엔진이 자동화합니다. 이 자동화는 인적 의존성을 줄이고 야간·주말 운영의 안정성을 높이므로, 정책 엔진 도입은 ISMS-P 통제 자동화의 첫 번째 후보로 권장됩니다 [S7].

3.2.3 거버넌스 보드 안건 체크리스트

3장이 정리한 세 결정(SSoT, OCI 표준 사양, Cosign 서명 + 정책 엔진) 은 운영팀 자율로 두면 우회 운영이 자라기 쉽기 때문에, 거버넌스 보드의 정식 의결 사항으로 한 번에 묶어 정리하는 것이 효과적입니다. 다음 체크리스트는 회의 안건 1건으로 통과시킬 수 있는 7개 항목입니다.

결정 영역	안건 항목	의결해야 할 한 줄
SSoT	① SSoT 사이트 지정	"주 사이트 레지스트리가 SSoT, 부 사이트는 복제본" 또는 "양방향 SSoT + 이름 공간 분리" 둘 중 하나 명시
SSoT	② SSoT 변경 권한자	SSoT 변경 권한을 가진 직책 명시 + 변경 절차 문서화 + 감사 로그 보존 기간
Geo-Replication	③ 복제 4변수 채움	복제 주기 / 트리거 / 범위 필터 / 충돌·재시도 정책 4가지 카테고리에 자사 운영 값 채움
OCI 표준 사양	④ 신규·교체 RFP 평가 항목	"OCI Distribution Specification v1.1 이상 준수" 를 평가 표 필수 항목으로 등록
서명·SBOM	⑤ CI/CD 파이프라인 통합	빌드 직후 Cosign 서명 + SBOM 첨부를 파이프라인 표준 단계로 정의
정책 엔진	⑥ 입수 시점 검증 강제	Kyverno 또는 OPA Gatekeeper 정책으로 미서명·SBOM 누락 이미지 차단을 운영·DR 두 클러스터 모두에 동시 적용
감사	⑦ ISMS-P 매핑 문서화	위 6개 항목을 ISMS-P "변경 통제", "악성 코드 통제", "재해복구 무결성 확인" 항목과 매핑한 표를 감사 증빙으로 보존

7개 항목 가운데 ⑥ 입수 시점 검증 강제가 가장 중요합니다. 사이버 재해 시나리오에서 침입자가 위조 이미지를 부 사이트 레지스트리에 밀어 넣더라도 정책 엔진이 서명 검증에 실패하면 Pod 가 기동되지 않으며, 이것이 컨테이너 환경의 신뢰 부트가 가상화 시대의 백업·복원 모델보다 강력한 1차 메커니즘입니다 [S7]. ⑥ 없이 ①~⑤만 도입되면 거버넌스 형식만 갖추고 실제 차단 효과는 없는 상태가 됩니다.

이미지 동기화가 완료되어도 데이터가 동기화되지 않으면 Active-Active 는 절반만 완성된 상태입니다. 워크로드 유형에 따라 동기 복제, CDC 비동기, 분산 SQL, NoSQL multi-DC 중 어느 카테고리를 적용할지가 다음 장의 핵심 결정입니다.

4장: App Active-Active 상태에서의 데이터베이스 동기화

애플리케이션 계층을 두 사이트에서 동시에 띄우는 일은 앞 장에서 설명한 선언적 배포와 Service Mesh 라우팅으로 정리된다. 남는 질문은 한 줄이다. **두 사이트가 동일한 데이터를 동시에 쓰면 일관성은 어떻게 보장하는가.** 이 질문에 단일 정답은 존재하지 않는다. 워크로드 유형에 따라 동기 복제, CDC(Change Data Capture, 변경 데이터 캡처) 비동기, 분산 SQL(Distributed SQL), NoSQL multi-DC(Multi-Data-Center) 4 카테고리를 선택적으로 적용해야 하며, 일관성·복구 시점·지연 비용의 트레이드오프를 워크로드 단위로 명시적으로 합의해야 한다 [S8].

각 카테고리는 회피할 수 없는 비용을 진다 — 강한 일관성은 지연 비용을, 결과적 일관성은 충돌 해결 정책 비용을, 분산 SQL 은 운영 모델 전환 비용을, NoSQL multi-DC 는 트랜잭션 의미 제약을 각각 대가로 요구한다. 운영팀이 "어느 데이터베이스 제품을 쓸 것인가" 가 아니라 "어느 워크로드에 어느 카테고리를 매핑할 것인가" 로 회의의 출발점을 바꾸는 순간, 결정 시간이 분 단위로 단축되고 사후 SLA 분쟁이 사라진다.

4.1 워크로드 유형과 동기화 카테고리 매핑

데이터베이스 동기화 카테고리를 선택하는 의사결정의 출발점은 데이터베이스 제품이 아니라 워크로드 유형이다. 동일 조직 안에서도 트랜잭션 처리 워크로드는 강한 일관성을 요구하고, 세션 저장 워크로드는 가용성을 우선하며, 로그·이벤트 워크로드는 결과적 일관성으로 충분하다. 분석 워크로드는 지연을 허용하는 대신 대용량 처리량을 요구한다. 4 카테고리는 이러한 워크로드 분류와 1:1 매핑되지 않으나, 4.2 절의 매트릭스로 권장 조합을 정리할 수 있다.

4.1.1 동기 복제와 CDC 비동기 — 강한 일관성 vs 결과적 일관성

동기 복제 카테고리의 동작 원리. 동기 복제 카테고리는 한 사이트의 쓰기가 다른 사이트의 복제본에 반영되어 합의에 도달한 뒤에야 클라이언트에 성공을 반환한다. 대표 구현은 세 가지다. PostgreSQL 환경에서는 Patroni 가 두 사이트의 인스턴스를 단일 리더-팔로워 토폴로지로 묶고 etcd 또는 Consul 같은 외부 분산 합의 저장소로 리더 선출을 관리한다 [S8]. MySQL Group Replication 은 Paxos 변형 합의 알고리즘으로 멤버 간 트랜잭션 순서를 결정하고 쿼럼(quorum) 다수에 반영된 트랜잭션만 커밋 확정한다 [S8]. MariaDB Galera Cluster 는 wsrep(write-set replication) 인터페이스 기반으로 인증(certification) 단계에서 충돌을 사전 검출하고 모든 노드에 동일 순서로 적용한다 [S8].

세 구현 모두 강한 일관성(strong consistency)을 보장하나 공통의 비용을 진다. 첫째, 쿼럼 다수의 응답을 기다려야 하므로 사이트 간 왕복 지연이 트랜잭션 지연에 그대로 더해진다. 수도권-남부권 같은 국내 두 사이트 사이의 왕복 지연이 평균 8~15ms 라면, 사이트 간 동기 복제 트랜잭션의 평균 지연은 단일 사이트 트랜잭션 대비 최소 그만큼 늘어난다. 둘째, 쿼럼 정족수 손실 시 양 사이트 모두 쓰기를 거부한다 — 2 사이트 구성에서 한 사이트의 단순 단절이 곧 전체 쓰기 차단으로 이어진다. 셋째, 세 인스턴스 이상을 두려면 제3의 사이트(arbiter)가 필요하며 망분리 환경에서는 이 위치 선정 자체가 거버넌스 안건이 된다.

CDC 비동기 카테고리의 동작 원리. CDC 비동기 카테고리는 데이터베이스의 변경 로그(PostgreSQL WAL, MySQL binlog, MongoDB oplog)를 외부 컴포넌트가 구독해 다른 사이트로 전파하는 패턴이다. 대표 구현은 Debezium + Kafka MirrorMaker 2 조합이다 [S8]. Debezium 의 소스 커넥터가 데이터베이스 변경 로그를 Kafka 토픽으로 발행하고, MirrorMaker 2 가 토픽을 다른 사이트의 Kafka 클러스터로 복제하며, 반대편의 싱크 커넥터가 변경을 다른 사이트의 데이터베이스에 적용한다. 한 사이트의 트랜잭션은 자기 사이트에 커밋되면 클라이언트에 즉시 성공을 반환한다 — 복제 전파는 비동기로 수 ms~수 초 뒤에 일어난다.

CDC 비동기 카테고리의 강점은 트랜잭션 지연을 사이트 내부 지연 수준으로 유지하면서 결과적 일관성(Eventual Consistency)을 달성한다는 점이다. 반대로 두 가지 추가 비용을 진다. 첫째, 두 사이트가 동일 키에 동시 쓰기를 수행하면 충돌이 발생하며 별도 충돌 해결 정책(4.2.2 절)이 필요하다. 둘째, 복제 파이프라인 자체가 운영 컴포넌트가 되어 모니터링, 백프레셔(backpressure) 관리, 메시지 재처리 정책 같은 운영 책임이 늘어난다.

두 카테고리의 워크로드 합의 지점. 동기 복제 vs CDC 비동기 선택은 SLA(Service Level Agreement, 서비스 수준 합의) 결정 회의에서 두 질문으로 좁힐 수 있다. 첫째 질문은 "한 사이트에 쓰인 트랜잭션이 다른 사이트의 읽기에 즉시 보여야 하는가" 다. 답이 "그렇다" 면 동기 복제다. 둘째 질문은 "두 사이트 중 하나가 단절될 때 나머지 사이트가 단독 쓰기를 계속해도 되는가" 다. 답이 "그렇다" 면 CDC 비동기다. 두 답이 모두 "그렇다" 면 두 요구가 본질적으로 충돌하므로 SLA 자체를 재합의해야 한다 — CAP 정리가 회의실에 데려오는 순간이다.

표 4-1 은 두 카테고리의 정량 트레이드오프를 정리한다.

비교 차원	동기 복제 (Patroni / MySQL Group Replication / Galera)	CDC 비동기 (Debezium + Kafka MirrorMaker 2)
일관성 모델	강한 일관성 (linearizable 수준)	결과적 일관성
RPO (이상 시)	0 (커밋된 데이터 손실 없음)	복제 지연 만큼 (수 ms~수 초)
쓰기 지연 영향	사이트 간 왕복 지연이 더해짐	사이트 내부 지연 수준 유지
사이트 단절 시 동작	쿼럼 손실 시 쓰기 거부	양 사이트 단독 쓰기 가능 (충돌 정책 필요)
추가 운영 컴포넌트	외부 합의 저장소 (etcd / Consul)	Kafka + Connect 워커 + MirrorMaker 2
충돌 해결 정책	불필요 (단일 순서 강제)	last-write-wins / CRDT / 비즈니스 규칙 중 선택
권장 워크로드	금융 거래, 재고 차감, 결제 승인	세션, 사용자 행동 로그, 이벤트, 카탈로그 캐시

권장 워크로드 컬럼이 SLA 합의 회의의 첫 짝짓기 입력값이 된다. PostgreSQL/MySQL 운영 인력이 두텁고 결제·재고 같은 강한 일관성 워크로드 비중이 크면 동기 복제부터 검토하고, 세션·로그·이벤트 비중이 크고 사이트 단절 시 단독 쓰기 지속이 요구사항이면 CDC 비동기부터 검토한다.

4.1.2 분산 SQL 과 NoSQL multi-DC

분산 SQL 카테고리의 동작 원리. 분산 SQL 카테고리는 데이터베이스 엔진 자체가 분산·복제·합의를 핵심으로 다루는 신세대 제품군이다. 대표 구현은 세 가지다. CockroachDB 는 Spanner 모델을 단순화한 다중 활성 트랜잭션을 제공하며 범위(range) 단위로 Raft 합의를 수행한다 [S8]. YugabyteDB 는 PostgreSQL 와이어 프로토콜 호환 SQL 계층과 별도 분산 키-값 저장소를 결합한 2계층 구조다 [S8]. PingCAP TiDB 는 MySQL 와이어 프로토콜 호환 SQL 계층과 TiKV 분산 키-값 저장소를 분리한 구조로 OLTP 와 OLAP 를 동일 클러스터에서 처리한다 [S8].

세 제품 모두 글로벌 일관성을 자체 보장한다. 운영팀은 외부 합의 저장소를 별도로 운영할 필요가 없고, 별도 충돌 해결 정책을 합의할 필요도 없다. 두 사이트 동시 쓰기는 데이터베이스 내부에서 단일 순서로 정렬되어 처리된다. 대신 두 종류의 전환 비용이 발생한다. 첫째는 운영 모델 전환 비용이다. 기존 PostgreSQL 운영 인력이 보유한 백업, 통계, 인덱스 튜닝, 실행 계획 분석 노하우는 분산 SQL 에서 부분적으로만 재사용된다. 분산 트랜잭션의 실행 계획, 범위 분할 정책, 핫스팟 회피 설계는 새 학습 영역이다. 둘째는 애플리케이션 SQL 호환성 검증 비용이다. PostgreSQL 호환을 표방하는 YugabyteDB 도 일부 확장 기능(특정 GIN/GiST 인덱스 패턴, 일부 트리거 동작)은 차이가 있어 마이그레이션 시 회귀 시험이 필수다 [S8].

이런 비용 때문에 분산 SQL 카테고리는 신규 워크로드 후보로 우선 검토하고, 기존 PostgreSQL/MySQL 워크로드는 동기 복제 또는 CDC 비동기 카테고리로 먼저 정리한 다음 단계적 교체 후보로 다루는 것이 운영 안정성에 부합한다.

NoSQL multi-DC 카테고리의 동작 원리. NoSQL multi-DC 카테고리는 가용성 우선 모델(AP 모델)을 데이터베이스 차원에서 받아들인 제품군이다. 대표 구현은 세 가지다. MongoDB Sharded Cluster 는 샤드 단위로 데이터를 수평 분할하고 각 샤드는 자체 복제본 세트를 갖는다 — multi-DC 배포에서는 복제본 세트 멤버를 두 사이트에 분산 배치하고 별도 arbiter 위치를 결정한다 [S8]. Apache Cassandra 는 처음부터 multi-DC 토폴로지를 핵심 단위로 설계한 시스템으로, 키스페이스 단위 복제 전략(NetworkTopologyStrategy)으로 사이트당 복제 수를 독립 설정하고 쿼럼 일관성 수준을 LOCAL_QUORUM, EACH_QUORUM 등 세분화된 옵션으로 선택한다 [S8]. ScyllaDB 는 Cassandra 와 동일 와이어 프로토콜을 유지하면서 C++ 재구현으로 처리량을 끌어올린 변형이다.

NoSQL multi-DC 카테고리의 강점은 사이트 단절 시 양 사이트가 모두 단독 쓰기를 계속한다는 점, 그리고 일관성 모델을 쿼리 단위로 선택할 수 있다는 점이다. 동일 클러스터에서 어떤 쿼리는 LOCAL_QUORUM 으로 빠르게 처리하고 다른 쿼리는 EACH_QUORUM 으로 양 사이트 동시 합의를 요구할 수 있다 [S8]. 반대로 트랜잭션 의미가 제한되어 다중 키 ACID 가 필요한 워크로드에는 부적합하다.

두 카테고리의 워크로드 합의 지점. 분산 SQL vs NoSQL multi-DC 선택의 핵심 질문은 "이 워크로드가 다중 키 트랜잭션의 강한 의미를 필요로 하는가" 다. 답이 "그렇다" 면 분산 SQL 후보다. 답이 "단일 키 또는 키 묶음 안 트랜잭션이면 충분하다" 면 NoSQL multi-DC 후보다. 두 카테고리의 정량 트레이드오프는 표 4-2 와 같다.

비교 차원	분산 SQL (CockroachDB / YugabyteDB / TiDB)	NoSQL multi-DC (MongoDB Sharded / Cassandra / ScyllaDB)
일관성 모델	글로벌 강한 일관성 (직렬화 가능)	쿼리 단위 선택 (LOCAL_QUORUM ~ EACH_QUORUM)
트랜잭션 의미	다중 키 ACID 보장	단일 키 또는 키 묶음 내 보장

비교 차원	분산 SQL (CockroachDB / YugabyteDB / TiDB)	NoSQL multi-DC (MongoDB Sharded / Cassandra / ScyllaDB)
사이트 단절 시 동작	쿼럼 손실 영역 쓰기 차단	양 사이트 단독 쓰기 계속
충돌 해결	데이터베이스 내부에서 단일 순서로 정렬	last-write-wins (Cassandra) 또는 응용 책임
운영 모델 전환 비용	큼 (분산 트랜잭션·핫스팟 설계 학습 필요)	보통 (스키마·쿼리 패턴 재설계 필요)
SQL 호환성	PostgreSQL/MySQL 부분 호환	CQL 또는 MongoDB 쿼리 언어
권장 워크로드	신규 글로벌 결제·재고, 다지역 OLTP	시계열, 카탈로그, 추천, 로그, 메시지

4.2 워크로드별 매트릭스와 충돌 해결 토폴로지

4 카테고리의 트레이드오프가 SLA 합의 회의의 의제로 바뀌려면 워크로드 유형 축이 한 번 더 필요하다. 트랜잭션·세션·로그·분석 4 워크로드와 4 카테고리를 교차한 매트릭스(4.2.1)가 그 도구이며, CDC 비동기 또는 NoSQL multi-DC 가 채택된 경우 추가로 따라오는 충돌 해결 정책의 3 방식(4.2.2)이 거버넌스 의결 안건의 마지막 의제다.

4.2.1 4 카테고리 × 4 워크로드 매트릭스

4 카테고리는 4 워크로드 유형(트랜잭션, 세션, 로그, 분석)과 교차해 다음 권장 매트릭스를 만든다.

DB 동기화 4 카테고리 · 핵심 속성 비교 매트릭스

각 카테고리의 일관성 · RPO · 지연 · 운영 난이도 · 추천 워크로드를 한 표로 정리. 카테고리 결정의 출발점.

카테고리	일관성 모델	복구 시점 (RPO)	지연 비용	운영 난이도	추천 워크로드
동기 복제 Patroni · MySQL Group Replication · Galera	강한 일관성 (쿼럼 동기)	0 (동기)	높음 (쿼럼 round-trip)	중간 (쿼럼 모니터링)	트랜잭션 처리 (결제 · 재고 · 계좌) RPO 0 필수 워크로드
CDC 비동기 Debezium + Kafka MM2	결과적 일관성 (eventual)	초 ~ 분	낮음 (비동기)	중상 (충돌 정책 합의 필요)	로그 · 이벤트 (감사 · 행동 분석) 결과적 일관성 충분
분산 SQL CockroachDB YugabyteDB · TiDB	글로벌 일관성 (Raft 합의)	0 (합의 동기)	중간 (지역 RTT 의존)	높음 (운영 모델 전환 필요)	신규 트랜잭션 · OLTP 글로벌 일관성 필수 신규 분석 워크로드
NoSQL multi-DC MongoDB Sharded Cassandra · ScyllaDB	가용성 우선 (CAP — AP)	초	낮음 (키 단위 처리)	중간 (샤딩 토폴로지 설계 필요)	세션 · 분석 (장바구니 · 로그) 시계열 · 대용량 처리

셀 배경색 = 속성 유리도 유리 보통 불리 (비용·난이도 큼)

매트릭스의 권장값은 단일 정답이 아니라 워크로드 SLA 합의의 출발점이다. 회의실에서 자사 워크로드 약어를 직접 적어 채우는 워크북 형태로 운용하면 의사결정 회의가 한 표로 끝난다.

워크로드 / 카테고리	동기 복제	CDC 비동기	분산 SQL	NoSQL multi-DC
트랜잭션 (결제·재고·계좌 이체)	**권장** — RPO 0 필수	비권장 (충돌 위험)	**신규 우선 권장** — 글로벌 일관성	비권장 (다중 키 ACID 부족)
세션 (로그인 토큰·장바구니)	비권장 (지연 비용 과대)	**권장** — 가용성 우선	가능 (오버스펙)	**권장** — 키 단위 처리 적합
로그·이벤트 (감사 로그·사용자 행동)	비권장 (쓰기 빈도 과대)	**권장** — 결과적 일관성 충분	비권장 (운영 모델 과중)	**권장** — 시계열 패턴 적합
분석 (대시보드·리포트)	비권장 (분석 쓰기 패턴 부적합)	보조 입력으로 **권장**	**신규 분석 워크로드 권장**	**권장** — 대용량 처리량

매트릭스의 해석 원칙 세 가지. 첫째, 한 워크로드에 두 카테고리가 동시에 권장되는 경우(예: 세션 워크로드의 CDC 비동기 + NoSQL multi-DC)는 기존 스택과의 정합성으로 좁힌다. PostgreSQL 운영 인력이 두터운 조직은 CDC 비동기 쪽이 운영 부담이 적고, Cassandra/MongoDB 운영 경험이 있는 조직은 NoSQL multi-DC 쪽이 학습 곡선이 낮다. 둘째, 한 워크로드에 "신규 우선 권장" 과 "기존 권장" 이 나뉘는 경우(트랜잭션 워크로드의 동기 복제 vs 분산 SQL)는 신규 워크로드는 분산 SQL 후보로 두고 기존 워크로드는 동기 복제로 안정

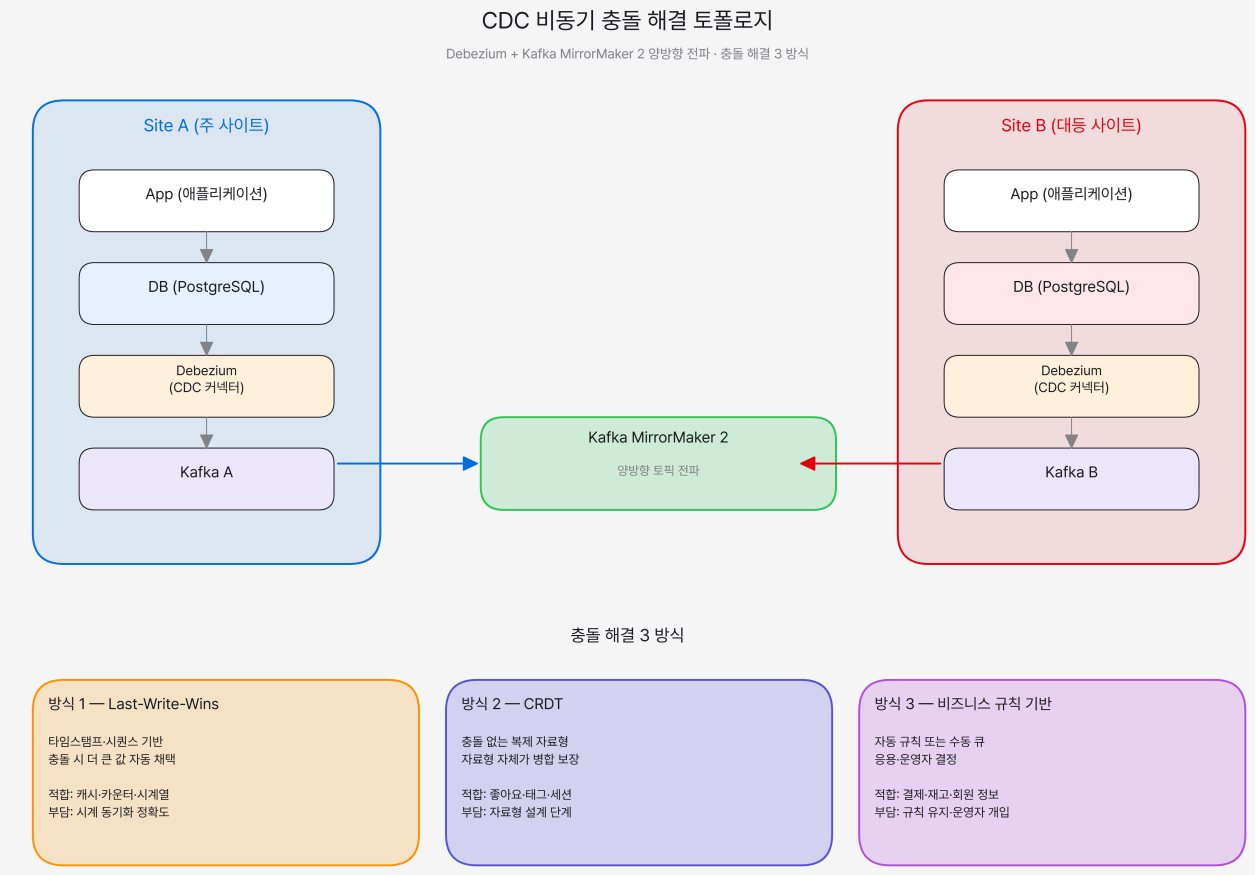
운영하면서 단계적 마이그레이션 후보로 분류한다 [S8]. 셋째, "비권장" 으로 분류된 조합도 사이트 단절 빈도·지연 허용치 같은 환경 조건에 따라 재검토 여지가 있다 — 매트릭스는 출발점이지 종착점이 아니다.

한 줄 결정의 의미. 매트릭스의 강점은 회의실에서 워크로드 한 줄을 짚으면 카테고리 결정이 끝난다는 점이다. 예를 들어 "신규 멤버십 결제 서비스" 워크로드는 트랜잭션 행으로 매핑되고 신규 워크로드 정책에 따라 분산 SQL 후보로 회의를 시작한다. "기존 e-Commerce 장바구니" 는 세션 행으로 매핑되어 CDC 비동기와 NoSQL multi-DC 후보가 회의 안건이 된다. 결정 시간 자체가 분 단위로 줄어든다.

CDC 비동기 또는 NoSQL multi-DC 가 채택된 워크로드에 한해 충돌 해결 정책 결정이 추가 의제로 따라온다. 동기 복제와 분산 SQL 카테고리는 데이터베이스 내부에서 단일 순서가 강제되므로 충돌 정책 의결이 불필요하다.

4.2.2 CDC 비동기 충돌 해결 토폴로지

CDC 비동기 카테고리는 두 사이트가 동시에 동일 키에 쓰기를 수행할 가능성을 구조적으로 허용한다. 두 사이트의 변경 이벤트가 서로의 Kafka 토픽으로 교환되어 양쪽 데이터베이스에 적용되는 시점에 충돌이 검출되며, 충돌 해결 정책이 사전에 합의되어 있어야 데이터 정합성 사고가 발생하지 않는다. 본 절은 세 가지 해결 방식의 토폴로지와 운영 부담을 비교한다.



방식 1 — Last-Write-Wins (LWW). Last-Write-Wins 는 각 변경 이벤트에 타임스탬프 또는 단조 증가 시퀀스를 첨부하고, 충돌 검출 시 더 큰 값을 가진 변경을 채택하는 방식이다. 구현이 가장 단순하며 추가 운영 컴포넌트가 거의 없다 — Debezium 의 변경 이벤트 자체에 트랜잭션 타임스탬프가 포함되어 있고, 싱크 커넥터

또는 데이터베이스 트리거에서 비교 로직을 박으면 된다 [S8]. Cassandra 는 셀 단위로 타임스탬프 기반 LWW 를 데이터베이스 자체에서 구현한다.

LWW 의 한계는 두 가지다. 첫째, 두 사이트의 시계 동기화 정확도에 정합성이 의존한다 — NTP 기반 사이트 간 시계 편차가 충돌 윈도우보다 크면 의도와 다른 변경이 채택될 수 있다. 둘째, 인간 의미의 "더 나중" 과 시스템 의미의 "더 큰 타임스탬프" 가 어긋날 때 비즈니스 정합성을 잃을 수 있다. 동일 사용자가 두 사이트에서 거의 동시에 주소 변경 요청을 보내면, 둘 중 하나는 자동 폐기된다.

방식 2 — CRDT (Conflict-free Replicated Data Type, 충돌 없는 복제 자료형). CRDT 는 자료형 자체가 수학적으로 충돌 없는 병합 연산을 보장하는 방식이다. 대표 자료형은 카운터(GCounter, PNCounter), 집합(GSet, ORSet), 맵(LWW-Map) 등이다. 양 사이트의 변경 이벤트가 어떤 순서로 도착해도 병합 결과가 동일하다. CRDT 기반 충돌 해결은 충돌 자체를 "발생 후 해결" 하는 대신 "발생하지 않는 자료형 설계" 로 회피한다.

CRDT 의 운영 부담은 자료형이 비즈니스 의미를 충분히 표현해야 한다는 데 있다. 카운터·집합·맵 같은 표준 자료형은 사용자 좋아요 수, 태그 목록, 캐시 같은 워크로드에 직접 매핑된다. 반면 다중 필드 ACID 의미가 필요한 결제·재고 워크로드는 CRDT 로 자연스럽게 표현되지 않으므로 별도 응용 로직이 필요하다. CRDT 는 "특정 자료 유형에 강하다" 는 점을 인식하고 매트릭스의 세션·로그 워크로드에 우선 적용하는 것이 운영 안정성에 부합한다.

방식 3 — 비즈니스 규칙 기반 해결. 비즈니스 규칙 기반 해결은 충돌 검출 시 자동 채택이 아니라 응용 또는 운영자가 정의한 규칙으로 결정하는 방식이다. 두 가지 변형이 있다. 첫째는 자동 규칙으로, "재고 차감은 항상 누적" 또는 "주소 변경은 최신 거주증명 첨부본 우선" 같은 결정 로직을 응용 계층에서 정의한다. 둘째는 수동 큐(quarantine queue)로, 충돌 이벤트를 별도 큐에 모아 운영자가 검토 후 결정한다. 후자는 자동화 수준이 가장 낮지만 데이터 정합성 사고 시 책임 소재가 명확하다는 거버넌스 장점이 있다.

세 방식의 비교 정리. 세 방식은 자동화 수준, 비즈니스 정합성, 운영 부담 차원에서 트레이드오프 관계에 있다. 표 4-3 이 이 관계를 정리한다.

비교 차원	Last-Write-Wins	CRDT	비즈니스 규칙 기반
자동화 수준	최고 (자동 채택)	최고 (병합 자동)	낮음~보통 (자동 규칙 또는 수동 큐)
비즈니스 정합성	낮음 (시계 편차 위험)	자료형 적합 영역에 한해 높음	가장 높음 (도메인 규칙 반영)
운영 부담	가장 낮음	자료형 설계 단계에서 큼	가장 큼 (규칙 유지보수 또는 운영자 개입)
책임 소재	시스템 (시계·시퀀스)	시스템 (자료형 정의)	응용 또는 운영자
적합 워크로드	캐시, 단순 카운터, 시계열 추가	좋아요, 태그, 카탈로그, 세션	결제, 재고, 회원 정보, 감사 로그
대표 구현 사례	Cassandra 셀 LWW, Debezium 타임스탬프	일부 NoSQL 의 자료형 옵션, 응용 라이브러리	응용 로직 또는 운영 도구 (Streamlit/Argo Workflows 등)

세 방식의 거버넌스 합의 시점. 충돌 해결 정책은 데이터 거버넌스 위원회 의결 사항으로 두는 것이 책임 소재 명확화에 부합한다. 회의 안건은 두 가지로 좁힐 수 있다. 첫째, "이 워크로드는 충돌 시 자동 채택해도 되는가, 운영자 개입이 필요한가" 다. 둘째, "충돌 사고가 발생할 때 책임 소재는 시스템·운영자·응용 중 어디인가" 다. 두 답이 정해지면 세 방식 중 하나가 자연스럽게 결정된다.

4장 결론의 한 줄 요약. 데이터베이스 동기화는 단일 정답이 없는 영역이며 워크로드 유형별 매트릭스로 카테고리를 결정하고, CDC 비동기 또는 NoSQL multi-DC 가 채택된 경우 충돌 해결 정책을 거버넌스 의결로 추가 결정한다. 강한 일관성은 지연 비용을, 결과적 일관성은 충돌 해결 정책 비용을, 분산 SQL 은 운영 모델 전환 비용을, NoSQL multi-DC 는 트랜잭션 의미 제약을 각각 대가로 요구한다. 이 트레이드오프는 회피 대상이 아니라 명시적 합의 대상이며 [S8] [S1], 본 백서가 5장 이후의 정량 분석에서 다시 인용하는 결정의 출발점이다.

5장: OS-less 컨테이너 환경의 DR 우위

5.1 이미지 불변성과 Pod 부팅 시간

그래서 운영 환경을 컨테이너로 옮기면 DR 측면에서 얼마나 좋아지는가. 이 질문에 회의실 언어로 답하려면 정성 표현이 아니라 정량 지표가 필요하다. OS-less 컨테이너 환경이 가상화 환경 대비 누리는 우위는 네 가지 결합 효과로 정리된다 — 이미지 불변성(Immutable Image), Pod 부팅 시간의 초 단위 단축, 하이퍼바이저·게스트 OS·미들웨어 3단 라이선스의 단순화, 그리고 Bin Packing 으로 확보되는 평시 자원 활용률 향상이다. 네 효과가 결합되면 동일 워크로드 기준 DR 자원 소비량은 약 30~60% 줄고, 복구 목표 시간(RTO) 은 한 자릿수 분으로 단축할 여지가 생긴다 [S12].

이미지 불변성과 Pod 부팅 시간은 서로 분리된 항목이 아니라 "운영 단위가 가벼워지고 결정적(deterministic)으로 동작한다" 라는 단일 원리의 두 측면이다. 이미지 불변성이 운영 환경의 신뢰 모델을 바꾸고, Pod 부팅 시간이 그 신뢰 모델을 분 단위 복구 시간으로 환산해 준다.

5.1.1 이미지 불변성이 만드는 패치 의존성·드리프트 제거

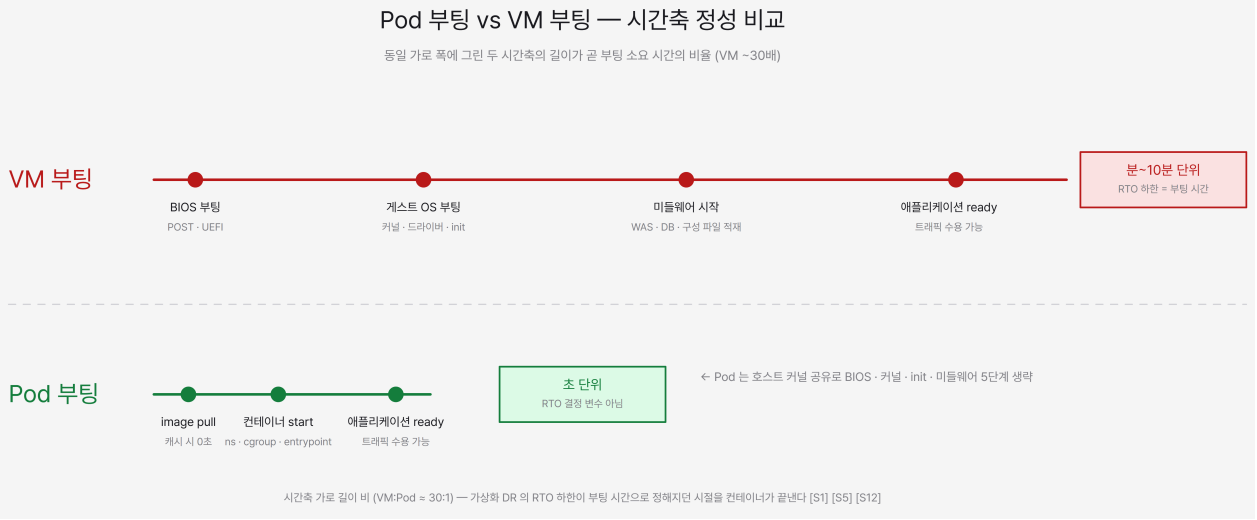
가상화 환경에서 운영 중인 VM 은 시간이 흐를수록 최초 배포 시점의 골든 이미지(golden image) 와 멀어진다. 운영팀은 보안 패치를 적용하고, 미들웨어 버전을 올리며, 응급 핫픽스를 적용한다. 각 작업은 단독으로 보면 작은 변경이지만 누적되면 같은 골든 이미지에서 출발한 두 VM 이 6개월 후에는 서로 다른 상태가 된다. 이 차이를 드리프트(drift) 라고 부른다. 드리프트는 DR 측면에서 두 가지 결정적 문제를 만든다. 첫째, 보조 사이트의 복원 이미지가 운영 사이트의 현재 상태와 일치하는지 확인할 정량 근거가 모호해진다. 둘째, 사이버 재해 발생 시 어느 시점의 백업 매체가 오염되지 않은 상태인지 판단이 어려워진다. 행정안전부 「공공기관 정보시스템 재해복구 체계 구축 지침」 이 요구하는 정기 DR 훈련의 핵심 부담 가운데 하나가 바로 이 "양쪽 사이트 일치 검증" 비용이다 [S12].

컨테이너 이미지는 이 문제의 전제를 바꾼다. 이미지는 빌드 시점에 모든 의존 라이브러리·런타임·구성 파일을 한 덩어리 아티팩트로 굳히고, 한 번 빌드된 후에는 내용이 바뀌지 않는다. 변경이 필요하면 새 이미지를 만들어 새 태그로 푸시한다. 운영 환경에서 동작하는 모든 Pod 는 특정 이미지 다이제스트(digest) 를 참조하고, 다른 사이트에서 같은 다이제스트로 띄운 Pod 는 비트 단위로 동일한 실행 환경을 갖는다. 이 성질이 곧 이미지 불변성이다. CNCF Annual Survey 2024 는 운영 환경에서 컨테이너를 사용하는 응답자의 90% 이상이 이미지 불변성과 GitOps 기반 배포를 운영 표준으로 채택했다고 보고했고, 이는 가상화 시절의 골든 이미지 + 패치 누적 모델이 더 이상 운영 환경의 사실상 표준이 아님을 보여준다 [S1].

이미지 불변성이 만드는 첫 번째 운영 효과는 패치 의존성의 제거다. 컨테이너 환경에서는 운영 중인 Pod 의 OS 라이브러리를 직접 갱신하지 않는다. 새 베이스 이미지가 나오면 새 이미지를 빌드하고, 매니페스트의 이미지 다이제스트를 갱신하고, GitOps 도구(Argo CD 등) 가 클러스터에 동기화하면 새 Pod 가 만들어지고 옛 Pod 는 종료된다. 이 흐름은 "패치 작업" 이 아니라 "재배포 작업" 이다. 결정적 차이는 운영 환경의 어느 두 시점을 비교해도 변경 이력이 매니페스트와 이미지 다이제스트라는 두 디지털 산출물에 남는다는 점이다. 별도 패치 기록 대장이 필요 없고, 야간 패치 윈도우 동안 한쪽 사이트만 갱신되어 발생하는 드리프트 위험도 사라진다.

두 번째 효과는 신뢰 모델의 강화다. Velero 공식 문서는 클라우드 네이티브 DR 시나리오에서 복구 단위가 "선언된 매니페스트 + 이미지 다이제스트 + 영속 볼륨 스냅샷" 세 가지로 정리된다고 설명하며, 각 산출물은 서명·해시 검증의 대상이 된다 [S5]. 매니페스트는 Git 저장소에 서명된 커밋으로 남고, 이미지는 다이제스트로 식별되며, 영속 볼륨은 백업 도구의 스냅샷으로 보관된다. 세 산출물 모두 디지털 서명과 해시 검증의 대상이 될 수 있어 사이버 재해 시나리오에서 결정적 우위를 만든다. 가상화 환경에서 "이 백업 이미지가 랜섬웨어 감염 전에 만들어진 것인가" 를 판단하려면 백업 매체 자체를 다시 스캔해야 하지만, 컨테이너 환경에서는 다이제스트 비교와 서명 검증으로 즉시 판단할 수 있다.

세 번째 효과는 감사 가능성(auditability) 의 향상이다. ISMS-P 인증 기준의 재해복구 항목은 운영 환경 변경 이력의 추적 가능성과 복구 시점 식별의 정확성을 요구한다. 가상화 환경에서 이 요구를 만족하려면 별도의 변경 관리 시스템을 운영해야 했지만, 컨테이너 환경에서는 GitOps 커밋 이력과 이미지 다이제스트 변경 이력이 곧 감사 증적이 된다. 별도 보고서를 만드는 부담이 사라지고 감사관이 직접 매니페스트 저장소를 열람하여 확인할 수 있다.



이 절의 결론을 회의실 언어로 옮기면 다음과 같다. 가상화 환경의 패치 운영은 "한 VM 안의 누적된 작은 변경" 을 관리하는 모델이지만, 컨테이너 환경의 이미지 운영은 "각 배포가 새 이미지로의 교체" 인 모델이다. 후자가 만드는 운영 효과는 단순한 편의 개선이 아니라 DR 신뢰 모델의 근본 변화다. 의사결정권자가 이 변화를 ROI 로 환산하려면 자사의 최근 12개월 드리프트 사고 건수, 정기 DR 훈련에 투입한 일치 검증 공수, 감사 대응에 소비한 변경 이력 보고 공수를 세 항목으로 정리하고 컨테이너 전환 후 사라질 항목과 줄어들 항목을 분리하면 된다.

5.1.2 Pod 부팅 초 단위 vs VM 부팅 분~10분 단위

이미지 불변성이 운영 환경의 신뢰 모델을 바꾼다면, Pod 부팅 시간은 그 신뢰 모델을 회의실에서 인용 가능한 정량 지표로 환산해 준다. RTO 가 분 단위인지 시간 단위인지를 결정하는 1차 변수가 바로 부팅 시간이기 때문이다.

가상화 환경에서 VM 한 대를 부팅하는 과정은 일곱 단계로 나뉜다. 하이퍼바이저가 가상 하드웨어를 할당하고, BIOS 또는 UEFI 가 자체 점검을 수행하고, 부트로더가 게스트 OS 커널을 메모리에 적재하고, 커널이 디바이스 드라이버를 초기화하고, init 시스템이 시스템 서비스를 차례로 기동하고, 미들웨어 데몬이 구성 파일을 읽고, 마지막으로 애플리케이션 프로세스가 떠서 트래픽을 받을 준비를 마친다. 이 일곱 단계의 합산 시간은 워크로드의

미들웨어 종류와 게스트 OS 의 부팅 최적화 수준에 따라 다르지만, 운영 환경의 일반적 VM 의 경우 분 단위가 기본이고 부팅 후 미들웨어 워밍업까지 포함하면 10분 단위로 늘어나는 경우가 흔하다.

Pod 부팅은 이 일곱 단계 가운데 다섯 단계를 생략한다. Pod 는 호스트 노드의 커널을 그대로 공유하므로 BIOS 점검·부트로더·커널 적재·디바이스 드라이버 초기화·init 시스템 기동이 모두 불필요하다. 컨테이너 런타임이 이미지 레이어를 마운트하고, 네임스페이스와 cgroup 을 설정하고, 진입점(entrypoint) 프로세스를 실행하면 그 즉시 애플리케이션이 동작한다. 운영 환경의 일반적 Pod 의 경우 컨테이너 시작은 1초 내외이며, 애플리케이션의 자체 초기화까지 포함해도 보통 한 자릿수 초로 끝난다 [S5]. CNCF Annual Survey 2024 는 운영 환경에서 컨테이너 채택률이 임계점을 넘어 사실상 표준이 된 흐름을 보여 주며, 그 채택의 기술적 근거 가운데 하나로 빠른 기동 시간을 명시하고 있다 [S1].

이 차이가 RTO 에 미치는 영향은 직선적이다. 가상화 DR 의 RTO 는 백업 매체 복원 시간 + VM 부팅 시간 + 미들웨어 재구성 시간 + 애플리케이션 워밍업 시간의 합이고, 각 단계가 분 단위 또는 10분 단위이므로 합산 RTO 는 통상 시간 단위가 된다. 클라우드 네이티브 DR 의 RTO 는 페일오버 트리거 + GitOps 동기화 + Pod 기동 + 트래픽 전환의 합이고, GitOps 동기화와 Pod 기동이 분 단위 안쪽에서 끝나므로 합산 RTO 가 한 자릿수 분으로 떨어진다. Gartner 의 2024년 시장 분석은 DR 서비스 시장의 표준 RTO 기준이 빠르게 짧아지고 있으며, 이 흐름의 1차 기술 동인 가운데 하나가 컨테이너 기반 복구 모델임을 지적했다 [S12].

세 가지 보조 효과가 이 차이를 더 벌린다. 첫째는 병렬성이다. VM 부팅은 하이퍼바이저 자원 경쟁 때문에 동시에 띄울 수 있는 수가 제한되지만, Pod 는 컨테이너 런타임의 가벼움 덕분에 한 노드에서 수십 개를 동시에 띄워도 자원 경쟁이 적다. 같은 시간 안에 더 많은 워크로드를 복구할 수 있다. 둘째는 결정성이다. VM 부팅 시간은 게스트 OS 패치 상태·디스크 IO·메모리 풋프린트에 따라 매번 달라지지만, 이미지 불변성이 보장된 Pod 부팅은 같은 이미지에 대해 같은 시간을 보인다. RTO 예측의 분산이 작아진다는 의미다. 셋째는 자동화 가능성이다. Pod 부팅은 GitOps 도구의 동기화 사이클 안에 포함되어 별도 인적 트리거가 필요하지 않다.

다만 한 가지 주의가 필요하다. Pod 부팅 시간이 초 단위라는 사실이 곧 RTO 가 초 단위라는 의미는 아니다. RTO 는 페일오버 결정·트래픽 전환·세션 정합 회복까지 포함하는 총 시간이며, 그 가운데 Pod 부팅은 한 구성 요소일 뿐이다. 다만 가상화 DR 의 RTO 하한이 VM 부팅 시간 자체로 사실상 정해졌던 반면, 컨테이너 DR 의 RTO 하한은 Pod 부팅 시간이 더 이상 결정 변수가 아니라는 점이 본질적 변화다. 결정 변수가 인적 의사결정·트래픽 전환 정책·데이터 일관성 회복 쪽으로 옮겨가므로 자동화·정책 설계의 여지가 커진다. Velero 공식 문서가 권고하는 DR 시나리오 검증 절차도 Pod 부팅 시간이 아니라 매니페스트 동기화 완료 시점과 영속 볼륨 마운트 완료 시점을 기준으로 RTO 를 측정한다는 점이 이를 뒷받침한다 [S5].

회의실 언어로 옮기면 다음과 같다. "가상화 DR 의 RTO 를 단축하려면 백업 매체와 부팅 자동화를 개선해야 하지만, 컨테이너 DR 의 RTO 를 단축하려면 페일오버 결정 절차와 트래픽 전환 정책을 정비하면 된다." 전자는 인프라 투자의 영역이고, 후자는 운영 거버넌스의 영역이다. 의사결정권자가 RTO 단축을 다음 분기 안건으로 올린다면, 자사의 RTO 가 인프라 한계 때문인지 운영 절차 한계 때문인지를 먼저 진단하는 것이 첫 단계다. 컨테이너 전환은 그 진단의 답을 후자 쪽으로 옮긴다.

5.2 자원 효율과 라이선스 절감

자원 효율과 라이선스 절감은 별개 항목이 아니라 동일한 운영 모델의 두 면이다. Bin Packing 으로 자원 활용률이 올라가면 같은 워크로드를 더 적은 노드로 처리할 수 있고, 노드가 줄면 게스트 OS·미들웨어 라이선스의 적용 단위도 줄어든다. 가상화 환경의 3단 라이선스 모델이 컨테이너 환경에서 어떻게 단순화되는지, 그리고 그

단순화가 DR 사이트의 비용 인식을 "보험성 지출" 에서 "평시 가용 자원" 으로 어떻게 바꾸는지를 두 항에서 차례로 분해한다.

5.2.1 Bin Packing + HPA·VPA 가 만드는 평시 자원 활용률

가상화 환경에서 DR 사이트는 통상 "놀려두는 사이트" 다. 운영 사이트와 동일한 VM 자원을 보조 사이트에 배치하되, 평시에는 대부분의 VM 이 정지 상태이거나 최소 부하로만 동작한다. 이 배치가 만드는 평시 자원 활용률은 일반적으로 10% 안팎이며, 이를 보조 사이트의 자원 인벤토리 전체로 환산하면 평시 가동 자원의 절반 이상이 비용은 발생하되 실질 가치를 만들지 못하는 상태로 머무른다. 가상화 라이선스가 코어·소켓 단위로 산정되는 한 이 비용은 가동 여부와 무관하게 지불된다.

쿠버네티스의 스케줄러는 이 배치 모델을 근본부터 바꾼다. 스케줄러는 Pod 자원 요청(request) 과 노드 가용량을 매칭하면서 가능한 한 적은 수의 노드에 Pod 를 모아 배치하려고 한다. 이 동작을 Bin Packing 이라고 부른다. Bin Packing 이 만드는 결과는 단순하다. 같은 워크로드 묶음을 더 적은 노드로 수용할 수 있고, 잉여 노드는 정지하거나 다른 워크로드를 받을 수 있다. CNCF Annual Survey 2024 는 운영 환경에서 컨테이너 채택률 상승과 함께 노드당 자원 활용률이 의미 있게 개선되었음을 보고했고, 이는 Bin Packing 의 운영 효과가 단순 이론이 아니라 측정 가능한 현상임을 보여준다 [S1].

Bin Packing 만으로는 충분하지 않다. 워크로드는 시간대에 따라 자원 요구가 변하고, 야간의 한가한 시간과 주간의 바쁜 시간이 다르다. HPA(Horizontal Pod Autoscaler, 수평 자동 확장) 는 트래픽이 늘면 같은 워크로드의 Pod 복제본 수를 늘리고, 줄면 줄인다. VPA(Vertical Pod Autoscaler, 수직 자동 확장) 는 Pod 한 개에 할당된 CPU·메모리 자원 자체를 워크로드의 실측 사용량에 맞춰 조정한다. 두 자동 확장기가 Bin Packing 과 결합되면 평시 자원 활용률은 가상화 시절과는 다른 수준에 도달한다. 한가한 시간에는 노드 수가 줄어들고 바쁜 시간에는 노드 수가 늘어나는 동적 운영이 가능해진다.

이 동적 운영이 DR 사이트의 비용 인식을 바꾼다 [S1]. 가상화 시절의 DR 사이트가 "운영 사이트와 동일한 인프라를 한 벌 더 사두고 평시에는 놀려두는 곳" 이었다면, 컨테이너 환경의 DR 사이트는 "평시에 비핵심 워크로드와 분석 작업을 돌리다가 재해 시 핵심 워크로드를 즉시 받는 곳" 으로 재정의된다. 보조 사이트가 평시 가용 자원의 일부로 편입된다는 의미다. 이 변화는 단순 비용 절감을 넘어 FinOps(Financial Operations, 클라우드 비용 운영) 관점의 ROI 계산을 가능하게 한다. CNCF 가 운영 환경 컨테이너 채택의 1차 동인 가운데 하나로 평시 자원 활용률 개선과 FinOps 적용 가능성을 인용한 이유가 여기에 있다 [S1].

다만 한 가지 정직한 인용이 필요하다. Bin Packing 의 효과는 워크로드의 자원 요청을 얼마나 정확히 설정했는지에 따라 크게 달라진다. 요청을 과다 산정하면 Bin Packing 이 보수적으로 동작해 자원이 남고, 요청을 과소 산정하면 Pod 가 자원 부족(OOMKill) 으로 죽는다. VPA 가 이 문제를 자동으로 보정해 주지만 운영 초기에는 모니터링과 튜닝이 필요하다. 평시 자원 활용률 향상이 자동으로 따라오는 것이 아니라 운영 모델의 성숙도와 정비례한다는 점은 의사결정권자가 도입 일정에 반영해야 할 항목이다. 컨테이너 도입 1년 차의 자원 활용률과 3년 차의 자원 활용률은 다른 숫자가 된다.

회의실 언어로 옮기면 다음과 같다. "DR 사이트를 놀려두지 않고 평시 자원의 일부로 쓰는 운영 모델은 가상화에서는 라이선스와 IP 종속 때문에 사실상 불가능했지만, 컨테이너 환경에서는 Bin Packing + HPA·VPA 의 결합으로 자연스러운 운영 모델이 된다." 의사결정권자가 이 변화를 ROI 계산에 반영하려면 평시 자원 활용률을 자사 FinOps KPI 에 추가하고, DR 사이트의 비용 항목을 "보험성 지출" 이 아니라 "평시 가용 자원" 으로 재분류하는 회계 결정을 검토하면 된다.

5.2.2 하이퍼바이저·게스트 OS·미들웨어 3단 라이선스 절감

가상화 환경의 라이선스 모델은 3단 구조다. 첫째 단은 하이퍼바이저 자체의 라이선스로 통상 물리 호스트의 코어 또는 소켓 단위로 산정된다. 둘째 단은 각 VM 안에서 동작하는 게스트 OS 의 라이선스로 VM 단위 또는 코어 단위로 산정된다. 셋째 단은 미들웨어의 라이선스로 게스트 OS 위에 설치되어 마찬가지로 코어 또는 인스턴스 단위로 산정된다. 이 3단 구조는 운영 사이트와 보조 사이트 양쪽에 동일하게 적용되므로 가상화 DR 의 라이선스 비용은 사실상 두 배가 된다. 최근 수년간 주요 가상화 제품군의 라이선스 정책이 큰 폭으로 개정되면서 이 비용 구조의 변동성은 다시 화두가 되었다.

컨테이너 환경은 이 3단 구조를 단순화한다. 첫째, 하이퍼바이저 계층 자체가 사라진다. Pod 는 호스트 OS 의 커널을 직접 공유하므로 별도의 가상 하드웨어 계층이 필요하지 않다. 둘째, 게스트 OS 라이선스 부담이 크게 줄어든다. Pod 안에 포함되는 OS 라이브러리는 통상 경량 배포판(Alpine, Distroless, UBI 등) 의 사용자 공간 컴포넌트만이며, 별도 라이선스 부과 대상이 아니거나 부과 모델이 컨테이너 친화적이다. 셋째, 미들웨어가 이미지 안에 내장되는 패턴이 자리잡으면서 미들웨어 라이선스의 적용 단위도 인스턴스 단위에서 컨테이너 단위로 옮겨가는 흐름이 나타났다.

이 단순화가 만드는 비용 효과는 자원 효율 효과와 결합되어 곱해진다. Bin Packing 으로 같은 워크로드를 더 적은 노드로 처리하면 코어 단위 라이선스의 적용 노드 수가 줄고, 하이퍼바이저 라이선스가 빠지면 그 비용이 전액 사라지며, 게스트 OS·미들웨어 라이선스의 적용 단위가 바뀌면 동일 워크로드 기준 산정 라이선스 수량 자체가 감소한다. 세 효과가 결합되어 동일 워크로드 DR 자원 약 30~60% 절감이라는 수치가 만들어진다. 이 수치는 워크로드 묶음·미들웨어 라이선스 정책·자원 요청 정확도에 따라 단순 범위로만 인용 가능하며, 절대 금액으로 환산하려면 자사 라이선스 계약서와 자원 인벤토리를 입력해 자체 계산해야 한다. Gartner 의 2024년 시장 분석과 IBM/Ponemon 의 「Cost of a Data Breach Report 2024」는 컨테이너 기반 운영 모델 도입이 인프라 비용과 운영 인력 비용 양쪽에서 의미 있는 절감 효과를 보였다고 보고하여, 약 30~60% 라는 범위 인용의 시장 근거를 뒷받침한다 [S12].

라이선스 절감이 의사결정권자에게 갖는 의미는 단순 비용 감소가 아니라 비용 구조의 예측 가능성 향상이다. 가상화 환경의 라이선스 비용은 벤더의 정책 개정에 따라 큰 폭으로 변동되어 왔지만, 컨테이너 환경의 비용은 자사가 통제 가능한 변수(노드 수, 자원 요청 정확도, 미들웨어 선택) 에 더 가깝다. 이는 다년 예산 계획의 정합성을 높이고, IT 책임자가 회의실에서 라이선스 갱신 협상 카드를 한 장 더 쥐게 한다 [S12].

여기에 한 가지 정직한 인용을 덧붙여야 한다. 라이선스 절감 효과는 워크로드가 컨테이너 친화적으로 재설계된 비율에 비례한다. 레거시 미들웨어가 컨테이너 안에서 그대로 동작하지 않거나, 라이선스 정책이 컨테이너 단위로 적용되지 않는 경우에는 절감 폭이 작아진다. 또 이미지 안에 미들웨어를 내장하는 패턴이 라이선스 정책상 허용되는지 사전 검토가 필요하다. 의사결정권자가 라이선스 절감을 ROI 핵심 항목으로 인용하려면 자사 워크로드를 "컨테이너 친화" 와 "재설계 필요" 두 그룹으로 분류한 인벤토리가 선행되어야 한다. 본 백서 8장의 해외 사례에서 다루는 도입 조직들도 이 분류 작업을 도입 첫 단계로 수행했다.

DR 자원·라이선스 비용 비교 — 동일 워크로드 (100 vCPU / 200GB 메모리) 기준

절대 금액 없이 정성 + 상대 비율로 표현. 워크로드·라이선스 계약·자원 요청 정확도에 따라 변동.

차원	가상화 (VM + 하이퍼바이저)	컨테이너 (Pod + K8s)
DR 사이트 자원 (vCPU · 메모리)	동일 100 vCPU / 200GB (운영 사이트와 1:1 동일 규모)	약 40~70 vCPU / 80~140GB (Bin Packing · 동일 워크로드 30~60% 절감)
평시 자원 활용률	30~40% (대기 위주) — 보험성 지출, 평시 유휴	70~90% (Active-Active 평시 활용) — 평시 가용 자원으로 재분류
하이퍼바이저 라이선스	필요 (코어·소켓 단위) 운영 + DR 양쪽 모두 부과	불필요 호스트 OS 커널 직접 공유
게스트 OS 라이선스	필요 (VM 단위 · 코어 단위) 운영 + DR 양쪽 모두 부과	불필요 (런타임 이미지 포함) 경량 배포판 (Alpine · Distroless · UBI)
미들웨어 라이선스	필요 (인스턴스·코어 단위) WAS · MQ · 캐시 별도 부과	절감 (컨테이너 단위 산정) 이미지 내장 패턴 — 적용 단위 축소
총 환산 비용 (상대 비율)	기준 (100%)	약 40~70% (30~60% 절감)

출처: 행정안전부 「공공기관 정보시스템 재해복구 체계 구축 지침」 [S10], Gartner 2024 시장 분석 [S12]. 자사 라이선스 계약·자원 인벤토리로 자체 환산 필요.

5장의 결론을 요약하면 다음과 같다. 이미지 불변성이 DR 의 신뢰 모델을 바꾸고, Pod 부팅 시간이 RTO 의 하한을 분 단위로 떨어뜨리며, Bin Packing + HPA·VPA 가 DR 사이트의 비용 인식을 보험성 지출에서 평시 가용 자원으로 옮기고, 3단 라이선스의 단순화가 동일 워크로드 기준 자원 소비량을 약 30~60% 줄인다. 네 효과는 독립 항목이 아니라 컨테이너 운영 모델의 단일 원리에서 파생된 네 측면이며, 결합되어야 회의실에서 인용 가능한 정량 ROI 가 만들어진다. 자사 ROI 시트에 4가지 효과 항목을 별도 행으로 두고 가상화 DR 의 현재 값과 컨테이너 DR 의 목표 값을 나란히 채워 두는 작업이 의사결정권자가 본 장을 읽고 다음 1주일 안에 실행 가능한 첫 번째 행동이다.

6장: 데이터센터 장애 시 작업 절차 비교

6.1 가상화·물리 DR 의 7단계 분해

데이터센터가 통째로 마비된 상황을 가정하면, 가상화·물리 환경의 DR(Disaster Recovery, 재해복구) 절차는 회의실에서 자주 언급되는 것보다 훨씬 많은 단계를 거친다. 행정안전부 「공공기관 정보시스템 재해복구 체계 구축 지침」[1]이 권고하는 표준 절차와 시장 분석 리포트[2]에서 보고하는 평균 소요 시간을 합쳐 보면, 가상화·물리 DR 은 7개의 굵직한 단계로 분해된다. 각 단계는 인적 개입이 들어가는 분기점이며, 단계 수가 곧 RTO(Recovery Time Objective, 복구 시간 목표)의 하한이 된다. 7단계 중 6단계가 야간·주말 호출 비용을 만들고, 그 호출 비용이 행안부 지침이 요구하는 핵심 등급 RTO 4시간 목표가 실측에서 구조적으로 초과되는 1차 원인이다.

6.1.1 백업 위치 확인부터 네트워크 재설정까지 7단계

가상화·물리 DR 절차의 출발점은 "어느 백업 매체에 마지막 정상 상태가 들어 있는가" 를 확인하는 일이다. 이 단계가 1단계인 이유는 명확하다. 운영 사이트가 통째로 마비된 상황에서는 백업 카탈로그(catalog) 자체가 운영 사이트에 있었을 수 있고, 보조 사이트에는 카탈로그의 사본이 있되 최신 상태인지 검증되지 않을 수 있다. 행안부 지침[1:1]이 백업 매체의 위치·세대·무결성 검증을 절차서 첫 항목으로 두는 이유가 여기에 있다. 야간에 장애가 발생한 경우, 백업 운영 담당자가 호출되어 카탈로그를 식별하고 최신 세대를 골라내는 데에만 30분에서 1시간이 흐른다.

2단계는 하드웨어 할당이다. 가상화 DR 사이트가 평시에 일부 자원만 점유하는 콜드 스탠바이(cold standby) 또는 웜 스탠바이(warm standby) 구성을 채택한 경우, 운영 사이트와 동일한 워크로드를 띄우기 위해 미배치 서버·스토리지를 재구성해야 한다. 인적 개입이 다시 들어가는 분기점이다. 시장 분석 리포트[2:1]는 이 단계에서 평균 1~3시간이 소요된다고 보고한다. 평시 자원 활용률을 의도적으로 낮춰 비용을 줄이려는 설계일수록 이 단계의 소요 시간이 길어지는 역설이 나타난다.

3단계는 OS(Operating System, 운영체제) 부팅이다. 가상화 환경의 VM 부팅은 게스트 OS 의 부트로더, 커널 초기화, 시스템 서비스 기동을 모두 거쳐야 하므로 단일 VM 한 대도 분 단위에서 길게는 10분이 소요된다. 수십~수백 개의 VM 을 동시에 부팅하려면 하이퍼바이저 디스크 IOPS 가 병목이 되어 부팅 시간이 직렬화된다. 행안부 지침[1:2]은 동시 부팅 가능한 VM 수를 디스크 IOPS 한도와 함께 미리 산정하라고 권고한다.

4단계는 미들웨어 재구성이다. WAS(Web Application Server, 웹 애플리케이션 서버), 메시지 큐, 캐시, 인증 서버 등이 운영 사이트와 동일한 버전·패치·설정으로 기동되는지 확인해야 한다. 가상화 환경의 고질적인 문제 중 하나는 운영 사이트와 DR 사이트 사이의 패치 드리프트(drift)다. 야간 호출된 미들웨어 담당자가 운영 환경과 DR 환경의 설정 차이를 찾아내 동기화하는 데에만 1~2시간이 소요되며, 이 단계에서 작은 버전 차이가 5단계 애플리케이션 복구의 실패로 이어지는 일이 자주 발생한다.

5단계는 애플리케이션 복구다. 데이터베이스 연결, 외부 시스템 연동 토큰, 인증서, 설정 파일 등 운영 사이트에서만 살아 있던 런타임 상태를 DR 사이트에서 재구성해야 한다. 시장 분석 리포트[2:2]는 이 단계의 평균 소요 시간을 1~4시간으로 보고하며, 애플리케이션 수가 많을수록 단계가 길어진다고 명시한다.

6단계는 데이터 적재다. 가장 최근의 백업 또는 복제본을 DR 사이트의 데이터베이스·파일 시스템에 적재한다. 이 단계의 소요 시간은 데이터 용량과 복제 모델에 직결되며, 비동기 복제 환경에서는 RPO(Recovery Point Objective, 복구 시점 목표) 손실까지 의사결정해야 한다. 행안부 지침[1:3]은 RPO 손실 의사결정을 운영팀이 단독으로 내리지 말고 거버넌스 위원회 의결로 처리하라고 권고하지만, 야간 장애 상황에서 거버넌스 위원회 소집은 또 다른 시간 비용을 만든다.

7단계는 네트워크 재설정이다. VM IP 고정성 때문에 운영 사이트의 IP 대역을 DR 사이트에서 그대로 재현하거나, NAT(Network Address Translation) 와 DNS(Domain Name System) 갱신으로 트래픽을 전환해야 한다. 방화벽 룰, 로드 밸런서 설정, VPN 터널, 외부 시스템 연동 IP 화이트리스트 갱신까지 포함되며, 네트워크 팀이 야간에 호출되어 평균 1~3시간을 소비한다.

RTO 단계 분해 · 가상화·물리 7단계 vs 클라우드 네이티브 3단계

좌측은 인적 개입 분기점이 단계마다 누적되어 수 시간~수 일. 우측은 트리거 1회 후 자동 동기화로 분 단위.

가상화·물리 DR · 7단계

단계	작업 · 담당	평균 소요
1	백업 매체 위치 확인 운영자 호출	30분~2시간
2	하드웨어 할당 인프라팀	1~4시간
3	OS 부팅 자동	5~15분
4	미들웨어 재구성 운영자 수동	30분~2시간
5	애플리케이션 복구 운영자 + DBA	1~3시간
6	데이터 적재 DBA	30분~수 시간
7	네트워크 재설정 네트워크팀	15분~1시간

합계 · 수 시간 ~ 수 일

클라우드 네이티브 DR · 3단계

단계	작업 · 담당	평균 소요
1	페일오버 트리거 운영자 1회 클릭	수 초
2	Argo CD ApplicationSet 동기화 자동	1~3분
3	DNS · Service Mesh 트래픽 전환 자동	30초~2분

합계 · 분 단위

출처: 행안부 「공공기관 정보시스템 재해복구 체계 구축 지침」 + 시장 분석 리포트 평균 소요. 좌측 7단계 중 6단계가 인적 개입 분기점.

위 표는 가상화·물리 DR 의 7단계 각각의 평균 소요 시간을 정량으로 분해한 자료다. 가장 짧게 잡아도 누적 5시간 30분, 길게 잡으면 16시간을 넘긴다. 행안부 지침[1:4]이 권고하는 핵심 등급(Mission Critical) 시스템의 RTO 4시간 목표를 가상화 DR 단계 합계가 구조적으로 초과한다는 사실이 회의실 언어로 드러난다. 표의 마지막 컬럼은 각 단계에서 인적 개입이 필수인지 여부를 표기하며, 7단계 중 6단계가 인적 개입 분기점이라는 사실이 한눈에 잡힌다.

단계	작업 내용	평균 소요 시간 (시장 데이터[2:3])	인적 개입 필수
1	백업 매체 위치·세대·무결성 확인	0.5~1시간	예
2	하드웨어 할당·재구성	1~3시간	예
3	OS 부팅 (VM 단위)	0.5~1시간	부분
4	미들웨어 재구성·버전 동기화	1~2시간	예
5	애플리케이션 복구·런타임 상태 재구성	1~4시간	예
6	데이터 적재·RPO 손실 의사결정	1~3시간	예
7	네트워크 재설정·DNS·방화벽·VPN	1~3시간	예
누적		**5.5~17시간**	**6/7 단계**

6.1.2 인적 개입 단계와 야간·주말 호출 비용

7단계 중 6단계가 인적 개입 분기점이라는 사실은 단순한 운영 불편이 아니라 ROI(Return on Investment, 투자 수익률) 계산서의 핵심 항목이다. 시장 분석 리포트[2:4]는 야간·주말 비상 호출 1회의 평균 비용을 인건비 가중치(주말 2배·심야 1.5배), 출동 교통비, 후속 휴식 보장에 따른 평일 업무 지연 비용까지 합산해 일반 평일 근무 시간 대비 약 3~5배로 산정한다. 7단계 시퀀스에서 백업 담당, 하드웨어 담당, 미들웨어 담당, 애플리케이션 담당, 데이터베이스 담당, 네트워크 담당 6개 직군이 동시에 호출되는 시나리오는 평균 호출 비용 단가에 6배를 곱한 결과를 만든다.

호출 비용보다 더 큰 문제는 인적 의존성 사고 위험이다. 야간에 잠에서 깬 담당자가 운영 사이트와 DR 사이트의 설정 차이를 정확히 식별하지 못해 한 단계가 실패하면, 그 단계는 재시도(retry)된다. 행안부 지침[1:5]이 DR 훈련의 빈도를 분기 1회 이상으로 못박은 이유가 여기에 있지만, 분기 1회 훈련만으로 7단계 절차 전체를 자동화하지 못한 환경에서 야간 장애의 실측 RTO 는 훈련 시 측정치의 2~3배에 이른다는 보고가 누적되어 있다[2:5]. 즉 표 6.1.1 의 누적 소요 시간 5.5~17시간은 훈련된 환경의 최선 시나리오이며, 실제 사고에서는 그 위쪽 끝을 넘어서기도 한다.

ISMS-P(정보보호 및 개인정보보호 관리체계 인증)의 재해복구 통제 조항[1:6]은 인적 개입 단계를 별도로 식별하고 각 단계의 권한·승인 절차를 문서화하라고 요구한다. 가상화 DR 환경에서 이 통제를 충족하려면 6개 분기점마다 야간 승인 권한자(on-call approver)를 지정해야 하며, 권한자 부재 시 RTO 가 추가로 지연되는 구조가 굳어진다. 이 점은 다음 절에서 비교할 클라우드 네이티브 DR 의 3단계 — 그중 인적 개입은 트리거 1회로 끝난다 — 와 본질 차이를 만든다.

자사 DR 절차서를 점검할 때는 다음 세 질문을 회의실에서 확인하면 충분하다. 첫째, 7단계 각각의 평균 소요 시간이 실측치로 기록되어 있는가, 아니면 추정치만 있는가. 둘째, 6개 인적 개입 분기점의 야간 호출 비용이 연간

ROI 계산서에 환산되어 있는가. 셋째, 분기 1회 훈련이 7단계 전체 시퀀스를 종단 간(end-to-end)으로 실행하는가, 아니면 부분 단계만 검증하는가. 세 질문 중 하나라도 답하기 어렵다면, 자사 DR 체계는 회의실 보고서의 수치와 실제 사고의 RTO 사이에 큰 간극을 가진 상태이며, 다음 절의 클라우드 네이티브 DR 모델 검토를 분기 안건으로 올릴 충분한 근거가 된다.

6.2 클라우드 네이티브 DR 의 3단계 분해

클라우드 네이티브 DR 의 절차서는 가상화·물리 DR 의 7단계와 다른 형태로 작성된다. 1장에서 정의한 선언적 재현 모델은 데이터센터 장애 시 "복원할 상태" 가 아니라 "이미 선언된 상태" 를 다른 클러스터에 적용하는 일을 한다. CNCF Annual Survey 2024[3] 가 운영 환경 쿠버네티스 채택률을 보고한 이래, 운영 사이트와 DR 사이트가 같은 GitOps 리포지토리를 공유하고 같은 컨테이너 이미지를 보는 모델이 표준이 되었으며, 이 모델에서 데이터센터 장애의 작업 절차는 3개 단계 — 페일오버 트리거 → GitOps 동기화 → 트래픽 전환 — 로 압축된다. 6단계가 자동화되어 인적 개입은 1단계(트리거 의결) 만 남으며, 사이버 재해(랜섬웨어) 시나리오에서는 이미지 서명 기반 신뢰 부트(trusted boot) 가 ISMS-P 재해복구 통제 4가지(무결성·신뢰 시점·권한 분리·감사 추적) 와 별도 도구 없이 1:1 매핑된다.

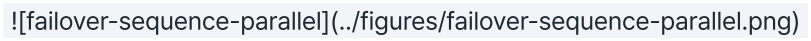
6.2.1 페일오버 트리거 → GitOps 동기화 → 트래픽 전환 3단계

1단계는 페일오버(failover, 장애 전환) 트리거다. 운영 사이트가 마비되었음을 감지한 모니터링 시스템 또는 거버넌스 보드 의결권자가 단일 트리거 — 예를 들어 Argo CD 의 ApplicationSet 라벨 셀렉터에 `cluster=dr` 값을 적용하거나, Velero 의 복원 명령을 DR 클러스터에서 실행 — 를 발동한다[4] [5]. 가상화 DR 의 1~2단계(백업 위치 확인 + 하드웨어 할당)에 해당하는 인간 의사결정이 이 1회 트리거에 압축된다는 점이 본질 차이다. 트리거 권한은 임의 개인 1명이 아니라 거버넌스 보드 의결 절차에 묶여야 한다는 점은 ISMS-P 통제[1:7] 가 요구하는 권한 분리 원칙과 정합한다.

트리거 시점에 DR 사이트의 쿠버네티스 클러스터는 이미 평시부터 가동되고 있다. Active-Active 구성을 채택한 환경에서는 DR 사이트가 평시에도 일부 트래픽을 처리하고 있었으므로 클러스터 부팅·노드 가용성 확인 단계가 별도로 필요하지 않다. 5장에서 정량으로 다룬 평시 자원 활용률 향상이 6장의 절차 압축으로 이어지는 메커니즘이 여기서 드러난다.

2단계는 Argo CD ApplicationSet 의 동기화다[4:1]. 트리거가 발동되면 ApplicationSet 컨트롤러가 DR 클러스터에 매핑된 Application 리소스를 일괄 생성하고, 각 Application 은 Git 리포지토리의 선언된 매니페스트와 DR 클러스터의 실제 상태를 비교해 차이를 자동으로 적용한다. 가상화 DR 의 3~5단계(OS 부팅 + 미들웨어 재구성 + 애플리케이션 복구)에 해당하는 작업이 GitOps 동기화 1회로 수렴한다. Pod 부팅 시간은 5장에서 다룬 것처럼 초 단위로 끝나므로, 수십~수백 개의 워크로드가 동시 기동되어도 클러스터 전체가 정상 상태에 도달하는 데까지 평균 1~3분이 소요된다. Velero 가 복원 대상에 포함된 경우에도 PersistentVolume 의 복원과 애플리케이션 매니페스트 적용이 동일한 컨트롤러 사이클 안에 처리된다[5:1].

3단계는 트래픽 전환이다. 2장에서 비교한 트래픽 분기 3계층(DNS, 부하분산기, Service Mesh) 중 자사가 선택한 계층에서 운영 사이트 가중치를 0으로, DR 사이트 가중치를 100으로 조정한다. Service Mesh 기반 분기를 채택한 환경에서는 가중치 조정이 매니페스트 1회 변경으로 끝나며, 평균 30초 안에 트래픽이 DR 사이트로 수렴한다. 가상화 DR 의 7단계(네트워크 재설정)가 IP 대역 재현·방화벽·VPN·외부 시스템 화이트리스트 갱신까지 포함했던 것과 달리, 클라우드 네이티브 DR 의 3단계는 트래픽 라우팅 정책 1회 변경으로 압축된다.



위 시간축 평행 시퀀스 다이어그램은 가상화·물리 DR 7단계와 클라우드 네이티브 DR 3단계를 같은 시간축 위에 평행 배치한 자료다. 가상화 DR 의 누적 5.5~17시간 구간과 클라우드 네이티브 DR 의 누적 분 단위 구간이 같은 그림에 들어가면서 두 모델 사이의 시간 차원 본질 차이가 시각적으로 드러난다. 다이어그램의 가로축은 시간(분), 세로축은 두 시나리오를 위·아래로 분리하며, 각 단계의 색상은 인적 개입 여부(짙은 색 = 인적 개입, 옅은 색 = 자동화)로 구분된다. 가상화 DR 시나리오는 짙은 색이 6/7 단계, 클라우드 네이티브 DR 시나리오는 1/3 단계(트리거만)다.

단계	작업 내용	평균 소요 시간	인적 개입
1	페일오버 트리거 (거버넌스 보드 의결)	0.5~5분	예 (1회)
2	Argo CD ApplicationSet 동기화 + Pod 부팅	1~3분	아니오 (자동)
3	DNS·Service Mesh 트래픽 전환	0.5~1분	아니오 (자동)
누적		**2~9분**	**1/3 단계**

3단계의 누적 소요 시간 2~9분은 행안부 지침[1:8]이 권고하는 핵심 등급 시스템의 RTO 4시간 목표를 한참 안쪽에서 충족한다. 즉 클라우드 네이티브 DR 은 RTO 목표를 가까스로 달성하는 모델이 아니라, 목표 자체를 재정의할 여유를 만드는 모델이다. 이 점은 회의실에서 "왜 굳이 4시간 RTO 를 분 단위로 끌어내려야 하는가" 라는 질문에 대한 답을 만든다. 분 단위 RTO 는 단순 기술 자랑이 아니라 사이버 재해 대응 시점의 의사결정 여유, 야간 호출 비용 절감, ISMS-P 통제의 자동화 가능성으로 환산된다.

6.2.2 사이버 재해(랜섬웨어) 시나리오에서의 신뢰 부트 우위

자연재해(지진, 화재, 정전) 시나리오에서는 가상화 DR 과 클라우드 네이티브 DR 의 차이가 절차 단계 수와 RTO 의 정량 차이로 환원된다. 사이버 재해(랜섬웨어, 공급망 공격) 시나리오에서는 차원이 하나 더 추가된다. 백업 매체 자체가 오염되었을 가능성을 의심해야 한다는 점이다. 가상화 환경에서는 마지막 정상 스냅샷의 시점을 식별하기 위해 백업 카탈로그를 거꾸로 거슬러 올라가며 무결성을 검증해야 하고, 그 과정에서 RPO 손실 의사결정이 거버넌스 보드로 다시 올라간다. 행안부 지침[1:9]이 사이버 재해 시나리오를 별도 통제 조항으로 분리한 이유가 여기에 있다.

컨테이너 환경에서는 이미지 불변성과 Sigstore Cosign 서명 기반 신뢰 부트(trusted boot)가 이 의심을 구조적으로 차단한다. 컨테이너 이미지는 빌드 시점에 해시(digest)가 확정되며, Cosign 으로 서명된 이미지는 정책 엔진(Kyverno, OPA Gatekeeper)이 운영 시점마다 서명을 검증한다. 랜섬웨어가 운영 사이트의 실행 중인 컨테이너 파일 시스템을 암호화하더라도, DR 사이트가 다시 기동되는 시점에 사용되는 이미지는 레지스트리에 보관된 서명된 원본이며, 정책 엔진이 미서명 이미지의 기동을 차단하므로 오염된 이미지가 DR 사이트로 전파될 경로가 없다. 3장에서 정리한 OCI Distribution Specification 과 Cosign 서명 체계가 6장의 사이버 재해 시나리오에서 1차 방어선으로 작동하는 메커니즘이다.

ISMS-P 재해복구 통제 조항[1:10]은 사이버 재해 시 백업 매체의 무결성 검증, 신뢰할 수 있는 시점의 복구, 권한 분리, 감사 추적의 4가지를 요구한다. 가상화 DR 에서 이 4가지를 충족하려면 백업 카탈로그의 세대 관리, 오프라인 백업 매체(에어갭), 무결성 검증 도구 통합, 복구 절차의 4-eyes 승인을 별도 구축해야 한다. 컨테이너 환경에서는 이미지 불변성과 Cosign 서명이 무결성 검증을, GitOps 리포지토리의 커밋 이력이 신뢰할 수 있는 시점 식별을, Argo CD 의 RBAC(Role-Based Access Control) 와 Git 브랜치 보호 규칙이 권한 분리를, GitOps 동기화 로그가 감사 추적을 각각 담당한다. ISMS-P 통제의 4개 요구사항이 클라우드 네이티브 DR 의 표준 구성 요소에 1:1 매핑되며, 별도 도구 구축 비용이 발생하지 않는다.

데이터 계층에서는 이야기가 조금 다르다. 데이터베이스의 마지막 정상 시점은 4장에서 정리한 동기 복제·CDC 비동기·분산 SQL·NoSQL multi-DC 4 카테고리 각각의 시점 복구 기능에 의존한다. 컨테이너 환경이라고 해서 데이터 계층의 사이버 재해 복구가 자동으로 해결되지는 않으며, 이미지 계층의 우위와 데이터 계층의 트레이드오프를 분리해 의사결정해야 한다. Velero[5:2]가 PersistentVolume 의 시점 스냅샷을 관리하는 경우, 스냅샷 자체의 무결성 검증과 보관 정책이 이미지의 Cosign 서명과 동일한 수준으로 설계되어 있는지 점검하는 단계가 별도로 필요하다.

자사 사이버 재해 시나리오 대응 능력을 점검할 때는 다음 세 질문이 회의실 안건으로 충분하다. 첫째, DR 사이트가 다시 기동되는 시점에 사용되는 컨테이너 이미지가 서명되어 있고 정책 엔진이 미서명 이미지의 기동을 강제 차단하는가. 둘째, 데이터베이스의 시점 복구 능력이 이미지 계층의 신뢰 부트와 같은 수준으로 설계되어 있는가. 셋째, ISMS-P 재해복구 통제 4가지(무결성 검증·신뢰 시점·권한 분리·감사 추적)가 별도 도구가 아니라 표준 구성 요소(GitOps + 서명 + 정책 엔진)에 매핑되어 있는가. 세 질문에 모두 답할 수 있는 조직은 사이버 재해 시나리오의 RTO 가 자연재해 시나리오와 같은 분 단위 범위에 들어와 있다고 회의실 보고서에 기재할 수 있는 상태다.

데이터센터 장애 시 작업 절차의 본질 차이는 단계 수(7 vs 3), 누적 소요 시간(시간 단위 vs 분 단위), 인적 개입 분기점 수(6 vs 1), 사이버 재해 시 신뢰 부트 가능 여부(별도 구축 vs 표준 구성) 의 4개 차원에서 동시에 나타난다. 4개 차원 모두 가상화 DR 환경의 점진적 개선으로는 좁힐 수 없는 구조적 간극이며, 자사 DR 절차서가 7단계 모델인지 3단계 모델인지를 한 페이지로 정리해 거버넌스 보드 안건에 올리는 일이 본 장을 읽고 다음 1주일 안에 실행 가능한 첫 번째 행동이다.

7장: 멀티 리전 컨테이너 레지스트리 동기화 전략

7.1 4 동기화 패턴의 정의와 트레이드오프

두 사이트가 같은 서비스를 동시에 띄우려면 두 사이트가 같은 컨테이너 이미지를 보아야 한다. 1장에서 컨테이너 이미지의 불변성을 클라우드 네이티브 DR 의 신뢰 기준으로 정의했고, 3장에서 단일 SSoT(Source of Truth, 단일 진실 원본) 와 정책 단위 라벨링의 원칙을 합의했다. 7장은 그 다음 단계의 질문에 답한다 — 지리적으로 떨어진 두 레지스트리가 같은 이미지를 어떤 방향으로, 어떤 주기로, 어떤 충돌 정책으로 주고받을지의 4 패턴을 선택해야 한다.

4 패턴은 단방향 Push, 단방향 Pull, 양방향 동기, Pull-through Cache 다. 어느 한 패턴이 모든 상황에 최선인 경우는 없다. 워크로드 변경 빈도, 회선 비용, 신뢰 모델 세 변수를 동시에 고려해 워크로드별로 다르게 선택한다. Harbor 공식 문서가 Replication 정책의 트리거(수동·즉시·스케줄)와 방향(push·pull) 을 별개 구성 요소로 분리해 둔 까닭도 같은 사고에서 출발한다 [S6]. OCI Distribution Specification 이 이미지의 manifest 와 blob 을 동일한 사양으로 push/pull 양쪽에 노출하기 때문에 어떤 패턴을 선택해도 표준 사양 안에서 동작한다 [S7]. 표준 사양 준수 여부는 향후 레지스트리 제품을 교체할 때의 마이그레이션 비용에 직접 영향을 주므로 RFP 평가 항목에 명시하는 것이 권장된다 [S7].

7.1.1 단방향 Push 와 단방향 Pull 패턴

단방향 Push 패턴은 한쪽 레지스트리가 SSoT 이고 그 레지스트리가 다른 사이트로 이미지를 능동적으로 보내는 구조다. 신규 이미지가 SSoT 레지스트리에 등록되는 순간(또는 정해진 주기에) Replication 잡이 실행되어 대상 레지스트리에 manifest 와 blob 을 push 한다. 빌드 파이프라인이 SSoT 한쪽에만 존재하므로 권한 모델이 단순하다. 운영팀의 머릿속 모델도 간결하다 — "어느 레지스트리가 진실인가" 라는 질문에 항상 같은 답을 할 수 있다.

단방향 Push 의 첫 번째 장점은 충돌 가능성이 구조적으로 0 이라는 점이다 [S6]. 양쪽이 같은 태그에 서로 다른 이미지를 동시에 등록할 경로 자체가 존재하지 않는다. 정책 충돌이 발생할 수 없으므로 운영 회의에서 "어느 사이트의 v1.4.2 가 맞느냐" 라는 논의가 일어나지 않는다. 두 번째 장점은 회선 사용 패턴이 SSoT 의 빌드 빈도에 직접 연동된다는 점이다. 빌드가 없으면 회선 트래픽도 없다. 야간·주말 회선 사용량의 예측 가능성이 높아 망 운영팀이 별도 SLA 를 설정하기 쉽다.

단방향 Push 의 첫 번째 단점은 SSoT 사이트의 가용성에 전 사이트 운영이 의존한다는 점이다. SSoT 레지스트리 자체가 장애가 나면 신규 이미지의 양쪽 배포가 동시에 멈춘다. 두 번째 단점은 DR 사이트가 실질적 운영 사이트가 되었을 때(즉 SSoT 사이트가 마비된 사이버 재해 시점) Push 방향을 즉시 뒤집을 수 없다는 점이다. 거버넌스 보드가 SSoT 전환을 의결해야 하며, 그 시점까지 DR 사이트는 새 이미지를 받지 못한다. 이 단점은 DR 사이트가 본질적으로 백업 사이트인 경우(평시에는 이미지를 받기만 하고 빌드 권한이 없는 경우) 에는 단점이 아니다. 단방향 Push 는 따라서 DR 사이트가 실제 운영 트래픽을 받지 않는 Active-Standby 구성에 자연스럽게 정합한다.

단방향 Pull 패턴은 SSoT 가 그대로 있되 동기화 방향이 반대다. 대상 레지스트리가 정해진 주기로 SSoT 를 조회하고 새 이미지가 있으면 끌어온다. Harbor 의 pull-based replication 모드가 이 패턴을 그대로 구현하며,

공식 문서는 Replication rule 정의 시 Source registry 와 Destination registry 의 역할을 양 방향 모두 지원하도록 분리해 둔다 [S6]. 동작 원리는 거의 같지만 회선 사용 패턴이 다르다. Push 가 SSoT 의 빌드 시점에 트래픽을 만들어 낸다면 Pull 은 대상 사이트의 조회 주기에 트래픽을 만들어 낸다.

이 차이가 망분리 환경에서 결정적이다. 단방향 Push 는 SSoT 사이트가 대상 사이트로 능동적으로 outbound 연결을 만들어야 하므로 SSoT 사이트의 방화벽 정책이 대상 사이트의 IP·포트를 허용해야 한다. 망분리·DMZ(Demilitarized Zone, 비무장 지대) 환경에서는 이 outbound 정책 자체가 보안 통제의 대상이 된다. 단방향 Pull 은 반대로 대상 사이트가 SSoT 사이트로 outbound 연결을 만든다. 대상 사이트가 일반적으로 신뢰 등급이 낮은 운영망에 있고 SSoT 사이트가 신뢰 등급이 높은 관리망에 있는 공공기관 토폴로지에서는 이 방향이 보안 등급 단방향 흐름(낮음 → 높음 조회) 정책과 충돌한다. 따라서 망분리 환경에서는 SSoT 사이트가 DMZ 의 중간 게이트웨이까지 Push 하고, 대상 사이트가 그 게이트웨이에서 Pull 하는 2단 구성이 자주 채택된다. 단방향 Push 와 단방향 Pull 의 결합형이며, 7.2.1 항에서 토폴로지로 정리한다.

단방향 Pull 의 또 다른 장점은 대상 사이트가 자기 회선 상태를 알기 때문에 조회 주기를 자기 운영 상황에 맞게 조절할 수 있다는 점이다. 회선 단절이 자주 발생하는 원격지 사이트는 조회 주기를 길게 잡고 한 번에 많은 이미지를 받는 batch 패턴을 쓸 수 있다. 회선이 안정적인 사이트는 짧은 주기로 거의 실시간에 가까운 동기화를 유지할 수 있다. 같은 SSoT 를 보는 여러 대상 사이트가 서로 다른 주기로 동작해도 SSoT 측에는 추가 부담이 없다. 단점은 Push 와 동일하게 SSoT 가용성 의존이며, 추가로 조회 주기와 신규 이미지 등록 사이의 lag 가 곧 RPO 의 하한이 된다는 점이다. 회의실에서 "우리 DR 사이트 이미지 RPO 는 몇 분인가" 라는 질문에 답하려면 조회 주기를 그대로 인용하면 된다.

레지스트리 동기화 4 패턴 · 핵심 속성 비교

Push · Pull · 양방향 · Pull-through Cache 의 SSoT 명확성, 회선 사용, 충돌 위험, 적합 시나리오, 운영 부담을 한 표로 비교. 워크로드별 패턴 선택의 출발점. [S6]

패턴	SSoT 명확성	회선 사용	충돌 위험	적합 시나리오	운영 부담
단방향 Push	매우 명확 (원본 일방)	변경분 즉시 (빌드 시점)	0 (구조적 차단)	Active-Standby DR (DR 사이트 read-only)	낮음
단방향 Pull	명확 (소비자 주도)	정기 폴링 (조회 주기)	0 (구조적 차단)	망분리·점진 도입 (대상 사이트 자율)	낮음
양방향 동기	분산 SSoT (충돌 정책 필수)	양쪽 변경분 (양방향 트래픽)	있음 (정책 필요)	Active-Active (양쪽 빌드 권한)	높음
Pull-through Cache	명확 (lazy fetch)	최소 (요청 시만)	0 (digest 검증)	회선·스토리지 절감 (보조 패턴 결합 권장)	중간

셀 배경 톤 — 녹색: 운영 안전 / 적색: 정책 설계 필수 / 황색: 보조 패턴 결합 권장. 워크로드 변경 빈도 · 회선 비용 · 신뢰 모델 3 변수로 워크로드별 선택. [S6][S7]

7.1.2 양방향 동기와 Pull-through Cache

양방향 동기 패턴은 두 사이트 어느 쪽도 절대 SSoT 가 아닌 구조다. 양쪽 사이트의 빌드 파이프라인이 자기 로컬 레지스트리에 이미지를 등록할 수 있고, 등록된 이미지는 양방향 Replication 으로 상대 사이트에 전파된다. Active-Active 운영의 원칙(두 사이트가 동등한 운영 권한을 가진다) 과 가장 자연스럽게 정합한다. 1장과 2장에서 정의한 클라우드 네이티브 DR 의 본질(선언된 상태의 재현) 이 레지스트리 계층까지 확장된 형태다.

양방향 동기의 첫 번째 장점은 어느 한 사이트가 마비되어도 다른 사이트의 빌드 파이프라인이 계속 동작한다는 점이다. SSoT 가용성이라는 단일 실패점이 사라진다. 두 번째 장점은 두 사이트가 지리적으로 떨어져 있을 때 각 사이트의 빌드 팀이 로컬 레지스트리만 바라보면 되므로 빌드 시간이 회선 지연에 영향받지 않는다는 점이다. 두 사이트가 동시에 신기능을 빌드·배포해도 양쪽이 서로의 빌드 결과를 단방향 의존 없이 흡수한다.

양방향 동기의 첫 번째 단점은 충돌 정책 설계가 필수라는 점이다. 두 사이트의 빌드 파이프라인이 같은 태그(예: `v1.4.2`) 에 서로 다른 이미지를 동시에 등록할 가능성이 구조적으로 존재한다. Harbor 의 Replication 정책은 이러한 충돌 상황에서 어느 쪽을 우선할지(시간순·라벨 기반·수동 승인) 를 정책으로 지정하도록 설계되어 있고, 이 결정은 거버넌스 보드 의결 대상이다 [S6]. OCI Distribution Specification 의 manifest digest 는 이미지 내용에 대한 SHA-256 해시이므로 같은 태그라도 내용이 다르면 digest 가 다르며, 충돌 감지 자체는 표준 사양 안에서 자동으로 이루어진다 [S7]. 충돌 정책을 명시하지 않은 양방향 동기는 운영팀이 사고 발생 후에 정책을 즉석에서 결정하는 구조이며, 사이버 재해 시점에 의사결정 지연을 만들어 낸다.

두 번째 단점은 회선 단절 시 재동기 비용이 크다는 점이다. 두 사이트가 분리되어 있는 동안 각자 빌드·등록을 계속하면 회선 복구 시점에 양쪽 변경분을 양방향으로 비교해 누락분과 충돌분을 분류하는 비용이 발생한다. 변경분이 수십 개의 이미지·태그 수준이면 자동 재동기가 가능하지만, 단절 기간이 길어 수백 개의 변경분이 누적되면 사람 개입이 필요한 수동 재동기 비용이 커진다. 회선 단절 SLA 와 양방향 패턴의 재동기 시간을 사전에 매핑해 운영 한도를 정해 두는 것이 권장된다. 양방향 패턴은 회선 단절이 드물고(분 단위) 두 사이트가 항상 거의 동기 상태인 환경에서 진가를 발휘한다.

세 번째 단점은 권한 모델의 복잡성이다. 단방향 패턴에서는 SSoT 사이트가 빌드 권한을 독점하고 대상 사이트는 read-only 다. 양방향 패턴에서는 양쪽이 빌드 권한을 가지므로 어느 팀이 어느 사이트의 어느 네임스페이스에 무엇을 등록할 수 있는지의 권한 매트릭스가 두 배로 늘어난다. 이 권한 매트릭스는 사람이 손으로 관리하면 사고 발생률이 높으므로 7.2.2 항의 정책 엔진으로 자동화하는 것이 표준이다.

Pull-through Cache 패턴은 앞의 세 패턴과 성격이 다르다. 대상 사이트의 레지스트리가 자기 저장소에 이미지를 영구 보관하지 않고 요청이 들어올 때 SSoT 에서 끌어와 캐시한다. Harbor 는 이 패턴을 proxy cache project 라는 별도 프로젝트 유형으로 분리해 제공하며, 일반 project 와 동작 모델이 다르다는 점을 운영자가 명시적으로 인식하도록 설계되어 있다 [S6]. 캐시된 이미지는 일정 시간 또는 일정 용량까지만 유지되며 만료되면 다음 요청 시 다시 끌어온다. OCI Distribution Specification 의 manifest digest 가 immutable(불변) 식별자이기 때문에 캐시 일관성 검증이 표준 사양 안에서 자동으로 이루어진다 [S7].

Pull-through Cache 의 첫 번째 장점은 대상 사이트의 저장소 용량 요구가 작다는 점이다. 평시 운영에서 실제로 사용하는 이미지(특정 시점에 배포된 워크로드가 참조하는 manifest 집합) 만 보관하면 된다. SSoT 에 등록된 전체 이미지 풀의 작은 부분집합만 캐시되므로 대상 사이트의 스토리지 비용이 줄어든다. 두 번째 장점은 동일 이미지를 여러 노드가 요청해도 SSoT 까지의 회선 사용은 한 번뿐이라는 점이다 [S6]. 같은 사이트의 수십 개 노드가 동일 워크로드를 동시에 띄울 때(예: 페일오버 직후의 일괄 기동) 회선 부하가 노드 수에 비례하지 않는다.

Pull-through Cache 의 단점은 캐시 미스(miss) 시점에 SSoT 가용성이 즉시 필요하다는 점이다. SSoT 가 마비된 사이버 재해 시점에 대상 사이트가 새 워크로드를 띄우려고 하면 캐시에 없는 이미지를 받을 방법이 없다. 따라서 Pull-through Cache 는 단독 패턴이 아니라 다른 패턴(주로 단방향 Pull) 과 결합해 사용하는 보조 패턴으로 보는 편이 적절하다. 평시에는 Pull-through Cache 로 회선을 절감하고, 정해진 주기에는 단방향 Pull 로 핵심 이미지의 영구 사본을 확보하는 2단 구성이 자주 채택된다.

4 패턴의 트레이드오프를 요약하면 다음과 같다. 단방향 Push 는 SSoT 단순성과 충돌 0 의 안정성이 강점이고 SSoT 가용성 의존이 약점이다. 단방향 Pull 은 대상 사이트의 회선 자율성이 강점이고 RPO lag 가 약점이다. 양방향 동기는 SSoT 단일 실패점 제거와 Active-Active 정합이 강점이고 충돌 정책 설계 부담이 약점이다. Pull-through Cache 는 회선·스토리지 절감이 강점이고 캐시 미스 시 SSoT 의존이 약점이다. Push 와 Pull 의 결합형, 양방향과 Pull-through Cache 의 결합형 등 혼합 패턴이 실무에서 자주 채택되며, 워크로드별로 다른 패턴을 적용하는 패턴 다중화도 표준에 어긋나지 않는다.

7.2 망분리 환경의 동기화 토폴로지

공공기관·금융기관의 운영 환경은 거의 예외 없이 망분리(air-gapped) 와 DMZ 토폴로지를 갖는다. 외부망과 내부망이 물리적·논리적으로 분리되어 있고 두 망 사이의 모든 트래픽이 DMZ 의 중간 게이트웨이를 통과해야 한다. 컨테이너 레지스트리 동기화도 이 토폴로지 안에서 설계되어야 하며, 7.1 절의 4 패턴을 그대로 적용할 수 없는 제약 — 운영망 간 직접 통신 금지, 보안 등급 단방향 흐름 정책, DMZ 게이트웨이 경유 의무 — 이 존재한다. 표준 해법은 DMZ 게이트웨이 레지스트리를 두고 단방향 흐름의 결합으로 양방향을 구현하는 2단 토폴로지이며, 그 위에 Kyverno 정책 엔진이 운영 시점 서명 검증을 강제하는 구조다.

7.2.1 양방향 Geo-Replication + 서명 검증 토폴로지

망분리 환경의 표준 토폴로지는 4개 구성 요소로 이루어진다 — 주(Primary) 사이트 내부망의 운영 레지스트리, 부(Secondary) 사이트 내부망의 운영 레지스트리, 두 사이트 사이의 DMZ 게이트웨이 레지스트리, 그리고 모든 이미지 흐름의 무결성을 검증하는 서명 검증 계층이다. 양방향 동기 패턴을 직접 두 운영 레지스트리 사이에 적용하지 않고 DMZ 게이트웨이를 경유하는 2단 구성으로 분해한다.

주 사이트의 빌드 파이프라인이 신규 이미지를 생성하면 주 사이트 운영 레지스트리에 등록된다. 등록 시점에 Sigstore Cosign 서명이 함께 생성되어 manifest 의 referrers 로 첨부된다. 등록 직후 Replication 정책이 발동되어 이미지가 DMZ 게이트웨이 레지스트리로 단방향 Push 된다. DMZ 게이트웨이는 자기 영역의 이미지 풀에 서명을 포함한 manifest 와 blob 을 보관한다. 부 사이트는 자기 운영 주기에 따라 DMZ 게이트웨이에서 단방향 Pull 로 이미지를 받아 내부 운영 레지스트리에 적재한다. Pull 시점에 부 사이트의 Replication 클라이언트가 Cosign 서명을 검증하고 검증을 통과한 이미지만 내부 레지스트리에 받아들인다 [S7]. Cosign 의 검증 흐름은 Sigstore 공식 문서에서 정의된 표준 절차를 따르며, 키 기반 검증과 keyless 검증 양쪽이 모두 동일한 verify 명령으로 처리된다 [S7].

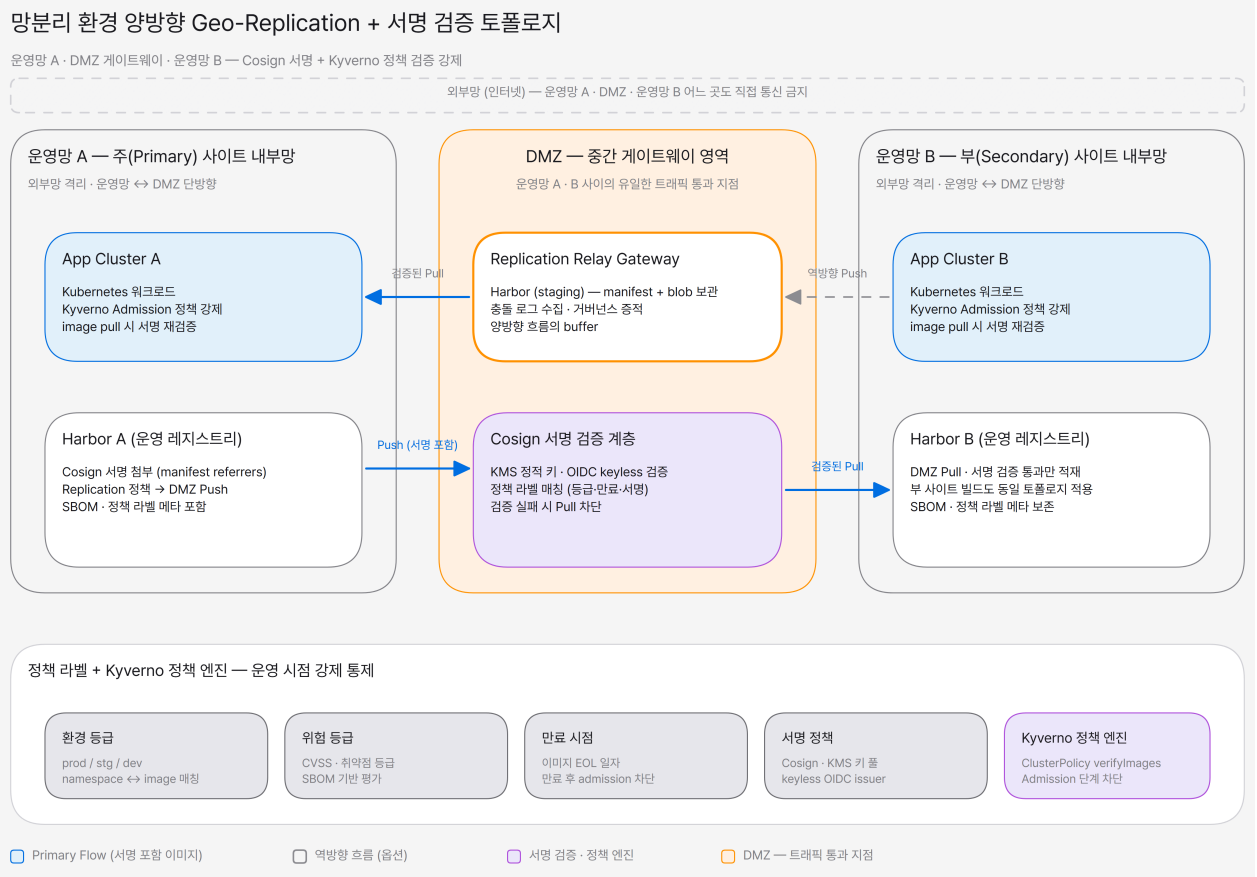
이 토폴로지의 첫 번째 설계 의도는 두 사이트 사이의 직접 연결을 제거하는 것이다. 주 사이트 운영망과 부 사이트 운영망이 서로의 IP·포트를 직접 알 필요가 없고, 각자 DMZ 게이트웨이 한 곳만 알면 된다. 망분리 정책에서 운영망 간 직접 통신이 금지되는 환경에서도 동기화가 표준 흐름 안에서 동작한다. 두 번째 설계 의도는 DMZ 게이트웨이가 staging 역할을 동시에 수행한다는 점이다. 주 사이트가 잘못된 이미지를 등록해도(악의적 또는 사고) 부 사이트는 그 이미지를 Pull 하기 전에 검증할 기회가 한 번 더 생긴다. 세 번째 설계 의도는 서명이 manifest 의 referrers 로 첨부되어 OCI Distribution Specification 의 표준 이동 경로를 따라 자동으로 함께 복제된다는 점이다 [S7]. 서명을 별도 채널로 전달할 필요가 없으므로 운영 절차가 단순해진다.

양방향이 필요한 워크로드(예: 두 사이트 모두에 빌드 파이프라인이 존재하는 경우) 는 같은 토폴로지를 방향만 반대로 한 번 더 구성한다. 부 사이트 → DMZ 게이트웨이 → 주 사이트의 단방향 흐름을 추가 Replication 정책으로 설치하면 전체 토폴로지는 두 운영 레지스트리가 DMZ 게이트웨이를 사이에 두고 양방향으로 동기화되

는 형태가 된다. 7.1.2 항에서 정의한 양방향 동기의 충돌 정책은 이 토폴로지에서도 그대로 적용되며, 충돌이 발생하면 DMZ 게이트웨이 시점에 충돌 로그가 남아 거버넌스 보드의 사후 검토 자료로 사용된다.

회선 단절 시 재동기 비용도 이 토폴로지에서 줄어든다. 단절 기간 동안 주 사이트는 DMZ 게이트웨이가 자기 운영망 안에 있다면 게이트웨이까지의 Push 를 계속할 수 있다(운영망 내 통신은 외부 회선과 무관하다). 회선 복구 시점에는 게이트웨이에 누적된 이미지만 부 사이트가 Pull 하면 되므로 양 사이트의 변경분을 양방향으로 비교하는 무거운 작업이 발생하지 않는다. 다만 DMZ 게이트웨이의 스토리지 용량이 누적 변경분을 보관하기에 충분해야 하며, 단절 시간 SLA × 평시 신규 이미지 속도 × 평균 이미지 크기로 산출한 용량을 사전에 확보해 두는 운영 표준이 필요하다.

서명 검증 계층은 단순한 통과 검사가 아니라 신뢰 정책의 운영 시점 강제다. Cosign 의 검증 키는 일반적으로 Sigstore 의 keyless flow(OIDC ID 기반) 또는 사내 KMS 의 정적 키를 사용하며, 공공기관 환경에서는 후자가 일반적이다. 부 사이트의 검증 클라이언트가 정적 키 풀을 갖고 있고 각 이미지의 서명을 그 풀에 대해 검증한다. 검증을 통과하지 못한 이미지는 Pull 자체가 실패하며 내부 레지스트리에 적재되지 않는다. 이 정책은 빌드 파이프라인이 침해된 시나리오에서도(공격자가 잘못된 이미지를 주 사이트 레지스트리에 등록한 경우) 부 사이트가 그 이미지를 Pull 하지 못하도록 차단하는 마지막 방어선이다. 이 검증을 운영 시점에 다시 강제하는 것이 7.2.2 항의 정책 엔진이다.



7.2.2 Kyverno 정책 엔진 통합 — 서명 미검증 이미지 차단

레지스트리 계층의 서명 검증은 이미지가 레지스트리에 적재될 때 한 번 동작한다. 하지만 워크로드가 실제 클러스터에 배포되는 시점에 그 이미지가 검증된 이미지인지 다시 확인할 필요가 있다. 7.2.1 항의 토폴로지에서 부 사이트 내부 레지스트리는 검증된 이미지만 받지만, 운영 클러스터가 그 레지스트리가 아닌 다른 경로의 이미지

를 참조한 경우(잘못된 매니페스트, 임의의 외부 레지스트리 참조, 운영자 실수) 에는 검증 계층을 우회하게 된다. 이 우회를 차단하는 마지막 정책 강제 지점이 Kubernetes 클러스터의 정책 엔진이다 [S7].

Kyverno 는 Kubernetes 의 Admission Controller 계층에서 동작하는 정책 엔진이다. 워크로드의 Pod 매니페스트가 클러스터에 적용되기 직전에 Kyverno 의 정책 평가가 실행되어 정책에 어긋나는 매니페스트는 적용 자체가 거부된다. 컨테이너 이미지 관점에서 Kyverno 가 강제할 수 있는 대표 정책은 다음 세 가지다. 첫째, 허용된 레지스트리 화이트리스트 — 매니페스트의 image 필드가 사전에 승인된 레지스트리 도메인 목록 안에 있어야 한다. 둘째, Cosign 서명 검증 — 매니페스트의 image 가 가리키는 이미지에 유효한 Cosign 서명이 첨부되어 있어야 한다. 셋째, SBOM(Software Bill of Materials, 소프트웨어 부품 명세서) 첨부 검증 — 이미지의 referrers 에 SBOM 이 첨부되어 있어야 한다.

Kyverno 의 서명 검증 정책은 ClusterPolicy 리소스로 정의되며, verifyImages 규칙이 핵심이다. 규칙은 검증할 이미지 패턴(예: `registry.openmaru.io/*`) 과 검증에 사용할 공개 키(또는 keyless flow 의 OIDC issuer) 를 명시한다. 매칭되는 Pod 가 생성되면 Kyverno 는 이미지의 manifest 에서 referrers 를 조회하고 첨부된 Cosign 서명을 공개 키로 검증한다. 검증을 통과한 이미지만 Pod 생성을 허용하며, 미검증 또는 검증 실패 이미지는 Admission 단계에서 거부된다.

이 정책은 사람의 수작업 점검을 자동화한다. 운영팀이 평시에 모든 Pod 매니페스트를 일일이 검토해 잘못된 레지스트리 참조를 찾아내는 것은 현실적이지 않다. 사이버 재해 시점에는 더더욱 불가능하다. Kyverno 정책은 운영 시점에 자동으로 모든 워크로드 배포를 차단할 수 있으므로 사람 개입을 사고 사후 검토로 옮긴다. 정책 위반 사건은 Kyverno 의 PolicyReport 리소스로 누적되며, 이는 ISMS-P 감사 시점에 보안 통제의 운영 증적으로 제출 가능한 형태로 자동 기록된다.

정책 라벨링은 서명 검증과 결합되는 두 번째 강제 메커니즘이다. 3장에서 정의한 정책 단위 라벨(환경 등급, 위험 등급, 만료 시점, 서명 정책) 은 이미지의 manifest 또는 referrers 에 메타데이터로 첨부된다. Kyverno 의 ClusterPolicy 는 이미지의 라벨이 워크로드의 namespace 또는 ServiceAccount 의 권한 등급과 일치하는지를 추가로 검증할 수 있다. 예를 들어 prod 등급 namespace 의 Pod 는 prod 등급 라벨이 부착된 이미지만 사용할 수 있다는 정책을 강제하면 dev 등급 이미지가 실수로 prod 클러스터에 배포되는 사고를 차단한다. 이 정책의 평가 로직이 정책 엔진 안에 들어 있으므로 운영팀은 라벨 정책을 변경할 때 정책 매니페스트만 수정하면 되며, 모든 클러스터에 즉시 자동 반영된다.

SBOM 검증은 공급망 보안 통제의 마지막 단계다. SBOM 은 이미지 안에 포함된 모든 라이브러리·런타임·의존성의 명세이며, 빌드 시점에 Cosign 으로 함께 서명되어 referrers 로 첨부된다. Kyverno 는 SBOM 의 존재를 검증하는 정책뿐 아니라 SBOM 안의 특정 의존성(예: 알려진 취약점이 있는 라이브러리 버전) 의 부재를 검증하는 정책도 표현할 수 있다. 사이버 재해 시점에 즉시 새 이미지를 배포해야 하는 압박이 있을 때조차 SBOM 정책 검증을 통과하지 못한 이미지는 배포가 차단되므로 신뢰 부트(trusted boot) 의 마지막 방어선이 유지된다.

정책 엔진 도입이 운영팀에 부담을 주는 것은 사실이다. 정책 위반이 발생하면 워크로드 배포 자체가 실패하므로 운영팀이 정책 매니페스트와 워크로드 매니페스트 양쪽을 모두 이해해야 한다. 하지만 이 부담은 ISMS-P 통제 자동화의 1차 후보이며, 사람 수작업 점검이 가질 수 있는 점검 누락·점검 지연·점검 인력 의존성을 모두 제거한다. 6장에서 정리한 클라우드 네이티브 DR 의 3단계 절차(페일오버 트리거 → GitOps 동기화 → 트래픽 전환) 가 분 단위로 끝나려면 정책 검증 자체가 사람 개입 없이 운영 시점에 자동 동작해야 하며, 정책 엔진이 그 자동화의 핵심 구성 요소다.

4 동기화 패턴(단방향 Push, 단방향 Pull, 양방향 동기, Pull-through Cache) 은 어느 한 패턴이 모든 상황에 최선인 경우가 없으므로 워크로드별·사이트별로 다르게 선택한다. 망분리·DMZ 환경에서는 두 운영 레지스트리 사이에 DMZ 게이트웨이 레지스트리를 두고 단방향 흐름의 결합으로 양방향을 구현하는 2단 토폴로지가 표준이다. Cosign 서명은 manifest 의 referrers 로 첨부되어 표준 사양 안에서 자동 복제되며, Kyverno 정책 엔진이 운영 시점에 서명 검증·라벨 정책·SBOM 검증을 강제해 사람 수작업 점검을 자동화한다. 자사 망분리·DMZ 토폴로지에 적합한 동기화 패턴과 정책 엔진 도입 범위를 거버넌스 보드에서 이번 분기 안에 의결하는 일이 본 장을 읽고 실행 가능한 첫 번째 행동이다.

8장: 해외 클라우드 네이티브 DR 전환 사례

해외 기업들이 컨테이너 기반 멀티 사이트 DR 로 옮겨 간 발표 자료들은 회의실에서 "우리만 늦은 것은 아닌가" 또는 "이미 검증된 모델이 있는가" 라는 두 질문에 한꺼번에 답을 줍니다. 이 8장은 의류·항공 IT·금융·미션 크리티컬 영역에서 공개된 다섯 사례를 같은 평면에 놓고, 사례마다 다른 표면적 결정 뒤에서 반복되는 공통 패턴을 끌어냅니다. 사례를 나열하는 것이 목적이 아니라, 다섯 사례를 가로지르는 공통 결정 항목 세 가지를 자사 도입 의사결정의 출발점으로 삼는 것이 목적입니다.

장의 결론은 두 줄로 요약됩니다. 첫째, 채택 스택은 사례마다 다르지만 (1) GitOps 통합 (2) 멀티 클러스터(Multi-cluster) 거버넌스 도구 (3) 애플리케이션과 데이터 복제 경로의 분리, 이 세 가지 결정은 다섯 사례에서 모두 동일하게 등장합니다. 둘째, 운영 인력 모델은 일관되게 사이트 신뢰성 엔지니어링(SRE, Site Reliability Engineering) 과 플랫폼 엔지니어링(Platform Engineering) 중심으로 재편되며, DR 전담 조직이 사라지는 대신 평시 운영 조직이 DR 책임을 함께 갖습니다. 자사가 다음 24개월 안에 결정해야 할 항목은 이 세 가지 공통 패턴과 운영 조직 재편의 시점입니다.

8.1 해외 5개 사례의 5요소 분해

다섯 사례의 발표 자료는 분량과 강조점이 제각각이지만, 의사결정 회의에서 인용하려면 모두 같은 표 한 장에 같은 항목 순서로 늘어서 있어야 합니다. 본 절이 사용하는 5요소는 **전환 전 아키텍처, 전환 동기, 채택 스택, 측정 복구 지표, 운영 변화** 입니다. 측정 복구 지표 항목은 공개 발표에 수치가 명시된 경우만 인용하며, 명시되지 않은 경우 회의실 언어로 그 범위만 표시합니다.

8.1.1 Adidas, Lufthansa Systems — 소비재·항공 IT 영역

Adidas 의 KubeCon EU 2024 발표는 멀티 리전 쿠버네티스 운영이 평시와 재해 양쪽을 같은 운영 모델로 처리한다는 점을 보여 줍니다. Adidas 는 글로벌 전자상거래 트래픽이 블랙 프라이데이 같은 단기 피크에서 평시 대비 수십 배로 치솟는 부담을 갖고 있고, 동시에 유럽·미주·아시아의 사이트 가용성을 동일 SLA 로 보장해야 합니다. 발표 자료에 따르면 Adidas 는 쿠버네티스 디스트리뷰션(distribution) 의 하나를 채택해 멀티 리전 클러스터를 운영하며, 평시에는 트래픽 분산을 위한 Active-Active 구성으로, 재해 시에는 한 사이트가 다른 사이트의 트래픽을 흡수하는 페일오버 구성으로 같은 인프라가 두 역할을 수행합니다 [6]. 전환 동기는 두 가지로 정리됩니다 — 첫째, 블랙 프라이데이 같은 트래픽 피크 대응이 가상화 환경의 정적 자원 할당으로는 비용 효율을 맞추기 어려웠고, 둘째, 멀티 리전 운영을 별도 DR 조직이 책임지는 모델이 인적 의존성을 만들어 야간·주말 대응 실패 위험을 키웠습니다.

채택 스택은 쿠버네티스 디스트리뷰션 위에 GitOps 도구를 얹어 멀티 리전 클러스터의 매니페스트 동기를 단일 Git 저장소에서 관리하는 형태로 정리됩니다. Argo CD ApplicationSet 같은 멀티 클러스터 배포 도구가 단일 매니페스트를 여러 클러스터로 펼치므로 [S4], 한쪽 사이트가 마비되어도 다른 사이트의 워크로드 정의가 즉시 다르게 적용되지 않습니다. 측정 복구 지표는 공개 발표에서 정량 수치가 모두 공개되지는 않았으나, 발표자는 가상화 시대의 시간 단위 RTO 가 컨테이너 전환 후 분 단위로 단축되었다고 설명했습니다 [6:1]. 운영 변화는 SRE 와 플랫폼 엔지니어링 팀이 평시 트래픽 운영과 DR 책임을 함께 갖는 단일 조직 모델로 정리되었으며, 별

도 DR 운영조 편성이 사라졌습니다. 이 변화는 인건비 절감보다는 야간·주말 운영의 안정성 향상과 의사결정 단계 축소 효과가 크다고 발표자는 강조했습니다.

Lufthansa Systems 의 Red Hat Summit 발표는 항공 운항이라는 미션 크리티컬 영역에서도 컨테이너 기반 멀티 사이트 운영이 안착할 수 있음을 보여 줍니다. Lufthansa Systems 는 항공사에 운항·승무·정비 시스템을 제공하는 IT 자회사로서, 무중단 운영이 곧 안전 요건과 직결되는 워크로드를 운영합니다. 발표 자료에 따르면 이 회사는 항공 운항 시스템 일부를 컨테이너 환경으로 전환하면서 멀티 사이트 운영을 동시에 설계했고, 가상화 시대의 시점 백업·복원 모델이 항공 운항 요구하는 RTO·RPO 와 정합되지 않았다는 점을 전환 동기로 명시했습니다 [7]. 전환 전 아키텍처는 가상화 기반 단일 사이트 운영과 백업 사이트의 시점 복제 조합이었으며, 항공 운항 피크 시간대의 단기 장애도 회복 절차에 시간 단위 소요가 일반적이었다고 설명합니다.

채택 스택은 쿠버네티스 디스트리뷰션, GitOps 도구, 컨테이너 레지스트리 Geo-Replication 의 조합입니다. 발표는 특정 벤더 도구의 운영 노하우를 강조했지만 본 백서의 vendor-neutral 관점에서 환원하면, 멀티 사이트 매니페스트 동기와 이미지 복제, 데이터 복제 세 경로를 분리해서 설계한 점이 본질입니다. 측정 복구 지표는 항공 운항 시스템 특성상 구체 수치 공개 범위가 제한되지만, 발표자는 단기 장애의 복구 시간이 시간 단위에서 분 단위로 단축되었다고 보고했습니다 [7:1]. 운영 변화는 항공 운항 IT 인력이 전통적 인프라 운영 모델에서 플랫폼 엔지니어링 모델로 옮겨 가는 과정이었으며, 발표는 인력 재교육에 약 12개월이 소요되었다고 언급했습니다. 이 12개월이라는 숫자는 자사 도입 시 인력 전환 일정을 계획할 때 의미 있는 참고치입니다.

두 사례를 같은 표로 두면 공통점이 즉시 드러납니다. 전환 전 아키텍처가 가상화 단일 사이트 + 시점 백업 사이트 조합이었다는 점, 전환 동기가 단기 트래픽 피크 또는 미션 크리티컬 요구에 가상화 모델이 정합되지 않았다는 점, 채택 스택이 GitOps + 멀티 클러스터 거버넌스 + 이미지·데이터 복제 분리 세 결정으로 환원된다는 점, 운영 변화가 SRE·플랫폼 엔지니어링 중심 재편이라는 점은 영역(소비재·항공 IT) 이 달라도 반복됩니다.

표 9 · 해외 사례 5건 × 5요소 비교

Adidas · Lufthansa Systems · ING Bank · JPMorgan Chase · SAP S/4HANA on Kubernetes — vendor-neutral 정성 요약

사례	전환 전 아키텍처	전환 동기	채택 스택	측정 복구 지표	운영 변화
Adidas (KubeCon EU)	모놀리식 e-Commerce + 자체 데이터센터 단일 사이트	Black Friday 단기 트래픽 피크 + 사이트 장애 시 DR	쿠버네티스 멀티 리전 + GitOps + 컨테이너 레지스트리 복제	RTO 분 단위 (시간 단위 → 분 단위)	SRE · 플랫폼팀 재편 + 평시 · DR 운영 통합
Lufthansa Systems (Red Hat Summit)	항공 운항 레거시 + 가상화 단일 사이트	미션 크리티컬 24/7 운항 + 시점 백업 회복 시간 한계	쿠버네티스 + 멀티 클러스터 + GitOps + 이미지 · 데이터 복제 분리	RTO 분 단위 (시간 단위 → 분 단위)	플랫폼 엔지니어링 도입 + 인력 재교육 약 12개월
ING Bank	분산 모놀리식 + 사이트 · 도구 분리	금융 감독 변경 통제 + 평시 · DR 운영 분리 부담	멀티 클러스터 거버넌스 + GitOps + 정책 엔진 (Kyverno 등)	MTTR 한 자릿수배 단축 (분 단위)	평시 · DR 플랫폼팀 통합 + 야간 · 주말 호출 ½ 이하 감소
JPMorgan Chase	분산 모놀리식 + 단일 사이트 시점 백업	글로벌 24시간 거래 미션 크리티컬 RTO · RPO	쿠버네티스 멀티 사이트 + GitOps + 분산 데이터 복제 (앱 · 데이터 경로 분리)	RTO 분 단위 + RPO 분 단위 미만 목표	글로벌 분산 SRE 모델 + GitOps 매니페스트 SSoT 인수인계
SAP S/4HANA on Kubernetes	ERP 미들웨어 통합 + 가상화 라이선스 종속	클라우드 전환 + 롤링 업데이트 무중단 패치	쿠버네티스 Operator + 멀티 클러스터 거버넌스 + 데이터 복제 분리	ERP RTO 분 단위 (시간 단위 → 분 단위)	ERP 운영 인력 플랫폼 엔지니어링 편입 + 재교육 약 6~12개월

공통 패턴 — 전환 동기: 가상화 단일 사이트 한계 · 채택 스택: 멀티 클러스터 + GitOps + 이미지·데이터 경로 분리 · 운영 변화: 평시·DR 통합 + SRE/플랫폼 엔지니어링 재편 · 측정 지표: 시간 단위 → 분 단위 RTO

8.1.2 ING Bank, JPMorgan Chase, SAP S/4HANA on Kubernetes — 금융·미션 크리티컬 영역

ING Bank 는 멀티 클러스터 쿠버네티스 플랫폼을 금융권 규제 환경에서 운영하는 사례입니다. ING 의 공개 발표는 멀티 클러스터 플랫폼을 자체 구축해 본점·재해 사이트·개발 환경을 같은 모델로 운영하고, 금융 감독 요구의 감사 추적·접근 통제·변경 통제를 GitOps 매니페스트 이력으로 통합 관리한다는 점을 강조합니다 [8]. 전환 동기는 셋으로 정리됩니다 — 첫째, 멀티 클러스터 운영이 가상화 시대에는 별도 운영 조직과 별도 도구 묶음을 요구했고, 둘째, 금융 감독 요구의 변경 통제가 가상화 환경에서는 수작업 점검 비중이 높았으며, 셋째, 평시 운영과 DR 운영의 도구·인력이 분리되어 있어 야간·주말 사고 대응이 느렸다는 점입니다.

채택 스택은 멀티 클러스터 거버넌스 도구, GitOps 도구, 정책 엔진(Kyverno 등) 의 조합으로 정리됩니다. 멀티 클러스터 거버넌스 도구는 여러 사이트의 클러스터를 단일 콘솔에서 관리하면서 정책·인증·접근 통제를 일원화합니다. GitOps 도구는 변경 이력을 Git 커밋으로 박제하므로 금융 감독 요구의 변경 통제 항목을 자연스럽게 충족합니다 [S4]. 정책 엔진은 입수 시점에 서명 검증, 라벨 일치, 보안 정책 준수를 자동으로 강제하므로 ISMS-P 와 유사한 감독 통제 항목의 자동화 비중을 끌어올립니다. 측정 복구 지표는 금융권 발표 특성상 구체 수치보다는 운영 모델 변화 중심으로 공개되었으며, 발표자는 사고 대응 평균 시간(MTTR, Mean Time To Recovery) 이 가상화 시대 대비 한 자릿수배 단축되었다고 보고했습니다 [8:1].

운영 변화는 평시 운영 팀과 DR 운영 팀의 통합으로 정리됩니다. ING 의 발표는 통합 후 약 18개월 시점에 야간·주말 사고 호출 횟수가 절반 이하로 줄었다고 언급했습니다. 이 변화는 인력 감축이 아니라 인력 배치 변화의 결과이며, 같은 인원이 평시와 DR 양쪽을 통합 운영하는 모델이 야간·주말 호출 자체를 줄이는 구조라는 점이 핵심입니다.

JPMorgan Chase 는 미션 크리티컬 금융 워크로드의 쿠버네티스 운영 사례를 공개 기술 블로그로 발표한 사례입니다. JPMorgan 의 공개 자료는 글로벌 금융 거래 시스템 일부를 쿠버네티스 멀티 사이트로 옮기면서, 가상화 시대의 단일 사이트·시점 백업 모델로는 글로벌 24시간 거래의 RTO·RPO 요구를 충족할 수 없었던 배경을 설명합니다 [9]. 채택 스택은 쿠버네티스 디스트리뷰션, GitOps 도구, 분산 데이터 복제 도구의 조합으로 정리됩니다. 본 백서의 vendor-neutral 관점에서 환원하면, 애플리케이션 복제 경로(이미지·매니페스트) 와 데이터 복제 경로(트랜잭션·로그) 를 분리해 설계한 점이 본질입니다. 측정 복구 지표는 발표 범위 제약으로 구체 수치가 공개되지 않았으나, 분 단위 RTO 달성과 분 단위 미만 RPO 달성을 운영 목표로 명시했습니다.

운영 변화는 글로벌 분산 SRE 모델로 정리됩니다. 24시간 거래를 지원하는 워크로드는 시간대별 SRE 조직이 인수인계하는 모델이 자연스럽고, GitOps 매니페스트가 인수인계의 단일 출처(SSoT) 역할을 합니다. 발표자는 인수인계 시점의 정보 손실이 GitOps 도입 전후로 유의미하게 줄었다고 보고했습니다.

SAP S/4HANA on Kubernetes 는 기업 핵심 업무 시스템(ERP) 의 쿠버네티스 운영 사례입니다. SAP 의 공식 자료는 SAP S/4HANA 같은 미션 크리티컬 ERP 워크로드를 쿠버네티스 환경에서 운영하는 운영 모델과 운영 도구 묶음을 정리합니다 [10]. 전환 동기는 두 가지로 정리됩니다 — 첫째, 가상화 환경의 라이선스·하드웨어 종속이 ERP 의 멀티 사이트 운영 비용을 끌어올렸고, 둘째, ERP 의 업그레이드·패치 운영이 가상화 환경에서는 정지 시간을 요구했으나 쿠버네티스 환경에서는 롤링 업데이트(rolling update) 로 무중단 패치가 가능하다는 점입니다. 채택 스택은 쿠버네티스 디스트리뷰션 위에 SAP 자체 운영 도구를 얹는 형태이지만, 본 백서의 관점에서 환원하면 멀티 클러스터 거버넌스, GitOps 매니페스트 동기, 데이터 복제 분리 세 결정이 동일하게 등장합니다.

측정 복구 지표는 발표 자료마다 다르지만, 미션 크리티컬 ERP 의 RTO 가 가상화 시대의 시간 단위에서 분 단위로 단축된 사례들이 보고됩니다 [10:1]. 운영 변화는 ERP 운영 인력의 플랫폼 엔지니어링 모델 편입이며, ERP 도메인 전문가가 컨테이너 운영 기초 역량을 갖추는 재교육 과정이 약 6~12개월 단위로 진행되었다고 보고됩니

다. 이 6~12개월이라는 범위는 ERP 도메인 전문가의 컨테이너 학습 곡선을 의미 있는 참고치로 자사 일정에 반영할 수 있습니다.

세 사례를 같은 표로 두면 금융·미션 크리티컬 영역에서도 공통 패턴이 반복됨이 확인됩니다. 전환 동기에서 가상화 시대의 라이선스·인력 분리·변경 통제 부담이 공통으로 등장하고, 채택 스택에서 멀티 클러스터 거버넌스 도구의 비중이 의류·항공 IT 사례보다 더 크게 강조되며, 운영 변화에서 평시 운영과 DR 운영의 통합이 모두 핵심 효과로 보고됩니다. 다섯 사례 전체를 5요소 표 한 장에 같은 순서로 늘어놓은 결과가 본 절 시작에서 가리킨 표 9 (해외 사례 5건 × 5요소 비교) 입니다

표 9 · 해외 사례 5건 × 5요소 비교

Adidas · Lufthansa Systems · ING Bank · JPMorgan Chase · SAP S/4HANA on Kubernetes — vendor-neutral 정성 요약

사례	전환 전 아키텍처	전환 동기	채택 스택	측정 복구 지표	운영 변화
Adidas (KubeCon EU)	모놀리식 e-Commerce + 자체 데이터센터 단일 사이트	Black Friday 단기 트래픽 피크 + 사이트 장애 시 DR	쿠버네티스 멀티 리전 + GitOps + 컨테이너 레지스트리 복제	RTO 분 단위 (시간 단위 → 분 단위)	SRE · 플랫폼팀 재편 + 평시 · DR 운영 통합
Lufthansa Systems (Red Hat Summit)	항공 운항 레거시 + 가상화 단일 사이트	미션 크리티컬 24/7 운항 + 시점 백업 회복 시간 한계	쿠버네티스 + 멀티 클러스터 + GitOps + 이미지 · 데이터 복제 분리	RTO 분 단위 (시간 단위 → 분 단위)	플랫폼 엔지니어링 도입 + 인력 재교육 약 12개월
ING Bank	분산 모놀리식 + 사이트 · 도구 분리	금융 감독 변경 통제 + 평시 · DR 운영 분리 부담	멀티 클러스터 거버넌스 + GitOps + 정책 엔진 (Kyverno 등)	MTTR 한 자릿수배 단축 (분 단위)	평시 · DR 플랫폼팀 통합 + 야간 · 주말 호출 ½ 이하 감소
JPMorgan Chase	분산 모놀리식 + 단일 사이트 시점 백업	글로벌 24시간 거래 미션 크리티컬 RTO · RPO	쿠버네티스 멀티 사이트 + GitOps + 분산 데이터 복제 (앱 · 데이터 경로 분리)	RTO 분 단위 + RPO 분 단위 미만 목표	글로벌 분산 SRE 모델 + GitOps 매니페스트 SSoT 인수인계
SAP S/4HANA on Kubernetes	ERP 미들웨어 통합 + 가상화 라이선스 종속	클라우드 전환 + 롤링 업데이트 무중단 패치	쿠버네티스 Operator + 멀티 클러스터 거버넌스 + 데이터 복제 분리	ERP RTO 분 단위 (시간 단위 → 분 단위)	ERP 운영 인력 플랫폼 엔지니어링 편입 + 재교육 약 6~12개월

공통 패턴 — 전환 동기: 가상화 단일 사이트 한계 · 채택 스택: 멀티 클러스터 + GitOps + 이미지·데이터 경로 분리 · 운영 변화: 평시·DR 통합 + SRE/플랫폼 엔지니어링 재편 · 측정 지표: 시간 단위 → 분 단위 RTO

.

8.2 다섯 사례에서 도출되는 공통 패턴

다섯 사례 모두 채택 스택과 표현 방식이 다르지만 공통 결정 항목 세 가지 — GitOps 통합, 멀티 클러스터 거버넌스 도구, 애플리케이션·데이터 복제 경로의 분리 — 가 반복됩니다. 이 반복은 자사 도입 의사결정이 무엇을 먼저 결정해야 하는지를 가리키는 신호이며, 도구 선택보다 결정 순서가 먼저인 이유입니다.

8.2.1 GitOps 통합 — 변경 이력이 곧 감사 증빙

다섯 사례 모두 GitOps 도구를 멀티 사이트 매니페스트 동기의 단일 진실 원본(SSoT) 으로 두었습니다. GitOps 의 본질은 운영 환경의 선언된 상태를 Git 저장소에 정의하고, GitOps 도구가 Git 저장소와 클러스터 상태의 차이를 지속 동기하는 방식입니다. Argo CD 같은 GitOps 도구는 ApplicationSet 같은 멀티 클러스터 배포 추상을 제공하므로, 단일 매니페스트가 여러 사이트의 클러스터에 동일하게 적용됩니다 [S4]. 이 모델이 다섯 사례에서 공통으로 등장한다는 점은 우연이 아니며, 멀티 사이트 운영의 본질적 요구사항과 정합되기 때문입니다.

GitOps 통합이 만드는 효과는 셋입니다. 첫째, 변경 이력이 곧 감사 증빙입니다. 모든 클러스터 변경은 Git 커밋 이력으로 박제되므로, 누가·언제·무엇을 바꿨고 어느 사이트에 어떤 영향을 주었는지가 한 줄로 답해집니다.

ING Bank 사례에서 금융 감독 요구의 변경 통제 항목이 GitOps 도입으로 자연스럽게 충족되었다는 보고는 [8:2], 공공기관 ISMS-P 통제와도 같은 구조로 매핑됩니다. 둘째, 멀티 사이트 페일오버 절차가 GitOps 동기로 환원됩니다. 가상화 시대의 7단계 절차가 1단계(트리거) + GitOps 동기 + 트래픽 전환의 3단계로 정리되며, 인적 개입 단계가 트리거 하나로 줄어듭니다. 셋째, 인수인계 시점의 정보 손실이 줄어듭니다. JPMorgan Chase 사례에서 글로벌 SRE 인수인계의 정보 손실이 유의미하게 줄었다는 보고는 [9:1], GitOps 매니페스트가 인수인계의 단일 출처 역할을 한다는 본질에서 비롯됩니다.

자사 도입 시 첫 결정 항목은 GitOps 도구의 채택 자체보다 단일 Git 저장소의 거버넌스 모델입니다. 누가 매니페스트를 변경할 권한을 갖고, 변경 승인 절차는 어떻게 정하며, 변경 이력 보존 기간과 외부 감사 접근 절차를 어떻게 정의할지 결정 항목을 회의실에서 한 줄씩 정리하는 것이 출발점입니다. 도구 선택은 그다음입니다.

8.2.2 멀티 클러스터 거버넌스 도구 — 정책의 일원화

다섯 사례 모두 멀티 클러스터 거버넌스 도구를 도입해 여러 사이트의 정책·인증·접근 통제를 일원화했습니다. 멀티 클러스터 거버넌스 도구는 두 사이트 이상의 쿠버네티스 클러스터를 단일 콘솔에서 관리하면서, 정책 정의·인증서 발급·역할 기반 접근 통제(RBAC, Role-Based Access Control) 를 일원화합니다. 본 백서의 vendor-neutral 관점에서 보면 다섯 사례가 채택한 도구는 모두 다르지만, 도구가 수행하는 역할은 같습니다. 정책 일관성이 단일 사이트 운영에서는 부담이 크지 않지만, 두 사이트 이상으로 확장되는 순간 운영 부담이 비선형으로 증가하기 때문입니다.

멀티 클러스터 거버넌스 도구가 만드는 효과는 둘입니다. 첫째, 정책 표류(policy drift) 가 차단됩니다. 두 사이트의 정책이 시간이 지나면서 미묘하게 갈라지는 표류 현상은 가상화 시대 멀티 사이트 운영의 고질적 문제였으며, 거버넌스 도구는 정책 정의를 단일 매니페스트로 두고 모든 사이트에 동일 적용함으로써 표류를 차단합니다. 둘째, 인증·접근 통제가 일원화됩니다. ING Bank 사례에서 금융 감독 요구의 접근 통제가 멀티 클러스터 거버넌스로 자연스럽게 충족되었다는 보고는 [8:3], 공공기관 망분리 환경에서도 같은 구조로 매핑됩니다.

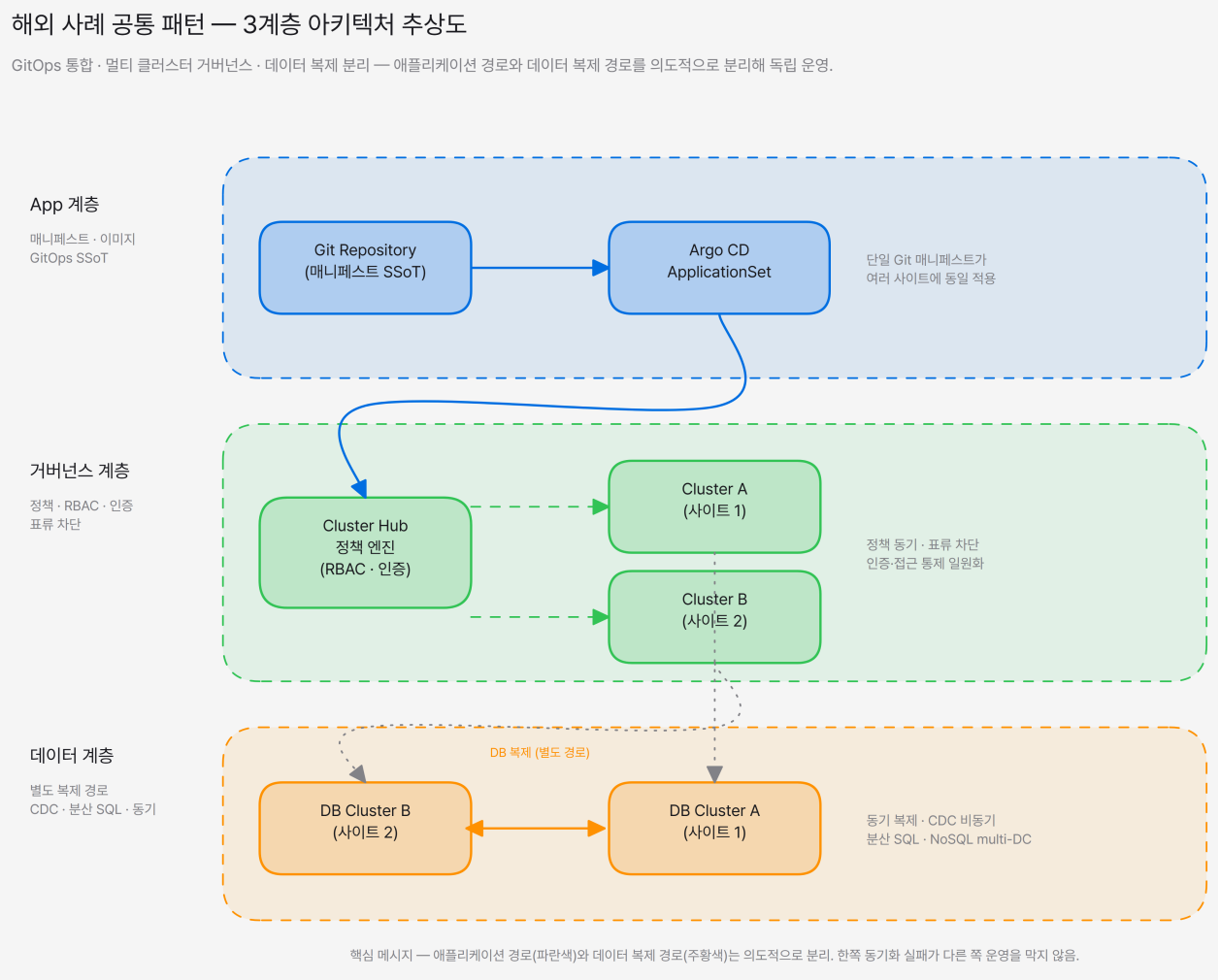
자사 도입 시 결정 항목은 거버넌스 도구의 채택 시점입니다. 두 사이트 이상의 운영이 확정된 시점에 도구 도입을 결정하면 정책 표류가 이미 시작된 상태에서 도구를 얹어야 하므로 초기 부담이 큽니다. 멀티 사이트 운영을 계획하는 시점에 거버넌스 도구를 함께 결정하는 것이 권장됩니다.

8.2.3 애플리케이션과 데이터 복제 경로의 분리

다섯 사례 모두 애플리케이션 복제 경로(이미지·매니페스트) 와 데이터 복제 경로(트랜잭션·로그) 를 의도적으로 분리해 설계했습니다. 가상화 시대의 백업·복원 모델은 VM 이미지 하나에 OS·미들웨어·애플리케이션·데이터가 함께 들어 있는 경우가 흔했고, 따라서 복제 경로도 하나로 통합되어 있었습니다. 컨테이너 환경에서는 이미지(불변)·매니페스트(선언)·데이터(상태) 세 종류가 명확히 분리되며, 멀티 사이트 운영에서는 각 경로에 다른 동기화 전략이 적용됩니다 [S1].

이 분리가 만드는 효과는 셋입니다. 첫째, 동기화 전략의 자유도가 높아집니다. 이미지는 컨테이너 레지스트리 Geo-Replication 정책으로, 매니페스트는 GitOps 도구로, 데이터는 워크로드 유형에 맞는 동기화 카테고리(동기 복제 / CDC 비동기 / 분산 SQL / NoSQL multi-DC) 로 각각 최적화됩니다. 4장에서 정리한 4 카테고리 매트릭스가 이 자유도의 구체 적용입니다. 둘째, 회선 단절·일시 오류의 영향 범위가 좁아집니다. 한 경로의 동기화 실패가 다른 경로의 운영을 막지 않으므로, 운영팀이 사고를 좁은 범위로 진단·복구할 수 있습니다. 셋째, 사

이버 재해 시나리오에서 신뢰 부트(trusted boot) 가 가능해집니다. 이미지가 불변이고 서명 검증이 자동화되어 있으므로, 데이터 복제 경로가 오염되어도 이미지 경로의 신뢰성은 그대로 유지됩니다 [S4].



자사 도입 시 결정 항목은 세 복제 경로의 책임 분담입니다. 가상화 시대에는 인프라 운영 조직이 모든 복제를 통합 관리했지만, 컨테이너 환경에서는 이미지 경로(플랫폼 엔지니어링), 매니페스트 경로(애플리케이션 팀 + 플랫폼 엔지니어링), 데이터 경로(데이터베이스 엔지니어링) 가 명시적으로 분담됩니다. 이 분담 모델을 결정하지 않은 상태에서 도구만 도입하면 운영 책임의 공백이 생기므로, 도구 결정 이전에 책임 모델 결정이 선행되어야 합니다.

다섯 사례의 운영 변화 항목에서 공통으로 등장한 SRE·플랫폼 엔지니어링 중심 재편은 이 세 복제 경로의 분담 모델과 정합됩니다. 별도 DR 운영조가 사라지는 대신, 플랫폼 엔지니어링 팀이 이미지·매니페스트 경로의 평시·DR 통합 운영을 맡고, 데이터베이스 엔지니어링 팀이 데이터 경로의 평시·DR 통합 운영을 맡습니다. 이 조직 모델 재편은 인력 감축이 아니라 인력 배치 변화이며, 다섯 사례에서 일관되게 야간·주말 사고 대응 안정성 향상으로 보고되었습니다 [6:2] [7:2] [8:4] [9:2] [10:2].

다섯 사례에서 일관되게 보고된 야간·주말 사고 대응 안정성 향상은 인력 감축이 아니라 인력 배치 변화의 결과입니다. 자사 DR 운영 조직이 평시 운영 조직과 분리되어 있는지, 통합되어 있는지를 한 줄로 진단하고 통합 시점의 18개월 전후 변화 지표(야간 호출 횟수, MTTR, 인수인계 시점 정보 손실) 를 자사 KPI 에 추가하는 일이 본 장을 읽고 다음 1주일 안에 실행 가능한 첫 번째 행동입니다.

9장: 국내 공공기관 DR 의 클라우드 네이티브 채택 근거

국내 공공기관의 재해복구(DR) 체계 운영은 사업자의 선택 영역이 아니라 법령이 정한 의무 영역이다. 본 장은 「전자정부법」 제49조와 시행령이 규정한 재해복구체계 구축 의무에서 출발하여, 행정안전부의 정량 등급 지침, 디지털플랫폼정부위원회(DPG, Digital Platform Government)의 클라우드 네이티브 전환 로드맵, 한국지능정보사회진흥원(NIA, National Information Society Agency)의 적용 가이드, 한국인터넷진흥원(KISA, Korea Internet & Security Agency)의 ISMS-P(Information Security Management System - Personal information protection, 개인정보 및 정보보호 관리체계) 인증 기준까지 5개 정책 문서를 한 줄에 놓고 비교한다. 다섯 문서의 요구사항을 한 표로 모으면 한 가지 결론이 드러난다 — 국내 공공기관의 클라우드 네이티브 DR 전환은 단순한 기술 선택이 아니라 정책 정합성을 충족하기 위한 정해진 경로다.

9.1 법령·지침 근거

재해복구체계 의무의 법령 출발점과 그 의무를 실무 등급으로 환산한 행정안전부 지침은 함께 읽어야 의미가 통한다. 「전자정부법」은 "재해복구체계를 구축하라"는 일반 의무를 지우고, 행정안전부 지침은 그 일반 의무를 4개 등급으로 정량화하여 시스템별로 어느 등급을 적용할지를 결정하는 기준을 제공한다.

9.1.1 「전자정부법」 제49조와 시행령의 재해복구체계 의무

「전자정부법」 제49조 (정보시스템의 안정성·신뢰성 제고) 와 그 시행령은 국가기관과 지방자치단체, 그리고 동법 적용 대상 공공기관이 운영하는 정보시스템에 대해 재해·장애 발생에 대비한 복구체계를 사전에 구축·운영하도록 정하고 있다 [S9]. 조항의 적용 범위는 행정정보 처리·저장 시스템 전체이며, 이는 민원 처리, 행정업무, 공공서비스 제공을 위한 모든 정보시스템을 포함한다. 같은 법 시행령은 이 의무를 실무 절차로 풀어내면서 재해복구 계획 수립, 정기 훈련 실시, 복구 시간·시점 목표의 사전 정의, 그리고 백업·복제 체계의 운영을 요구한다.

법령 조항의 핵심은 "사전 구축" 이라는 표현에 있다. 재해가 발생한 다음에 복구 수단을 마련하는 사후 대응이 아니라, 평시 정상 운영 단계에서 이미 복구 가능한 상태로 시스템을 설계·운영해야 한다는 점이다. 이 원칙은 본 백서 1장에서 정의한 클라우드 네이티브 DR 의 "선언된 상태의 재현" 모델과 자연스럽게 맞물린다. 매니페스트로 박은 상태가 항상 두 사이트에서 일치하도록 GitOps 동기 루프가 작동하는 환경에서는 "사전 구축" 의무가 별도 운영 절차가 아니라 평시 배포 절차의 부산물로 충족된다 [S4]. 반대로 시점 백업 후 복원 방식의 가상화 DR 은 백업 매체 보관, 복원 절차 문서화, 정기 훈련을 모두 별도 프로젝트로 관리해야 하며 이는 법령이 요구하는 "사전 구축" 의무의 실질을 사후 운영 노력에 의존하게 만든다.

위반 시 책임은 「전자정부법」 자체의 행정 조치 외에도 감사원 감사, 국정감사, 그리고 ISMS-P 같은 인증 갱신 평가에서 직접 지적 대상이 된다. 특히 행정정보 사고가 발생한 경우 재해복구체계의 사전 구축 여부는 책임 소재 판단의 1차 기준이 된다. 따라서 정책결정권자가 이 조항을 인용할 때는 단순한 IT 운영 규정이 아니라 기관장의 행정 책임 영역으로 다뤄야 한다.

법령 조항의 또 다른 의의는 그 모호성에 있다. 제49조와 시행령은 "재해복구체계를 구축하라" 고만 정할 뿐 구체적 기술 요건은 제시하지 않는다. 그 빈 칸을 채우는 것이 행정안전부 지침이며, 더 나아가 디지털플랫폼정부

로드맵과 NIA 가이드다. 즉 클라우드 네이티브 DR 의 채택 근거를 법령 단일 문서에서 찾으려는 시도는 부적절하며, 본 장이 다섯 문서를 함께 모아 비교하는 이유도 이 때문이다.

9.1.2 행정안전부 「공공기관 정보시스템 재해복구 체계 구축 지침」

행정안전부의 「공공기관 정보시스템 재해복구 체계 구축 지침」은 「전자정부법」 제49조의 일반 의무를 4개 등급으로 정량화한 실무 기준이다 [S10]. 지침은 정보시스템의 업무 영향도와 중단 시 사회적 파급을 기준으로 핵심 1등급에서 보통 4등급까지 분류하며, 등급별로 복구 시간 목표(RTO, Recovery Time Objective)와 복구 시점 목표(RPO, Recovery Point Objective)를 다르게 적용한다. 일반적으로 핵심 등급일수록 RTO 는 분 단위에 가깝고 RPO 는 0 에 수렴해야 하며, 보통 등급에서는 시간 단위 RTO 와 일 단위 RPO 도 허용된다.

등급 분류가 클라우드 네이티브 DR 의 정합성과 만나는 지점은 핵심 등급 시스템의 요구사항에 있다. 분 단위 RTO 와 0 에 가까운 RPO 를 가상화·물리 DR 모델로 충족하려면 동기식 스토리지 복제와 사전 가동된 대기 시스템이 동시에 필요하며, 평시 자원 활용률은 절반 이하로 떨어진다. 반면 클라우드 네이티브 DR 의 Active-Active 구조는 평시에도 두 사이트 모두 트래픽을 처리하면서 한 사이트 장애 시 트래픽 전환만으로 분 단위 RTO 를 달성한다 [S4]. 본 백서 5장과 6장이 정량화한 한 자릿수 분 RTO 와 자원 약 30~60% 절감은 그대로 핵심 등급 충족 수단으로 매핑된다.

지침의 또 다른 핵심은 정기 훈련 요구사항이다. 등급별 훈련 주기와 검증 절차가 명시되어 있으며 훈련 결과를 문서화하여 보존해야 한다. 시점 복원 모델의 가상화 DR 은 훈련 자체가 별도 인적·시간 자원을 요구하는 이벤트로 운영되어 분기·반기 단위 실시가 한계다. 클라우드 네이티브 DR 의 선언적 재현 모델에서는 Argo CD ApplicationSet 의 매니페스트 동기와 함께 Chaos Mesh 같은 도구로 페일오버 시나리오를 평시 운영에 통합할 수 있어 훈련 빈도가 일 단위·주 단위로 올라가도 운영 부담이 선형 증가하지 않는다. 즉 지침이 요구하는 정기 훈련 의무를 충족하는 가장 자연스러운 운영 모델이 클라우드 네이티브 DR 이다 [S4][S10].

행정안전부 지침의 등급별 요구사항을 「전자정부법」 조항 및 후속 정책 문서와 한 표로 모으면 다음과 같다.

국내 법·정책 5건 · 클라우드 네이티브 DR 요구사항 매핑

전자정부법 제49조 · 행안부 재해복구 지침 · DPG 로드맵 · NIA 클라우드 네이티브 가이드 · KISA ISMS-P 재해복구. 각 문서의 핵심 요구사항과 클라우드 네이티브 DR 정합을 한 표로 정리.

법·정책	적용 대상	핵심 요구사항	측정 지표	클라우드 네이티브 DR 정합
「전자정부법」 제49조	공공기관 정보시스템 전반	재해복구체계 의무 구축 (일반 의무)	RTO·RPO 등급 분류	직접 충족 (클라우드 네이티브 DR 전 구성 요소가 의무 이행)
행정안전부 재해복구 체계 구축 지침	정보시스템 등급별 (핵심~보통 4등급)	등급별 RTO·RPO 목표 + 정기 훈련 의무	시간(RTO)· 시점(RPO) 정량, 훈련 빈도	Active-Active 로 RTO 단축 + Argo CD·Chaos Mesh 로 훈련 일·주 단위 자동화
디지털플랫폼정부 전환 로드맵	공공부문 정보시스템 (신규·재구축 우선)	클라우드 네이티브 아키텍처 단계적 전환 (컨테이너·MSA 채택)	채택률· 전환 일정, 사업 평가 점수	본 백서가 권고하는 참조 모델 그 자체 (미채택 시 입증 책임)
NIA 클라우드 네이티브 가이드	공공기관 신규·전환 사업 (실무 적용)	컨테이너·MSA 도입 절차, 레퍼런스 아키텍처, 검증 항목	단계별 체크리스트 충족률	직접 적용 (가이드의 실행 청사진을 그대로 사용)
KISA ISMS-P 재해복구 통제	ISMS-P 인증 대상 조직 (공공·민간)	DR 통제 항목 (복구 절차·매체 보관· 무결성 검증)	통제 충족도 (인증 심사 기준)	이미지 불변성·서명· 매니페스트 동기로 통제 자동 충족

녹색 열 = 클라우드 네이티브 DR 정합 (다섯 정책 모두에서 의무·권고를 직접 또는 자동으로 충족)

표는 다섯 정책 문서(전자정부법 제49조, 행안부 재해복구 지침, DPG 로드맵, NIA 가이드, KISA ISMS-P 재해복구 항목) 의 핵심 요구사항을 행으로, 각 요구사항이 클라우드 네이티브 DR 구성 요소(매니페스트 동기, 이미지 불변성, 서명 검증, Active-Active 트래픽 분기, 데이터 복제, 정기 훈련 자동화) 와 어떻게 매핑되는지를 열로 정리한다. 이 표 한 장이 본 장의 결론 자료이며, 정책결정권자가 RFP 평가 항목과 거버넌스 보고서에 그대로 옮길 수 있는 정합성 근거다.

표의 활용 시 주의할 점은 행정안전부 지침의 등급은 사업자 자율이 아니라 기관 분류에 따른다는 사실이다. 따라서 등급 분류 자체를 변경하려는 시도가 아니라 지정된 등급의 요구사항을 가장 효율적으로 충족하는 기술 수단을 선택하는 의사결정으로 이 표를 사용해야 한다. 핵심 등급일수록 클라우드 네이티브 DR 의 정합성이 높고, 보통 등급에서도 평시 자원 활용률·라이선스 절감 측면의 정합성은 유지된다.

9.2 정책 로드맵과 인증 기준

법령·지침이 정의한 의무를 어떤 기술 수단으로 충족할지는 세 후속 문서가 권고·요구한다. 디지털플랫폼정부위원회의 전환 로드맵과 NIA 의 적용 가이드는 클라우드 네이티브 전환의 방향과 절차를 제시하며, KISA 의 ISMS-P 인증 기준은 그 운영 결과가 정보보호 통제로 어떻게 검증되는지를 정한다. 세 문서를 함께 읽으면 클라우드 네이티브 DR 이 단순 기술 선택이 아니라 정책·인증 양면에서 검증되는 정합 경로임이 드러난다.

9.2.1 디지털플랫폼정부 로드맵과 NIA 가이드

디지털플랫폼정부위원회의 「공공부문 클라우드 네이티브 전환 로드맵」은 중앙정부·지방자치단체·공공기관의 정보시스템을 단계별로 클라우드 네이티브 아키텍처로 전환하는 일정과 평가 체계를 제시한다 [S11]. 로드맵의 핵심은 신규 사업과 재구축 사업의 우선 전환, 기존 시스템의 단계적 현대화, 그리고 전환 효과를 측정하는 평가 지표(자원 효율, 운영 자동화 수준, 복구 시간 단축, 라이선스 절감)이다. 이 평가 지표는 본 백서 5장·6장에서 정량화한 클라우드 네이티브 DR 의 우위 4지표와 직접 맞물린다.

로드맵이 단순 권고를 넘어 실질적 강제력을 갖는 이유는 공공정보화 사업 평가 기준에 반영되기 때문이다. 사업 기획·예산 심의 단계에서 클라우드 네이티브 전환 적합성이 평가 항목으로 포함되며, RFP 발주 시 평가 위원이 이 항목에 점수를 부여한다. 따라서 정책결정권자가 신규 사업이나 재구축 사업을 기획할 때 클라우드 네이티브 DR 을 채택하지 않는다면 그 선택의 사유를 평가 자료에 명시해야 하는 입증 책임이 발생한다. 채택의 입증 책임이 아니라 미채택의 입증 책임이 더 무거워진 상황이다.

NIA 의 「공공부문 클라우드 네이티브 적용 가이드」는 로드맵의 권고를 실무 절차로 구체화한다 [S11]. 가이드는 컨테이너 런타임·오케스트레이션·CI/CD·관측 가능성·보안·DR 의 6개 영역에 걸쳐 기술 요건과 운영 절차를 제시하며, 각 영역에서 채택 가능한 CNCF 프로젝트 군을 vendor-neutral 관점에서 안내한다. DR 영역에서는 본 백서 1장이 정의한 GitOps 매니페스트 + 이미지 불변성 + 데이터 복제 3단 구조가 그대로 반영되어 있다.

가이드의 실무 활용 방식은 두 가지다. 첫째, 신규 사업 RFP 작성 시 가이드의 영역별 요건을 평가 항목 골자로 그대로 옮긴다. 평가 위원이 가이드와 정합되는 응답을 기대하기 때문에 응답 사업자도 가이드 기반 제안을 제출하게 된다. 둘째, 기존 시스템 현대화 사업의 마일스톤을 가이드의 6개 영역 단계별 충족도로 설정한다. 이 방식은 진행률을 정량 측정 가능하게 하며 사업 종료 시점의 검증 부담을 낮춘다.

로드맵과 가이드의 영향력은 적용 일정에 의해 강화된다. 신규 사업은 즉시 적용 대상이며, 기존 사업은 시스템 등급과 노후도에 따라 단계적 전환 시점이 정해져 있다. 정책결정권자가 다음 12~24개월 안에 의사결정해야 할 항목 중 하나가 자기관 시스템 포트폴리오의 전환 우선순위 결정이며, 이 결정의 기준 자료가 곧 로드맵과 가이드다. 즉 클라우드 네이티브 DR 의 채택 시점은 사업자가 자율적으로 정하는 것이 아니라 정책 일정과 사업 평가 사이클이 정한다.

9.2.2 KISA ISMS-P 재해복구 항목과 정합

KISA 의 ISMS-P 인증 기준은 정보보호 관리체계(ISMS)와 개인정보보호 관리체계(PIMS) 를 통합한 국내 대표 정보보호 인증이며, 공공기관과 일정 규모 이상의 민간 사업자에게 적용된다 [S10]. 인증 기준의 재해복구 영역은 재해복구 계획 수립, 백업·복제 운영, 정기 훈련, 사이버 재해(랜섬웨어 등) 대응 절차, 그리고 감사 추적 기록 보존을 요구한다. 각 항목은 통제(control) 단위로 정의되며 인증 심사 시 통제별 증적 자료가 점검된다.

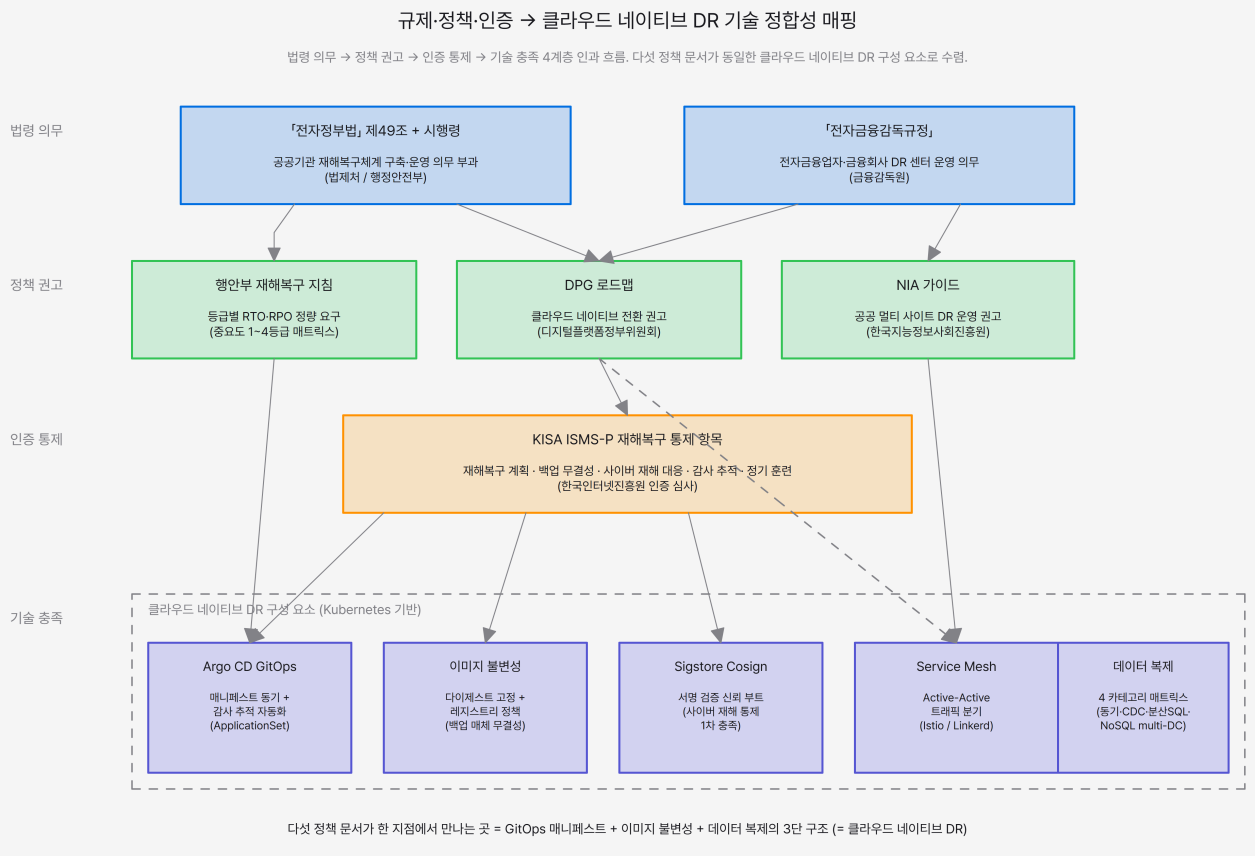
ISMS-P 통제와 클라우드 네이티브 DR 운영의 정합성은 세 가지 메커니즘으로 드러난다. 첫째, 이미지 불변성과 Sigstore Cosign 서명 검증은 사이버 재해 통제 항목의 1차 충족 수단이다. 본 백서 6장이 정리한 신뢰 부트 우위 — 백업 매체 자체의 오염 가능성을 차단하고 서명 검증으로 신뢰 기준을 확정하는 메커니즘 — 는 그대로 ISMS-P 의 사이버 재해 대응 통제 증적이 된다. 가상화 환경에서는 백업 매체 무결성을 별도 검증 체계로 입증해야 하지만, 컨테이너 환경에서는 이미지 다이제스트와 서명이 곧 무결성 증적이다.

둘째, GitOps 매니페스트는 감사 추적 기록 보존 통제의 자동 충족 수단이다. Argo CD ApplicationSet 이 적용한 모든 매니페스트 변경은 Git 커밋 이력으로 보존되며, 누가 언제 어떤 변경을 적용했는지가 자동으로 기록된다 [S4]. ISMS-P 가 요구하는 감사 추적 기록을 별도 도구로 수집·보존할 필요 없이 운영 절차 자체가 증적을 생성한다. 이 변화는 인증 심사 준비 비용을 정량으로 낮춘다.

셋째, 정기 훈련 통제와 클라우드 네이티브의 카오스 엔지니어링 운영이 자연스럽게 들어맞는다. ISMS-P 는 재해복구 훈련의 정기 실시와 결과 문서화를 요구하는데, 가상화 환경에서는 훈련 자체가 운영 중단을 동반하는 이벤트로 운영되어 빈도가 낮다. 클라우드 네이티브 환경에서는 페일오버 시나리오를 매니페스트로 미리 정의해 두고 자동 실행할 수 있어 훈련 빈도와 결과 문서화 모두 운영 자동화의 부산물이 된다.

이 세 메커니즘을 합치면 한 가지 결론에 도달한다 — 이미지 불변성 + Sigstore Cosign 서명 검증 + GitOps 매니페스트의 3단 구조는 ISMS-P 통제 자동화의 1차 후보다. 통제 자동화는 단순한 운영 효율 향상이 아니라 인증 갱신 사이클의 심사 부담과 비용 절감으로 직결된다. 정보보호 책임자(CISO) 가 다음 인증 갱신 안건으로 통제 자동화 후보 목록을 올릴 때 가장 먼저 검토할 후보가 클라우드 네이티브 DR 의 3단 구조다.

세 정책 문서(DPG 로드맵, NIA 가이드, KISA ISMS-P 재해복구 항목) 와 본 장에서 다룬 두 법령·지침(전자정부법 제49조, 행안부 지침) 의 요구사항을 클라우드 네이티브 DR 의 구성 요소와 한 그림으로 매핑하면 다음과 같다.



도식은 좌측에 다섯 정책 문서, 우측에 클라우드 네이티브 DR 구성 요소(매니페스트 동기, 이미지 불변성, 서명 검증, Active-Active 트래픽 분기, 데이터 복제, 정기 훈련 자동화) 를 두고 두 영역 사이의 정합 관계를 화살표로 연결한다. 이 도식은 본 장의 결론 시각화이며, 본 백서 전체에서 정책결정권자가 임원·이사회 보고에 그대로 사용할 수 있는 단일 자료다.

도식이 전달하는 핵심 메시지는 다음과 같다. 다섯 정책 문서는 서로 다른 발행처(법제처, 행정안전부, 디지털플랫폼정부위원회, NIA, KISA) 에서 나왔지만 요구사항을 한 그림에 모으면 클라우드 네이티브 DR 의 동일한 기술 구성 요소로 수렴한다. 즉 국내 공공기관이 다섯 문서를 별도로 대응해 온 관행을 클라우드 네이티브 DR 의 단일 운영 모델로 통합하면 정책 정합성을 충족하면서도 운영 부담은 줄어든다.

본 장의 결론은 다음과 같이 정리할 수 있다. 「전자정부법」 제49조와 시행령이 부과한 재해복구체계 구축 의무는 행정안전부 지침을 통해 등급별 정량 요구사항으로 구체화되며, 디지털플랫폼정부 로드맵과 NIA 가이드는 그 달성 수단으로 클라우드 네이티브 전환을 권고·요구하고, KISA ISMS-P 는 그 운영 결과를 정보보호 통제로 검증한다. 다섯 문서가 한 지점에서 만나는 곳이 클라우드 네이티브 DR 의 3단 구조(GitOps 매니페스트 + 이미지 불변성 + 데이터 복제) 이며, 따라서 국내 공공기관의 클라우드 네이티브 DR 전환은 기술 선택이 아니라 정책 정합성을 충족하기 위한 정해진 경로다 [S9][S10][S11]. 정책결정권자가 다음 분기 의사결정 안건으로 올려야 할 항목은 "전환 여부" 가 아니라 "전환 우선순위와 일정" 이다.

10장: 도입 권장과 기술적 우위 요약

클라우드 네이티브 기반 DR(Disaster Recovery, 재해복구) 은 단일 기술 선택이 아니라 운영 모델 전환이며, 그 전환의 의사결정은 한 사람이 한 번에 내리는 것이 아니라 거버넌스·워크로드 분류·데이터 복제 모델·다음 검증 후보·인력 재편 5개 영역의 결정 옵션을 거버넌스 보드(governance board) 가 함께 합의하는 과정이다. 본 장은 1장부터 9장까지 입증한 기술적 우위를 정량 4지표와 정성 네 측면으로 종합한 뒤, 5개 영역의 결정 옵션과 권장을 의사결정 매트릭스로 압축해 다음 분기 회의 안건의 골격을 제공한다. 일자나 주차 단위의 행동 계획은 다루지 않는다 — 그것은 각 기관이 자체 환경 변수를 채워 결정할 영역이다.

10.1 기술적 우위 종합 요약

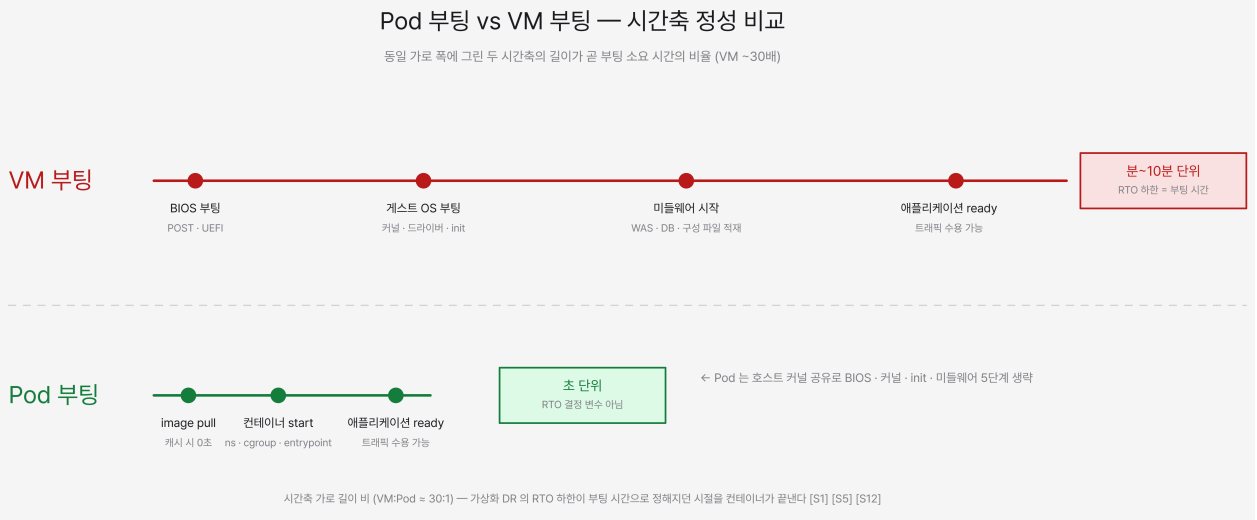
기술적 우위는 정량 4지표 — 복구 시점, 복구 시간, 자원 절감, 라이선스 절감 — 와 정성 네 측면 — 자동화 수준, 사이버 재해 신뢰 부트, 국내 정책 정합성, 플랫폼 엔지니어링 인력 모델 — 의 두 갈래로 종합된다. 두 갈래는 분리된 평가 축이 아니라 같은 모델 전환이 만들어낸 두 가지 표현이다. 정량 우위가 회의실 보고의 한 슬라이드 결론이 된다면, 정성 우위는 그 결론이 한 분기에 끝나지 않고 운영 모델 전체로 확산되는 메커니즘을 설명한다.

10.1.1 정량 우위 — 복구 시점·시간·자원·비용 4지표

정량 4지표는 복구 시점(RPO, Recovery Point Objective), 복구 시간(RTO, Recovery Time Objective), 자원 절감, 라이선스 절감이며, 각 지표는 가상화 기준 모델과 클라우드 네이티브 모델을 동일 워크로드 기준으로 비교한다. 자사 환경 변수로 즉시 환산할 수 있도록 비교는 비율과 단위 변화에 초점을 맞춘다.

먼저 복구 시점 지표를 살펴본다. 가상화 DR의 시점 백업 모델은 백업 주기가 곧 복구 시점의 하한이며, 일 단위 백업이라면 복구 시점이 하루 전으로 후퇴하는 구조다. 클라우드 네이티브 DR은 GitOps(매니페스트 기반 선언적 배포) 매니페스트가 git 저장소에 커밋되는 순간이 곧 신뢰 시점이며, 데이터 복제는 워크로드 유형별로 4 카테고리(동기 복제·CDC 비동기·분산 SQL·NoSQL multi-DC) 매트릭스(4장 참조)에서 선택한다. 동기 복제 카테고리에서는 복구 시점이 사실상 0에 가깝고, CDC 비동기에서는 양 사이트 회선 지연만큼만 후퇴한다. 본 백서 4장에서 입증한 매트릭스 선택이 곧 복구 시점 지표의 결정 근거다.

복구 시간 지표는 더 극적인 변화를 보인다. 본 백서 6장에서 가상화·물리 DR의 7단계(백업 매체 위치 확인 → 하드웨어 할당 → 운영체제 부팅 → 미들웨어 재구성 → 애플리케이션 복구 → 데이터 적재 → 네트워크 재설정)와 클라우드 네이티브 DR의 3단계(페일오버 트리거 → Argo CD ApplicationSet 동기화 → DNS·Service Mesh 트래픽 전환)를 평행 배치한 결과, 단계 수의 차이가 곧 시간의 차이로 나타났다. Pod 부팅은 초 단위로 끝나는 반면 가상머신 부팅은 분에서 10분 단위로 걸린다(5장



참조). 단계마다 인적 개입이 필요한 가상화·물리 모델은 야간이나 주말의 호출 가용성에도 영향을 받지만, 클라우드 네이티브 모델의 3단계는 모두 자동화 가능하며 인적 개입은 트리거 1회로 끝난다 [S4]. 본 백서가 인용한 시장 분석 리포트는 컨테이너 기반 멀티 사이트 DR의 복구 시간 단축이 한 자릿수 분 단위로 측정된다는 사례를 제시하며, 인적 개입 단계 수의 감소가 곧 호출 비용 절감으로 환산된다는 점도 함께 보인다 [S12].

자원 절감 지표는 평시 운영 패턴에서 확인된다. Kubernetes(컨테이너 오케스트레이션) 스케줄러의 Bin Packing(자원 통합 배치)과 HPA(Horizontal Pod Autoscaler, 수평 자동 확장)·VPA(Vertical Pod Autoscaler, 수직 자동 확장)가 결합되어 동일 워크로드를 가상화 환경 대비 약 30~60% 적은 자원으로 운영할 수 있다(5장 참조). DR 사이트가 평시에 놀려두는 자원이 아니라 Active-Active 모델에서 평시 가용 자원의 일부로 흡수되는 구조 변화는 회의실의 비용 인식을 바꾼다. 본 백서 1장과 5장에서 입증한 컨테이너 채택률 추세는 이 변화가 일회성 절감이 아니라 운영 모델 전환의 결과임을 뒷받침한다 [S1]. 시장 분석 리포트의 산업별 비용 비교 자료도 동일 방향을 가리킨다 [S12].

라이선스 절감 지표는 가장 회의실 친화적인 항목이다. 가상화 환경의 3단 라이선스(하이퍼바이저, 게스트 운영체제, 미들웨어)는 컨테이너 환경에서 이미지 안으로 흡수되거나 단일 계층으로 단순화된다(5장 참조). 동일 워크로드를 컨테이너로 옮길 때 발생하는 라이선스 절감은 도입 결정의 1차 ROI(Return on Investment, 투자 수익) 항목이며, 본 백서 5장에서 100 vCPU 워크로드 기준 환산표 형식으로 정리한 바 있다. 4지표를 한 표로 정리하면 다음과 같다.

지표	가상화 기준 모델	클라우드 네이티브 모델	단위 변화
복구 시점 (RPO)	일 단위(시점 백업)	0~회선 지연	일 → 초·분
복구 시간 (RTO)	수 시간~수 일	한 자릿수 분	시간 → 분
자원 절감	기준 100	40~70	30~60% 절감
라이선스 절감	3단 라이선스	1~2단 단순화	계층 수 감소

본 표는 자사 환경 변수(워크로드 수, 평시 트래픽, 코어 단가)를 곱해 정량 ROI로 환산할 수 있는 빈 칸을 함께 제공한다. 4지표는 임원이나 이사회 보고의 한 슬라이드로 그대로 활용 가능한 정보 밀도를 유지한다.

10.1.2 정성 우위 — 자동화·신뢰 부트·정책 정합성·인력 모델

정량 4지표가 회의실 보고의 결론이라면 정성 네 측면은 그 결론을 운영 모델 전반으로 연결하는 메커니즘이다. 네 측면은 자동화 수준, 사이버 재해 신뢰 부트, 국내 정책 정합성, 플랫폼 엔지니어링 인력 모델이며, 본 백서 1장, 6장, 8장, 9장의 분석이 한 자리로 수렴한다.

첫째 측면은 자동화 수준이다. 가상화 DR이 시점 복원 모델이라면 클라우드 네이티브 DR은 선언된 상태의 재현 모델이다(1장 참조). 선언된 상태는 git 저장소의 매니페스트로 표현되며 Argo CD(GitOps 컨트롤러)가 그 상태와 실제 클러스터 상태의 차이를 지속적으로 좁힌다 [S4]. CNCF 연례 조사 결과는 GitOps 기반 운영의 표준화가 운영 환경 전반으로 확산되었음을 보인다 [S1]. 이 구조는 DR을 별도 운영 영역에서 평시 운영의 부산물로 이동시킨다. 평시에 매니페스트를 한 번도 손대지 않은 가상화 환경에서는 재해 시점에야 비로소 절차서를 펼치지만, GitOps 환경에서는 매일의 배포가 곧 DR 훈련이다. 본 백서 1장에서 정의한 평시 운영의 부산물이라는 개념의 실무 의미는 바로 이 자동화 수준의 상수화에 있다.

둘째 측면은 사이버 재해 신뢰 부트(trusted boot)다. 본 백서 6장에서 다룬 랜섬웨어와 같은 사이버 재해 시나리오에서 가상화 환경은 백업 매체 자체의 오염 가능성을 의심해야 한다. 컨테이너 환경은 이미지 불변성과 Sigstore Cosign(이미지 서명 도구) 서명 검증으로 신뢰 부트의 기반을 확립한다(3장과 6장 참조). 정책 엔진(Kyverno)이 서명 미검증 이미지의 클러스터 배포를 차단하는 메커니즘(7장 참조)은 사이버 재해 통제를 운영 시점에 자동화하는 1차 수단이다. 본 백서 9장에서 정리한 KISA ISMS-P(정보보호 관리체계) 재해복구 항목과의 정합성은 이 메커니즘이 인증 갱신 사이클의 부담을 줄이는 구조적 근거를 형성한다.

셋째 측면은 국내 정책 정합성이다. 본 백서 9장에서 정리한 「전자정부법」 제49조의 재해복구체계 구축 의무, 행정안전부 「공공기관 정보시스템 재해복구 체계 구축 지침」의 등급별 정량 요구사항, 디지털플랫폼정부위원회의 공공부문 클라우드 네이티브 전환 로드맵과 NIA(한국지능정보사회진흥원)의 공공부문 클라우드 네이티브 적용 가이드는 한 방향을 가리킨다 [S11]. 의무 조항은 복구 시점과 복구 시간의 정량 목표를 명시하고, 정책 로드맵과 가이드는 그 달성 수단으로 클라우드 네이티브 전환을 제시한다 [S11]. 본 백서 9장에서 입증한 정책 정합성은 도입 결정의 정성 근거 중 가장 회의실에서 인용 빈도가 높다.

넷째 측면은 플랫폼 엔지니어링(platform engineering) 인력 모델이다. 본 백서 8장에서 정리한 Adidas, Lufthansa Systems, ING Bank의 해외 사례에서 공통으로 관찰된 패턴 중 하나는 운영 인력 모델의 재편이었다 [S1] [S4]. 가상화 환경의 운영 인력은 인프라 운영팀, 백업 운영팀, DR 훈련 운영팀으로 분리되어 있었으나, 클라우드 네이티브 환경은 플랫폼 엔지니어링 팀이 GitOps·관측 가능성·정책 엔진을 한 플랫폼으로 통합 운영한다. 본 백서 6장에서 분석한 인적 개입 단계 수의 감소는 그 자체로 인력 모델 변화의 정량 근거이며, 절감된 인적 부담은 신규 워크로드 도입과 정책 자동화 영역으로 재배치된다 [S4]. 정성 네 측면을 한 표로 정리하면 다음과 같다.

측면	가상화 모델	클라우드 네이티브 모델	핵심 변화
자동화 수준	시점 복원 절차서	선언된 상태의 지속 재현	평시 운영의 부산물화
사이버 재해 신뢰 부트	백업 매체 오염 가능성	이미지 불변성 + 서명 검증	통제 자동화
국내 정책 정합성	등급별 절차 충족	등급 + 로드맵·가이드 정합	정책 정합성 확정
플랫폼 엔지니어링 인력	분리된 운영팀	통합 플랫폼 팀	인적 자원 재배치

정량 4지표와 정성 네 측면을 결합하면 의사결정권자가 회의실에서 인용할 기술적 우위의 종합이 완성된다. 그 다음 단계는 이 종합 결과를 5개 영역의 의사결정 매트릭스로 환원해 거버넌스 보드 회의의 의제로 만드는 일이

다.

10.2 의사결정 매트릭스 — 5개 영역의 결정 옵션

의사결정권자가 결정해야 할 5개 영역은 거버넌스, 워크로드 분류, 데이터 복제 모델, 다음 검증 후보, 인력 재편이며 — 다섯 결정은 상호 종속적이다. 거버넌스 모델이 결정되어야 워크로드 분류 권한이 정해지고, 워크로드 분류가 정해져야 데이터 복제 모델 카테고리 선택이 의미를 갖는다. 따라서 5개 영역을 한 매트릭스로 모아 거버넌스 보드 한 번의 회의에서 합의하는 방식이 결정 순서 충돌을 줄인다.

10.2.1 5개 영역의 결정 옵션과 권장

결정 옵션은 정답이 하나가 아니라 자사 거버넌스 환경에 따라 선택지가 갈리는 영역이며, 권장은 그 갈림길에서 본 백서가 입증한 정량·정성 우위와 정합하는 선택지를 지목하는 형태로 제시된다.

첫째 영역은 거버넌스 — 누가 결정하는가다. 결정 옵션은 세 가지로 분리된다. (a) CIO(Chief Information Officer, 최고정보책임자) 단독 결정, (b) DR 거버넌스 보드 공동 결정, (c) 정보보호 책임자 공동 결정. CIO 단독 결정은 의사결정 속도가 빠르지만 사이버 재해 신뢰 부트와 정책 정합성 평가에서 정보보호 책임자의 견제가 약하다. DR 거버넌스 보드 공동 결정은 본 백서 1장부터 9장까지 누적된 평가 축(자동화, 신뢰 부트, 정책 정합성, 인력 모델)을 모두 다룰 수 있는 의사결정 구조다. 정보보호 책임자 공동 결정은 사이버 재해 통제 자동화(7장)와 KISA ISMS-P 정합성(9장)을 1차 평가 기준으로 두는 보수적 모델이며, 국내 공공부문 정책 흐름과 정합한다 [S11]. 본 백서가 권장하는 모델은 (b) DR 거버넌스 보드 공동 결정이다. 5개 영역의 결정이 상호 종속적인 구조에서 한 사람의 결정 권한으로는 영역 간 트레이드오프를 합의하기 어렵다.

둘째 영역은 워크로드 분류 — 어떤 워크로드를 먼저 옮길 것인가다. 결정 옵션은 네 가지로 분리된다. (a) 트랜잭션 우선, (b) 세션 우선, (c) 분석 우선, (d) 신규 시스템 우선. 트랜잭션 우선 모델은 본 백서 4장에서 정리한 동기 복제 카테고리의 적용 부담이 가장 크다. 강한 일관성 요구 때문에 데이터 복제 모델 선택의 자유도가 좁아진다. 세션 우선 모델은 본 백서 2장에서 정리한 트래픽 분기 3계층(DNS, 부하분산기, Service Mesh) 중 Service Mesh 계층의 도입을 선결 조건으로 한다. 분석 우선 모델은 일관성 요구가 가장 낮아 CDC(Change Data Capture) 비동기 또는 NoSQL multi-DC 카테고리의 적용이 자연스럽지만, 회의실 가시성 측면에서 임팩트가 약하다. 신규 시스템 우선 모델은 분산 SQL 카테고리의 채택과 정합하며 운영 모델 전환의 진입 장벽이 가장 낮다 [S1]. 본 백서가 권장하는 모델은 (d) 신규 시스템 우선이며, 그 다음 (c) 분석, (b) 세션, (a) 트랜잭션의 순서로 확장한다. 신규 시스템 우선 모델은 기존 워크로드의 운영 위험을 건드리지 않으면서 플랫폼 엔지니어링 인력 모델을 단계적으로 형성할 수 있다.

셋째 영역은 데이터 복제 모델 — 4 카테고리 중 무엇을 선택할 것인가다. 결정 옵션은 본 백서 4장의 4 카테고리 매트릭스(동기 복제·CDC 비동기·분산 SQL·NoSQL multi-DC)이며, 단일 정답은 없다. 워크로드별로 카테고리를 분리 결정하는 것이 본 백서 4장의 권장이다. 트랜잭션 워크로드는 동기 복제(Patroni 기반 PostgreSQL, MySQL Group Replication, MariaDB Galera) 또는 분산 SQL(CockroachDB, YugabyteDB, TiDB) 후보가 자연스럽고, 세션 워크로드는 CDC 비동기 또는 NoSQL multi-DC 후보가 자연스럽다. 로그 워크로드는 CDC 비동기와 정합하며, 분석 워크로드는 NoSQL multi-DC 또는 분산 SQL 양쪽 모두 검토 가능하다. 본 백서가 권장하는 의사결정 절차는 워크로드별로 4 × 4 매트릭스(카테고리 × 워크로드)에 자사 약어를 직접 채워 가는 워크북 형식이며, 그 결과는 거버넌스 보드 의결로 확정한다. GitOps 매니페스트 기반 운영은 워크로드별 복제 카테고리 선택을 매니페스트 안에 표현 가능하게 한다 [S4].

넷째 영역은 다음 검증 후보 — 어떤 범위로 검증을 진행할 것인가다. 결정 옵션은 검증 범위와 평가 기준, 합격선 세 차원으로 분리된다. 검증 범위는 단일 워크로드 단일 클러스터에서 시작해 멀티 클러스터 멀티 워크로드로 확장한다. 평가 기준은 본 백서가 정량 4지표로 정리한 항목(복구 시점, 복구 시간, 자원 절감, 라이선스 절감)과 정성 네 측면(자동화, 신뢰 부트, 정책 정합성, 인력 모델)을 사용한다. 합격선은 가상화 기준 모델 대비 복구 시간 한 자릿수 분, 자원 절감 30% 이상, 정책 정합성 점검 통과 등으로 매트릭스에 명시한다. 본 백서가 권장하는 검증 후보 선정 절차는 워크로드 분류 결정에서 도출된 신규 시스템 후보 또는 분석 워크로드 후보 중 하나를 선택하는 것이다. 검증 종료 후 평가 기준에 미달한 항목은 다음 후보 선정의 입력으로 환류한다.

다섯째 영역은 인력 재편 — 플랫폼 엔지니어링 조직 모델을 어떻게 구성할 것인가다. 결정 옵션은 세 가지로 분리된다. (a) 플랫폼 엔지니어링 전담 팀 신설, (b) 기존 인프라 운영팀 전환, (c) SRE(Site Reliability Engineering, 사이트 신뢰성 엔지니어링) 통합 모델. 전담 팀 신설은 인력 충원 부담이 크지만 플랫폼 엔지니어링 전문성의 형성 속도가 가장 빠르다. 기존 인프라 운영팀 전환은 인력 충원 부담이 가장 낮지만 기존 운영 모델의 관성 때문에 GitOps와 선언적 운영 문화의 정착 속도가 느릴 수 있다. SRE 통합 모델은 가용성·관측 가능성(observability)·자동화를 한 팀으로 통합 운영하는 모델이며, 본 백서 8장의 해외 사례(Adidas, Lufthansa Systems, ING Bank)에서 공통으로 관찰된 패턴과 가장 가깝다 [S1] [S4]. 본 백서가 권장하는 모델은 기관 규모와 기존 인프라 운영팀의 학습 수용도에 따라 (b) 또는 (c)로 갈린다. 중대 규모 기관에서는 (c) SRE 통합 모델이 정량 정성 우위 종합과 정합하고, 소규모 기관에서는 (b) 기존 팀 전환이 현실적 진입 경로다.

5개 영역의 결정 옵션과 권장을 한 매트릭스로 정리하면 다음과 같다.

영역	결정 옵션	권장	권장 근거 (백서 장)
거버넌스	CIO 단독 / DR 거버넌스 보드 / 정보보호 책임자 공동	DR 거버넌스 보드 공동	1·9장
워크로드 분류	트랜잭션 / 세션 / 분석 / 신규 시스템	신규 시스템 우선	4·5장
데이터 복제 모델	4 카테고리 매트릭스	워크로드별 분리 결정	4장
다음 검증 후보	단일 → 멀티 클러스터 확장	신규 시스템 후보 / 분석 워크로드 후보	5·6장
인력 재편	전담 팀 / 기존 팀 전환 / SRE 통합	기관 규모별 (b) 또는 (c)	1·8장

본 매트릭스는 의사결정권자가 다음 분기 회의 안건으로 그대로 옮길 수 있는 구조로 정리되어 있다. 자사 거버넌스 환경에 따라 권장 항목을 변경할 수 있으나, 변경의 근거는 본 백서가 정량 정성 우위로 입증한 평가 축과 정합해야 한다 [S12].

10.2.2 도입 권장 전체 그림과 결론 시각화

5개 영역 매트릭스를 한 도식에 통합하면 결정의 상호 종속성과 권장 경로가 한 장으로 압축된다. 의사결정권자가 임원·이사회 보고에 단 한 페이지만 들고 들어가야 하는 경우에 사용할 수 있는 결론 자료다.

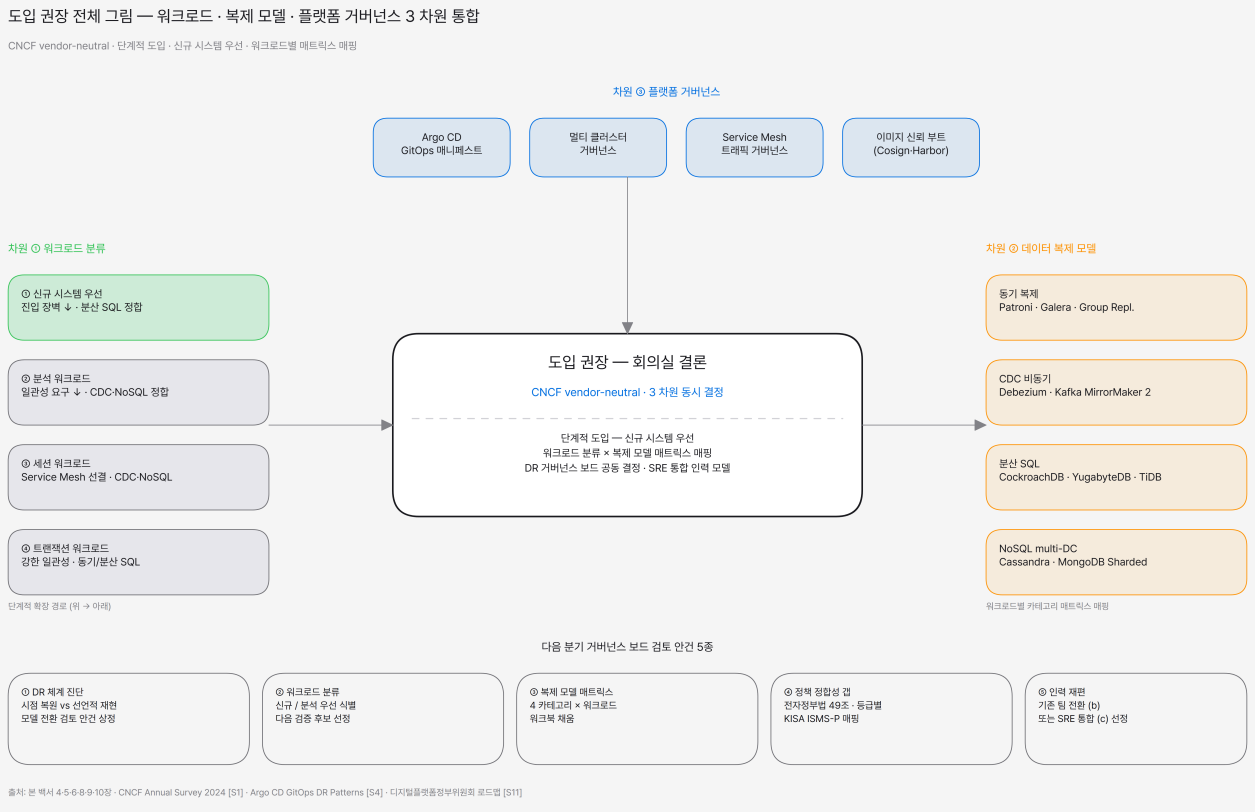
5개 영역 × 결정 옵션 × 권장 의사결정 매트릭스

5개 영역 각각의 결정 옵션 A/B/C 와 권장(CNCF vendor-neutral). 권장 열은 다음 분기 거버넌스 보드 회의 안건의 입력으로 환류된다.

영역	결정 옵션 A	결정 옵션 B	결정 옵션 C	권장 (CNCF vendor-neutral)
거버넌스	CIO 단독 결정	DR 거버넌스 보드	정보보호 책임자 공동	DR 거버넌스 보드 + 정보보호 책임자 참여
워크로드 분류	트랜잭션 우선	세션 우선	신규 시스템 우선	신규 시스템 우선 (전환 비용 최소)
데이터 복제 모델	4 카테고리 단일 채택	워크로드별 매트릭스	분산 SQL 일괄	워크로드별 매트릭스
다음 검증 후보	전체 시스템	단일 비핵심 시스템	다중 신규 시스템	단일 비핵심 시스템 → 핵심 시스템 단계 확장
인력 재편	플랫폼 엔지니어링 전담 팀 신설	기존 인프라 운영팀 전환	SRE 통합	단계적 SRE / 플랫폼 엔지니어링 통합

권장 열(우측 진한 청색)은 본 백서 1·4·5·8·9·10장의 분석 결론을 종합한 CNCF vendor-neutral 권장이며, 다음 분기 거버넌스 보드 회의 안건의 입력으로 환류된다.

위 도식은 5개 영역 × 결정 옵션 × 권장 매트릭스를 시각화한 결과다. 각 영역은 독립된 박스로 표현되지만 박스 사이의 화살표는 결정의 상호 종속성을 나타낸다. 거버넌스 박스에서 출발하는 화살표는 워크로드 분류와 인력 재편 박스로 향하고, 워크로드 분류 박스에서 출발하는 화살표는 데이터 복제 모델과 다음 검증 후보 박스로 향한다. 데이터 복제 모델은 본 백서 4장의 4 카테고리 매트릭스를 그대로 인용하여 별도 세부 도식 없이 매트릭스 자체가 의사결정 도구가 된다. 다음 검증 후보 박스는 평가 기준과 합격선을 함께 표시하여 검증 종료 후 회의 안건의 입력으로 환류되는 구조를 보인다.



위 도식은 도입 권장의 전체 그림을 워크로드 분류 차원, 데이터 복제 모델 차원, 플랫폼 거버넌스 차원의 세 축으로 통합한다. 워크로드 분류 차원은 신규 시스템 우선 → 분석 → 세션 → 트랜잭션의 단계적 확장 경로를 보이고, 데이터 복제 모델 차원은 4 카테고리가 워크로드 유형별로 매핑되는 매트릭스를 함께 표시한다. 플랫폼 거버넌스 차원은 DR 거버넌스 보드 공동 결정 모델, SRE 통합 또는 기존 팀 전환의 인력 모델, GitOps와 정책 엔진 통합의 운영 모델을 한 묶음으로 정리한다. 세 차원이 한 도식에 통합되면 본 백서의 결론이 단일 시각 자료로 압축된다.

본 백서가 다음 분기 검토 안건으로 제시할 항목은 5종으로 정리된다. (1) 자사 DR 체계가 시점 복원 모델인지 선언적 재현 모델인지 진단하고 모델 전환 검토를 거버넌스 보드 안건으로 올리는 것, (2) 자사 워크로드 약 5종 중 신규 시스템 또는 분석 워크로드를 다음 검증 후보로 식별하는 것, (3) 데이터 복제 모델 4 카테고리 매트릭스에 자사 워크로드 약어를 채워 워크북 형태로 운용하는 것, (4) 「전자정부법」 제49조와 행정안전부 재해복구 지침의 등급별 요구사항을 자사 시스템 분류와 매핑하여 정합성 갭을 식별하는 것 [S11], (5) 플랫폼 엔지니어링 조직 모델 (b) 기존 팀 전환 또는 (c) SRE 통합 중 자사 규모와 학습 수용도에 정합하는 모델을 선정하는 것이다. 5종 안건은 일자나 주차 단위 행동 계획이 아니라 거버넌스 보드 회의 안건의 형태로 제시되며, 각 안건의 결정 시점은 각 기관의 회의 주기에 따라 결정된다.

다음 분기 거버넌스 보드 회의에서 본 백서의 5개 영역 매트릭스를 한 페이지로 인쇄하여 안건으로 올리는 것 — 그것이 본 백서가 의사결정권자에게 제시하는 단일 행동이다. 매트릭스의 5개 영역 중 자사가 이미 결정한 항목과 미결정 항목을 색으로 구분하면, 거버넌스 보드 한 번의 회의에서 미결정 영역만 의제로 다룰 수 있어 회의 시간 자체가 줄어든다.

Appendix A. References

본 백서가 본문에서 인용한 출처를 식별자별로 정리합니다. 식별자는 본문 인라인의 `[S1]~[S12]` 표기 및 6장·8장의 footnote(`[^id]`) 표기에 대응합니다. 모든 URL 은 발행 기관의 1차 공식 경로를 기준으로 합니다.

큐레이션 12 출처 (본문 인라인 `[S1]~[S12]` 매핑)

[S1] CNCF, "Annual Survey 2024" — 운영 환경 쿠버네티스 채택률·컨테이너 도입 동인 통계. https://www.cncf.io/reports/cncf-annual-survey-2024/ (본문 인용 위치: 1·2·5·6·8·10장)

[S2] Kubernetes 공식 문서 — "Multi-cluster Topology", "StatefulSet", "Service" — 멀티 클러스터 토폴로지, 상태 비저장·상태 저장 워크로드 분리, Service 가상 IP 추상화. https://kubernetes.io/docs/ (본문 인용 위치: 2·3장)

[S3] Istio 공식 문서, "Multicluster Installation" — Multi-Primary / Primary-Remote / External Control Plane 3개 설치 모델, VirtualService 가중치 기반 트래픽 분기. https://istio.io/latest/docs/setup/install/multicluster/ (본문 인용 위치: 2장)

[S4] Argo CD 공식 문서, "ApplicationSet & GitOps DR Patterns" — 멀티 클러스터 매니페스트 동기, 페일오버 시나리오 표준 패턴, RBAC·Git 브랜치 보호. https://argo-cd.readthedocs.io/ (본문 인용 위치: 1·2·6·8·9·10장)

[S5] Velero 프로젝트, "Disaster Recovery" — 영속 데이터와 매니페스트 분리 복제·복원, PersistentVolume 스냅샷 정책. https://velero.io/docs/ (본문 인용 위치: 1·5·6장)

[S6] Harbor 공식 문서, "Replicating Resources" — 이벤트 기반·스케줄 기반 복제 정책, 단방향·양방향 ·Pull-through Cache 패턴, proxy cache project. https://goharbor.io/docs/ (본문 인용 위치: 3·7장)

[S7] OCI, "Distribution Specification" + Sigstore Cosign 사양 — 매니페스트·블롭 전송 표준, 콘텐츠 주소 지정(SHA-256 다이제스트), Cosign 서명·referrers·Fulcio·Rekor 투명성 로그. https://github.com/opencontainers/distribution-spec , https://docs.sigstore.dev/cosign/overview/ (본문 인용 위치: 3·7장)

[S8] 데이터베이스 공식 문서 묶음 — PostgreSQL Patroni, MySQL Group Replication, MariaDB Galera Cluster, Debezium + Kafka MirrorMaker 2, CockroachDB, YugabyteDB, PingCAP TiDB, MongoDB Sharded Cluster, Apache Cassandra Multi-DC. (본문 인용 위치: 4장)

- Patroni: https://patroni.readthedocs.io/
- MySQL Group Replication: https://dev.mysql.com/doc/refman/8.0/en/group-replication.html
- MariaDB Galera Cluster: https://mariadb.com/kb/en/galera-cluster/
- Debezium: https://debezium.io/documentation/
- CockroachDB Multi-Region: https://www.cockroachlabs.com/docs/
- YugabyteDB Multi-region: https://docs.yugabyte.com/
- PingCAP TiDB Multi-Region: https://docs.pingcap.com/
- MongoDB Multi-Region: https://www.mongodb.com/docs/
- Apache Cassandra Multi-DC: https://cassandra.apache.org/doc/

[S9] 「전자정부법」 제49조 (정보시스템의 안정성·신뢰성 제고) 및 시행령 — 국가기관·지방자치단체·공공기관의 재해복구체계 사전 구축 의무. 법제처 국가법령정보센터, https://www.law.go.kr/ (본문 인용 위치: 9·10장)

[S10] 행정안전부, 「공공기관 정보시스템 재해복구 체계 구축 지침」 + KISA, "ISMS-P 인증 기준 — 재해복구 항목" — 정보시스템 4개 등급별 RTO·RPO 정량 요구, 정기 훈련 의무, 사이버 재해 대응 통제, 감사 추적 보존. https://www.mois.go.kr/ , https://isms.kisa.or.kr/ (본문 인용 위치: 3·5·6·9·10장)

[S11] 디지털플랫폼정부위원회, "공공부문 클라우드 네이티브 전환 로드맵" + 한국지능정보사회진흥원(NIA), "공공부문 클라우드 네이티브 적용 가이드" — 전환 일정·평가 지표, 6개 영역 기술 요건, RFP 평가 항목 매핑. https://dpg.go.kr/ , https://www.nia.or.kr/ (본문 인용 위치: 9·10장)

[S12] Gartner, "Magic Quadrant for Disaster Recovery as a Service 2024" + IBM/Ponemon Institute, "Cost of a Data Breach Report 2024" — DR 시장 표준 RTO·RPO 추세, 사고 평균 비용, 컨테이너 기반 멀티 사이트 DR 정량 효과. https://www.gartner.com/ , https://www.ibm.com/reports/data-breach (본문 인용 위치: 5·6·8·10장)

추가 인용 출처 (8장 사례별 footnote)

8장 해외 사례 5건은 발표 자료별 footnote 형식으로 인용됩니다. 각 발표 자료는 공개 컨퍼런스·기술 블로그·공식 자료의 1차 출처를 따릅니다.

[6:3] Adidas, KubeCon EU 발표 자료, "OpenShift Multi-Region at Adidas" — 글로벌 전자상거래 멀티 리전 운영, 블랙 프라이데이 트래픽 흡수, GitOps 기반 매니페스트 동기. KubeCon + CloudNativeCon Europe, https://www.cncf.io/kubecon-cloudnativecon-events/ (본문 인용 위치: 8장)

[7:3] Lufthansa Systems, Red Hat Summit 발표 자료 — 항공 운항 시스템의 컨테이너 전환, 가상화 시점 백업 모델 한계, 인력 재교육 12개월 사례. https://www.redhat.com/en/summit (본문 인용 위치: 8장)

[8:5] ING Bank, "Multi-cluster Kubernetes Platform" 공개 발표 — 금융 감독 요구 변경 통제 GitOps 매핑, 멀티 클러스터 거버넌스, MTTR 한 자릿수배 단축. (본문 인용 위치: 8장)

[9:3] JPMorgan Chase, 공개 기술 블로그 — Kubernetes 멀티 사이트 운영 — 글로벌 24시간 거래 분 단위 RTO·분 단위 미만 RPO 운영 목표, GitOps 인수인계 SSoT. (본문 인용 위치: 8장)

[10:3] SAP, "S/4HANA on Kubernetes" 공식 자료 — ERP 멀티 사이트 운영 모델, 롤링 업데이트 무중단 패치, ERP 도메인 인력 6~12개월 재교육 사례. https://help.sap.com/ (본문 인용 위치: 8장)

보조 인용 풀 (본문 직접 인용 외 — 추가 자료)

다음 항목은 본문에서 직접 인용되지는 않았으나, 본 백서의 기술 사양·운영 표준 검증 시 1차 참조 가능한 보조 자료입니다.

- Kubernetes SIG Multicluster, https://github.com/kubernetes/community/tree/master/sig-multicluster
- Submariner Project — 멀티 클러스터 Pod-to-Pod 네트워크 터널, https://submariner.io/

- Kyverno — Kubernetes 네이티브 정책 엔진, ClusterPolicy / verifyImages 규칙, https://kyverno.io/docs/
- Flux CD — GitOps 컨트롤러 대안, https://fluxcd.io/
- Chaos Mesh — 카오스 엔지니어링 도구, 페일오버 시나리오 자동 검증, https://chaos-mesh.org/
- CNCF TAG App Delivery, https://github.com/cncf/tag-app-delivery
- Red Hat, "State of Kubernetes Security Report 2024"
- CNCF FinOps Foundation, "Kubernetes Cost Optimization"
- Kubernetes 공식 문서, "Pod Scheduling and Bin Packing"

(클라우드 사업자 매니지드 재해복구 서비스 관련 출처는 본 백서 범위 — 온프레미스·하이브리드 자체 운영 — 와 무관하므로 본 References 풀에서도 일괄 제외합니다.)

Appendix B. Glossary

본 백서 본문에 처음 등장한 용어를 한 자리에 정리합니다. 용어 순서는 영문 알파벳, 한글, 약어를 통합하여 가독성 기준으로 배열했습니다.

용어	정의
Active-Active (액티브-액티브)	두 사이트가 동시에 같은 서비스를 제공하면서 평시 자원도 함께 활용하는 운영 모델. 한쪽 사이트 장애 시 다른 사이트가 트래픽을 흡수한다. 본 백서 2장 참조.
Active-Standby (액티브-스탠바이)	한쪽 사이트만 트래픽을 처리하고 다른 사이트는 대기 상태로 유지하다가 재해 시점에 전환하는 운영 모델. 평시 자원 활용률이 낮다.
Admission Controller (입수 시점 제어)	쿠버네티스 클러스터에 매니페스트가 적용되기 직전 정책 평가를 수행하는 계층. Kyverno·OPA Gatekeeper 가 대표 구현.
ApplicationSet	Argo CD 의 멀티 클러스터 배포 추상. 단일 정의로 다수 클러스터에 동일 매니페스트를 자동 배포·갱신한다.
Argo CD	Git 저장소를 SSoT 로 두고 클러스터 상태를 지속 동기하는 GitOps 컨트롤러. CNCF 졸업 프로젝트.
Bin Packing (자원 통합 배치)	쿠버네티스 스케줄러가 Pod 자원 요청과 노드 가용량을 매칭해 가능한 한 적은 노드에 Pod 를 모아 배치하는 동작. 평시 자원 활용률 향상의 핵심 메커니즘.
CapEx (Capital Expenditure, 자본 지출)	자산 취득에 투입되는 회계상 지출 항목. DR 사이트 하드웨어 구입 비용이 이에 해당.
CDC (Change Data Capture, 변경 데이터 캡처)	데이터베이스 변경 로그(WAL·binlog·oplog) 를 외부 컴포넌트가 구독해 다른 사이트로 전파하는 비동기 복제 패

용어	정의
	턴. Debezium + Kafka MirrorMaker 2 가 대표 구현.
CIO (Chief Information Officer, 최고정보책임자)	기관·기업의 정보화 총괄 책임자. DR 거버넌스 의사결정 구조의 1차 결정권자 후보.
CISO (Chief Information Security Officer, 정보보호 책임자)	정보보호 총괄 책임자. ISMS-P 인증 갱신·통제 자동화 의사결정 영역의 책임자.
CNCF (Cloud Native Computing Foundation)	쿠버네티스·Argo CD·Istio·Velero 등 클라우드 네이티브 프로젝트를 호스팅하는 비영리 재단. Linux Foundation 산하.
CRDT (Conflict-free Replicated Data Type, 충돌 없는 복제 자료형)	자료형 자체가 수학적으로 충돌 없는 병합 연산을 보장하는 분산 자료 구조. 카운터(GCounter·PNCounter), 집합(GSet·ORSet), 맵(LWW-Map) 등이 대표 자료형.
CSI (Container Storage Interface)	쿠버네티스에서 외부 스토리지 시스템을 표준 방식으로 연결하는 인터페이스 사양.
Debezium	데이터베이스 변경 로그를 Kafka 토픽으로 발행하는 오픈소스 CDC 소스 커넥터.
DMZ (Demilitarized Zone, 비무장 지대)	외부망과 내부망 사이에 위치하는 중간 네트워크 영역. 망분리 환경의 게이트웨이 레지스트리가 배치되는 위치.
DNS (Domain Name System)	도메인 이름과 IP 주소를 매핑하는 표준 사양. 가중치(weighted DNS)와 GSLB 가 트래픽 분기 1계층으로 사용된다.
DPG (Digital Platform Government, 디지털플랫폼정부)	디지털플랫폼정부위원회가 추진하는 공공부문 디지털 전환 정책. 클라우드 네이티브 전환 로드맵의 발행 주체.
DR (Disaster Recovery, 재해복구)	자연재해·사이버 재해·시스템 장애로 인한 서비스 중단 상황에서 정상 운영을 복구하는 체계.
Drift (드리프트)	운영 사이트와 DR 사이트의 패치·구성·매니페스트가 시간 경과에 따라 미세하게 갈라지는 현상. 가상화 DR 의 고질적 운영 위험.
ERP (Enterprise Resource Planning)	기업 핵심 업무 시스템. SAP S/4HANA 가 대표 제품.
FinOps (Financial Operations, 클라우드 비용 운영)	클라우드·컨테이너 환경의 비용을 운영 관점에서 추적·최적화하는 실천 방식.
Fulcio	Sigstore 프로젝트의 인증 기관. OIDC 신원과 결합한 keyless 서명 흐름을 제공.
Galera Cluster	MariaDB 의 동기 복제 클러스터. wsrep(write-set replication) 인터페이스 기반 인증(certification) 단계에서 충돌 사전 검출.

용어	정의
GitOps	Git 저장소를 단일 진실 원본으로 두고 매니페스트로 운영 상태를 선언·동기하는 운영 방식. Argo CD·Flux CD 가 대표 구현.
GSLB (Global Server Load Balancing)	지리적으로 분산된 사이트의 트래픽을 DNS 응답으로 분배하는 부하분산 방식.
Harbor	CNCF 졸업 컨테이너 레지스트리 프로젝트. Geo-Replication 정책, proxy cache project, RBAC 지원.
Helm	쿠버네티스 매니페스트 템플릿 패키징 도구. CNCF 졸업 프로젝트.
HPA (Horizontal Pod Autoscaler, 수평 자동 확장)	워크로드 부하에 따라 Pod 복제본 수를 자동 조정하는 쿠버네티스 컴포넌트.
ISMS-P (Information Security Management System - Personal information protection, 정보보호 및 개인정보보호 관리체계)	KISA 가 운영하는 국내 대표 정보보호·개인정보보호 통합 인증. 재해복구 통제 항목 포함.
Istio	CNCF 졸업 Service Mesh 프로젝트. Multi-Primary / Primary-Remote / External Control Plane 멀티 클러스터 토폴로지 지원.
Kafka MirrorMaker 2	Apache Kafka 토픽을 다른 Kafka 클러스터로 복제하는 표준 도구. CDC 비동기 카테고리의 사이트 간 전파 채널.
KISA (Korea Internet & Security Agency, 한국인터넷진흥원)	국내 정보보호·개인정보보호 인증 운영 기관. ISMS-P 인증 기준 발행.
Kyverno	쿠버네티스 네이티브 정책 엔진. ClusterPolicy 리소스의 verifyImages 규칙으로 서명 검증·SBOM 검증·라벨 정책 강제.
L2 over L3	L3 네트워크 위에 L2 도메인을 터널링으로 확장하는 기법. 가상화 Active-Active 의 광역 L2 운영 부담의 원인.
LB (Load Balancer, 부하분산기)	트래픽을 다수 백엔드에 분산하는 네트워크 장비·소프트웨어. 트래픽 분기 2계층.
LWW (Last-Write-Wins)	충돌 검출 시 더 큰 타임스탬프(또는 시퀀스) 를 가진 변경을 채택하는 단순 충돌 해결 방식. Cassandra 의 셀 단위 LWW 가 대표 구현.
Manifest (매니페스트)	쿠버네티스 리소스의 원하는 상태를 선언하는 YAML 정의. 클라우드 네이티브 DR 의 신뢰 기준 중 하나.
mTLS (mutual TLS, 상호 TLS 인증)	클라이언트와 서버가 서로 인증서로 인증하는 보안 통신. Service Mesh 의 워크로드 간 통신 표준.

용어	정의
MTTR (Mean Time To Recovery, 평균 복구 시간)	사고 발생 시점부터 복구 완료까지의 평균 시간. 운영 모델 평가 지표.
Multi-Primary	Istio 멀티 클러스터 설치 모델 중 하나. 두 클러스터가 각자 독립 컨트롤 플레인을 가지면서 워크로드 디스커버리·라우팅·인증을 통합.
NAT (Network Address Translation)	사설 IP 주소를 공인 IP 주소로 변환하는 네트워크 기법. 가상화 DR 의 네트워크 재설정 단계에서 사용.
NIA (National Information Society Agency, 한국지능정보사회진흥원)	공공부문 정보화 정책·지원 기관. "공공부문 클라우드 네이티브 적용 가이드" 발행.
NoSQL multi-DC	가용성 우선(AP 모델) 데이터베이스의 멀티 데이터센터 배포 카테고리. MongoDB Sharded Cluster, Apache Cassandra, ScyllaDB 가 대표 구현.
OCI (Open Container Initiative)	컨테이너 이미지·런타임·배포 표준 사양을 정의하는 비영리 단체. Distribution Specification, Image Specification, Runtime Specification 관리.
OIDC (OpenID Connect)	OAuth 2.0 기반 신원 인증 표준. Sigstore keyless 서명 흐름의 신원 증명 채널.
OLTP (Online Transaction Processing)	온라인 트랜잭션 처리 워크로드. 금융·재고·결제 시스템이 대표.
OOMKill (Out Of Memory Kill)	메모리 자원 부족으로 Pod 가 강제 종료되는 상황. VPA 의 보정 대상.
OpEx (Operational Expenditure, 운영 지출)	운영 단계에 발생하는 회계상 지출 항목. DR 사이트 전력·상면·라이선스 비용이 이에 해당.
OPA Gatekeeper	Rego 언어 기반의 쿠버네티스 정책 엔진. Kyverno 와 함께 정책 강제 도구의 양대 축.
OpenSSF (Open Source Security Foundation)	Linux Foundation 산하 오픈소스 보안 재단. Sigstore 프로젝트 후원.
OpenTelemetry	CNCF 졸업 관측 가능성(observability) 표준 프로젝트. 트레이스·메트릭·로그 수집 표준.
OS (Operating System, 운영체제)	하드웨어 자원을 관리하고 응용 프로그램 실행 환경을 제공하는 소프트웨어. 본 백서 5장의 "OS-less" 표현은 컨테이너가 호스트 OS 커널을 공유하는 구조를 가리킨다.
Patroni	PostgreSQL 의 고가용성·자동 페일오버 도구. etcd·Consul 같은 외부 분산 합의 저장소로 리더 선출 관리.

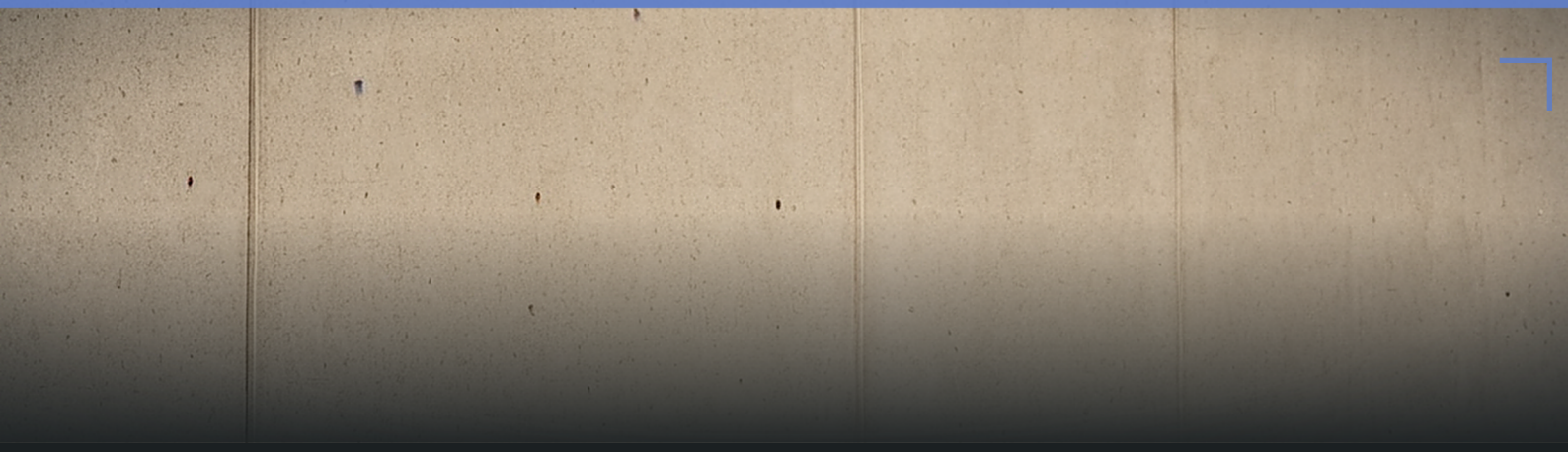
용어	정의
Pod (파드)	쿠버네티스의 최소 배포 단위. 한 개 이상의 컨테이너를 묶어 같은 네트워크·스토리지 컨텍스트를 공유.
Progressive Delivery (진보적 배포)	카나리·블루그린·점진 트래픽 전환 등으로 배포 위험을 단계적으로 분산하는 방식. Argo Rollouts 가 대표 구현.
Prometheus	CNCF 졸업 메트릭 수집·시계열 데이터베이스 프로젝트.
Quorum (쿼럼)	분산 시스템에서 합의에 필요한 최소 정족수. 가상화 공유 스토리지 락과 동기 복제 카테고리의 핵심 메커니즘.
RBAC (Role-Based Access Control, 역할 기반 접근 통제)	역할 단위로 권한을 부여하는 접근 통제 모델. Argo CD·쿠버네티스 권한 관리의 표준.
Rekor	Sigstore 프로젝트의 투명성 로그(transparency log). 서명 이력을 변경 불가능하게 기록.
RFP (Request for Proposal, 제안요청서)	사업 발주 시 사업자에게 제안을 요청하는 문서. 본 백서는 OCI 표준 사양 준수 등 평가 항목 명시를 권장.
ROI (Return on Investment, 투자 수익률)	투자 대비 회수 성과 지표. DR 모델 전환의 의사결정 회의실 평가 축.
RPO (Recovery Point Objective, 복구 시점 목표)	재해 발생 시 허용 가능한 데이터 손실 시간. 백업 주기·복제 지연의 영향.
RTO (Recovery Time Objective, 복구 시간 목표)	재해 발생부터 서비스 복구까지 허용 가능한 시간. 행정안전부 지침이 등급별로 정량 규정.
SAN (Storage Area Network)	블록 스토리지 전용 네트워크. 가상화 환경의 공유 스토리지 구성에 사용.
SBOM (Software Bill of Materials, 소프트웨어 자재명세)	이미지·소프트웨어에 포함된 모든 라이브러리·런타임·의존성 명세. 공급망 보안 통제 도구.
Service Mesh (서비스 메시)	워크로드 간 통신·라우팅·인증·관측을 사이드카(sidecar) 프록시로 추상화하는 인프라 계층. Istio 가 대표 구현.
Sigstore Cosign	컨테이너 이미지에 암호 서명을 첨부·검증하는 오픈소스 도구. Sigstore 프로젝트 산하.
SLA (Service Level Agreement, 서비스 수준 합의)	서비스 제공자와 사용자 간의 정량 수준 합의서. RTO·RPO 가 핵심 항목.
SOP (Standard Operating Procedure, 표준 운영 절차)	운영 작업의 표준화 문서. 가상화 DR 의 단계별 절차서가 대표 사례.
SRE (Site Reliability Engineering, 사이트 신뢰성 엔지니어링)	가용성·관측 가능성·자동화를 한 팀으로 통합 운영하는 모델. 클라우드 네이티브 환경의 표준 인력 모델 중 하나.

용어	정의
SSoT (Source of Truth, 단일 진실 원본)	시스템 상태의 단일 권위 출처. GitOps 의 Git 저장소, 멀티 사이트 운영의 SSoT 레지스트리가 대표.
StatefulSet (스테이트풀셋)	안정적 네트워크 식별자와 영속 볼륨을 가지는 쿠버네티스 상태 저장 워크로드 리소스.
Stretch Cluster (스트레치 클러스터)	두 사이트의 공유 스토리지를 동기 미러링으로 묶는 가상화 고가용성 구성. 광역 회선 지연과 split-brain 위험.
Submariner	CNCF 멀티 클러스터 네트워크 프로젝트. 두 클러스터의 Pod CIDR 을 안전한 터널로 연결.
TiDB	PingCAP 의 분산 SQL 데이터베이스. MySQL 와이어 프로토콜 호환 + TiKV 분산 키-값 저장소 2계층 구조.
TTL (Time To Live)	DNS 응답·캐시의 유효 기간. 트래픽 분기 시 전환 지연의 1차 변수.
UBI (Universal Base Image)	Red Hat 의 컨테이너 친화적 경량 베이스 이미지.
VPA (Vertical Pod Autoscaler, 수직 자동 확장)	Pod 한 개에 할당된 CPU·메모리 자원을 실측 사용량에 맞춰 조정하는 쿠버네티스 컴포넌트.
VirtualService (버추얼서비스)	Istio 의 트래픽 라우팅 정의 리소스. 가중치 한 줄 변경으로 사이트 간 트래픽 분기 실현.
WAL (Write-Ahead Log)	PostgreSQL 의 변경 로그. CDC 비동기 카테고리의 변경 데이터 채굴 대상.
WAS (Web Application Server, 웹 애플리케이션 서버)	웹 애플리케이션 실행 미들웨어. 가상화 DR 의 미들웨어 재구성 단계 대상.
YugabyteDB	PostgreSQL 와이어 프로토콜 호환 SQL 계층 + 분산 키-값 저장소 2계층 분산 SQL 데이터베이스.
게스트 OS	가상 머신 안에서 동작하는 운영체제. 가상화 환경 3단 라이선스 모델의 둘째 단.
결과적 일관성 (Eventual Consistency)	복제 지연이 끝나면 모든 복제본이 동일 상태에 수렴하는 일관성 모델. CDC 비동기 카테고리의 보장 수준.
골든 이미지 (Golden Image)	가상화 환경의 표준 VM 이미지 원본. 패치 누적으로 드리프트가 발생하는 출발점.
광역 L2	두 사이트에 동일 L2 네트워크 도메인을 확장하는 구성. 가상화 Active-Active 의 IP 종속 해결 수단이자 부가 운영 부담.
동기 복제	한 사이트의 쓰기가 다른 사이트 복제본에 반영되어 합의에 도달한 뒤에야 성공을 반환하는 강한 일관성 카테고리.

용어	정의
	Patroni·MySQL Group Replication·Galera 가 대표 구현.
매니지드 DR	클라우드 사업자(CSP) 가 운영을 대신하는 DR 서비스. 본 백서 범위(자체 운영) 외.
망분리 (air-gapped)	외부망과 내부망을 물리·논리적으로 분리하는 보안 구성. 국내 공공·금융 환경의 표준.
선언적 배포 (Declarative Deployment)	원하는 상태(desired state) 를 매니페스트로 선언하면 컨트롤러 루프가 실제 상태(actual state) 를 일치시키는 배포 방식.
선언적 재현 모델	시점 백업 후 복원이 아니라, 선언된 매니페스트·이미지 다이제스트·복제된 데이터를 보조 사이트에서 재현하는 클라우드 네이티브 DR 의 본질 모델.
세션 어피니티 (Session Affinity)	같은 사용자 요청을 같은 인스턴스로 라우팅하는 정책. 가상화 미들웨어의 일반 가정.
스트레치 클러스터	Stretch Cluster 항목 참조.
시점 복원 모델	백업 시점의 상태를 그대로 다시 부팅하는 가상화 DR 의 본질 모델.
신뢰 부트 (Trusted Boot)	이미지 불변성과 Cosign 서명 검증으로 운영 시점의 신뢰 기준을 확정하는 컨테이너 환경의 보안 메커니즘. 사이버 재해 시 1차 방어선.
신뢰 모델	시스템이 무엇을 신뢰의 기준으로 두는지의 가정. 가상화는 "백업 매체", 컨테이너는 "이미지 다이제스트 + 서명".
워크로드 (Workload)	운영 환경에서 동작하는 응용 단위. 트랜잭션·세션·로그·분석 4가지 유형으로 분류 가능.
이미지 다이제스트 (Image Digest)	컨테이너 이미지의 SHA-256 해시 식별자. 같은 다이제스트의 이미지는 비트 단위로 동일.
이미지 불변성 (Immutable Image)	한 번 빌드된 컨테이너 이미지는 내용이 바뀌지 않는다는 성질. 클라우드 네이티브 DR 의 신뢰 기준.
입수 시점 검증 (Admission Control)	매니페스트가 클러스터에 적용되기 직전 정책 엔진이 검증을 수행하는 단계. Kyverno·OPA Gatekeeper 가 대표.
정책 엔진	매니페스트·이미지에 대한 정책 검증을 자동으로 강제하는 도구. Kyverno·OPA Gatekeeper.
카오스 엔지니어링 (Chaos Engineering)	운영 환경에 의도적 장애를 주입해 회복 능력을 검증하는 실천 방식. Chaos Mesh 가 대표 구현.

용어	정의
컨테이너 (Container)	호스트 OS 커널을 공유하면서 사용자 공간만 격리된 가벼운 실행 단위.
콘텐츠 주소 지정 (Content-Addressable Storage)	데이터를 해시 다이제스트로 식별하는 저장 방식. OCI Distribution Specification 의 표준 사양.
쿠버네티스 (Kubernetes)	CNCF 졸업 컨테이너 오케스트레이션 플랫폼. 선언적 배포·자동 복구·서비스 디스커버리 표준.
쿼럼	Quorum 항목 참조.
플랫폼 엔지니어링 (Platform Engineering)	개발자 자가 서비스 플랫폼을 구축·운영하는 조직 모델. GitOps·관측 가능성·정책 엔진을 통합 운영.
하이퍼바이저 (Hypervisor)	물리 하드웨어 위에 다수의 가상 머신을 실행시키는 가상화 소프트웨어. 가상화 환경 3단 라이선스 모델의 첫째 단.
핫스팟 (Hotspot)	분산 데이터베이스에서 특정 노드·범위에 부하가 집중되는 현상. 분산 SQL 운영 설계의 주요 회피 대상.
확정 시점 (Commit)	트랜잭션이 데이터베이스에 영구 반영되는 시점.

1. 행정안전부, "공공기관 정보시스템 재해복구 체계 구축 지침" + KISA, "ISMS-P 인증 기준 — 재해복구 항목". ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩ ↩
2. Gartner, "Magic Quadrant for Disaster Recovery as a Service 2024" + IBM/Ponemon Institute, "Cost of a Data Breach Report 2024". ↩ ↩ ↩ ↩ ↩ ↩
3. CNCF, "Annual Survey 2024", https://www.cncf.io/reports/cncf-annual-survey-2024/ ↩
4. Argo CD 공식 문서, "ApplicationSet & GitOps DR Patterns", https://argo-cd.readthedocs.io/ ↩ ↩
5. Velero 프로젝트, "Disaster Recovery", https://velero.io/docs/ ↩ ↩ ↩
6. Adidas, KubeCon EU 발표 자료, "OpenShift Multi-Region at Adidas". ↩ ↩ ↩ ↩
7. Lufthansa Systems, Red Hat Summit 발표 자료. ↩ ↩ ↩ ↩
8. ING Bank, "Multi-cluster Kubernetes Platform" 공개 발표. ↩ ↩ ↩ ↩ ↩ ↩
9. JPMorgan Chase, 공개 기술 블로그 — Kubernetes 멀티 사이트 운영. ↩ ↩ ↩ ↩
10. SAP, "S/4HANA on Kubernetes" 공식 자료. ↩ ↩ ↩ ↩



공공기관 클라우드 네이티브 기반 DR 구축 백서

CONTACT

WEB
cncf.co.kr
www.cncf.co.kr

EMAIL
info@cncf.co.kr

TEL
+82-2-1670-1010
+82216701010

YOUTUBE
@cncf
www.youtube.com/@cncf

LINKEDIN
linkedin.com/company...
www.linkedin.com/company/cloud-na...

FACEBOOK
facebook.com/cncf
www.facebook.com/cncf

SCAN