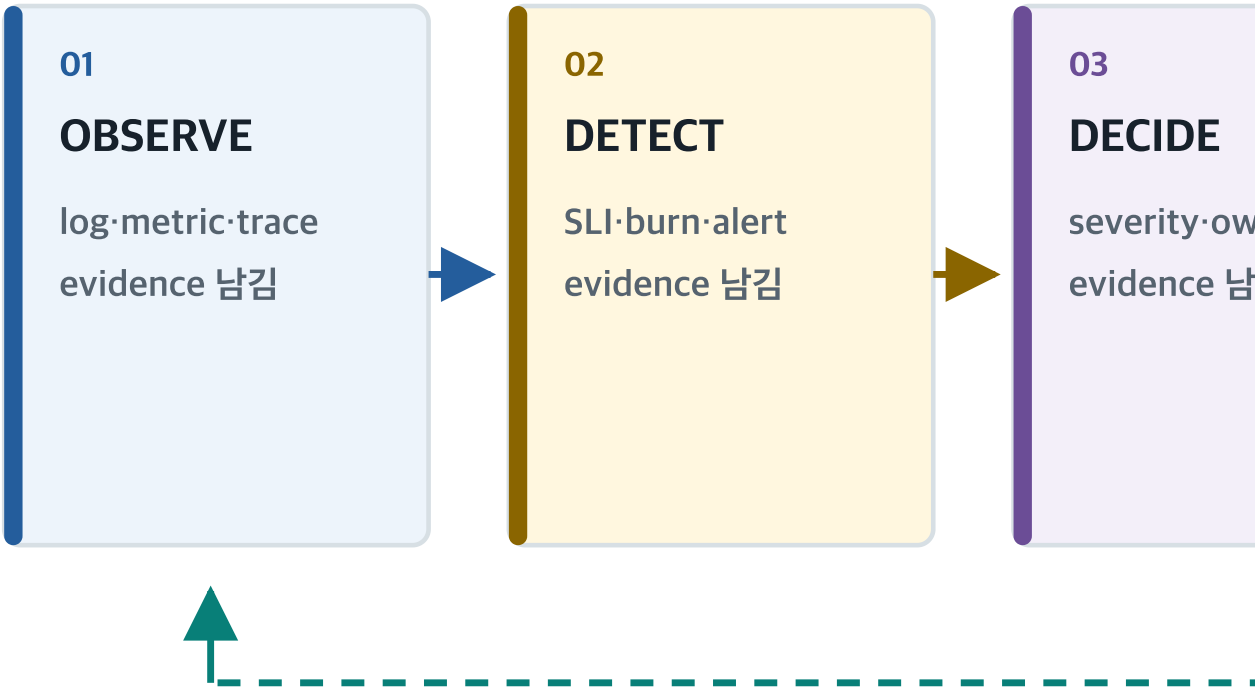
M11-03 · GIBALJA VISUAL MANUAL

로그·모니터링·백업·장애 대응 설계하기

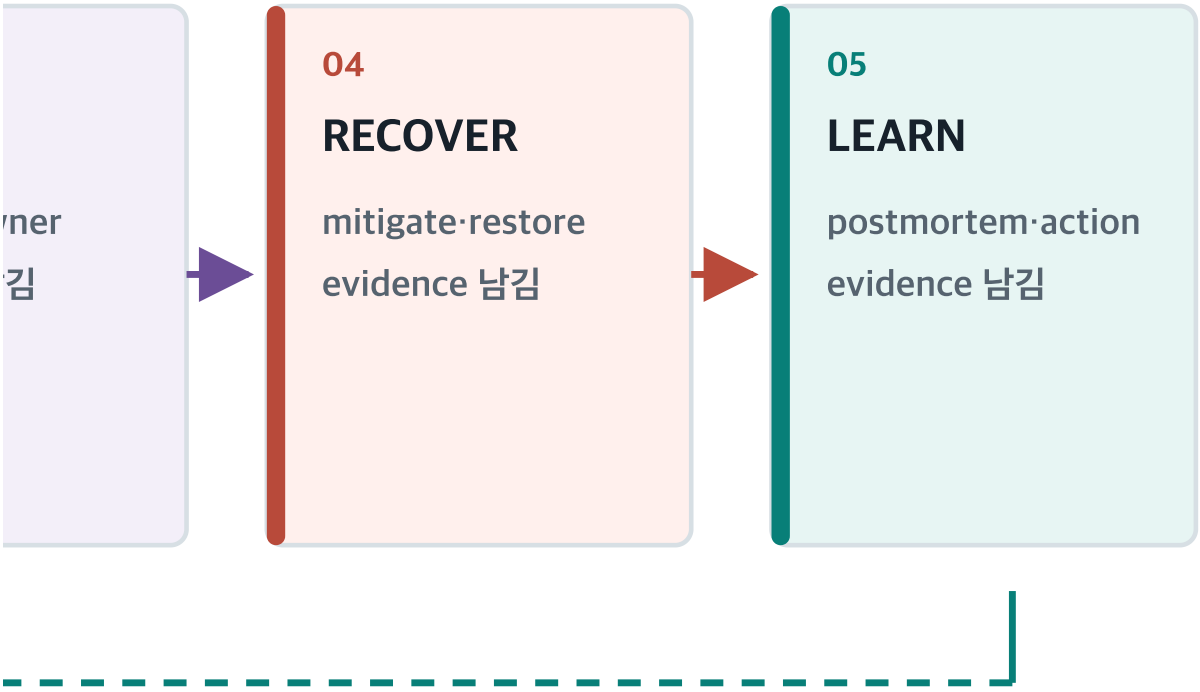
한 문장 목표: 실제 production telemetry·alert·notification·backup·restore·cloud·외부 network를 사용하지 않고, 관측에서 복구와 학습까지 24개 합성 시나리오로 연결해 `operations_ready` 후보를 판정합니다.

그림 1. 운영은 dashboard를 보는 데서 끝나지 않습니다. Observe·detect·decide·recover·learn이 owner와 evidence로 이어지고, 학습이 다시 계측과 통제를 바꿔야 닫힌 순환이 됩니다.

Dashboard를 보는 일에서 끝나지 않고 탐지·결정·복구·후속 조치가 다시 관



좌측 설계로 돌아옵니다.


wner
김
04
RECOVER
mitigate·restore
evidence 남김
05
LEARN
postmortem·action
evidence 남김

0. 한눈에 보는 두 번째 지도

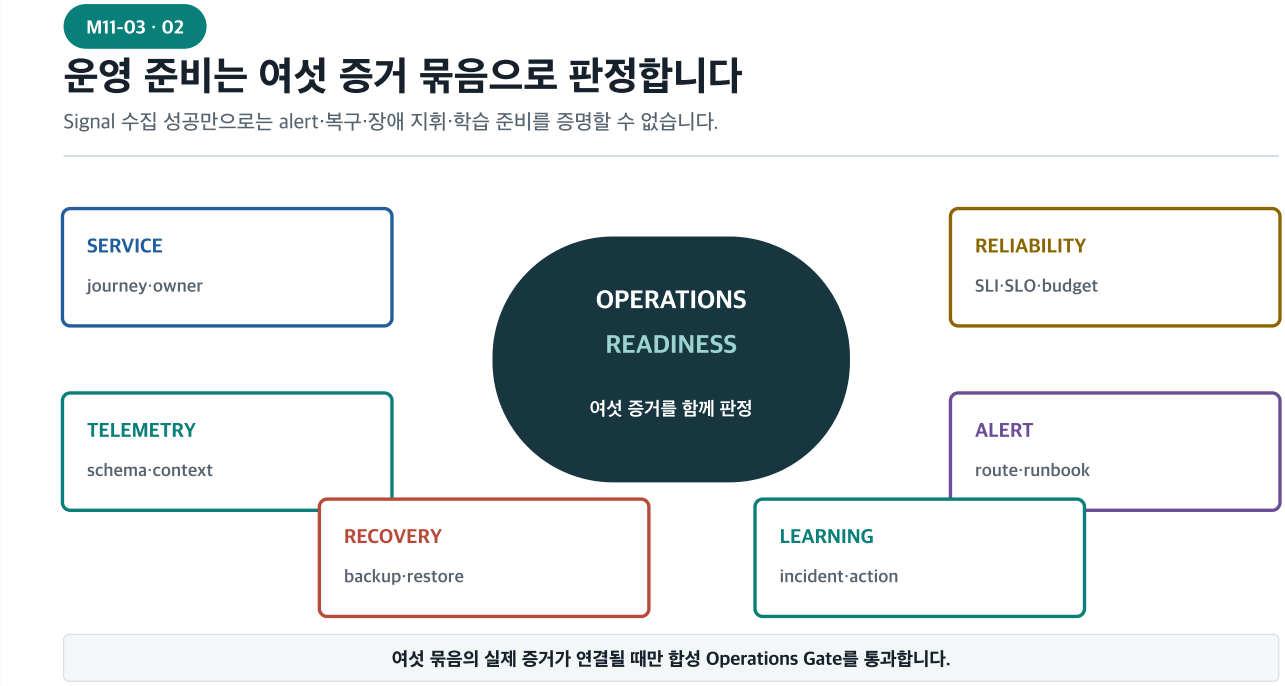
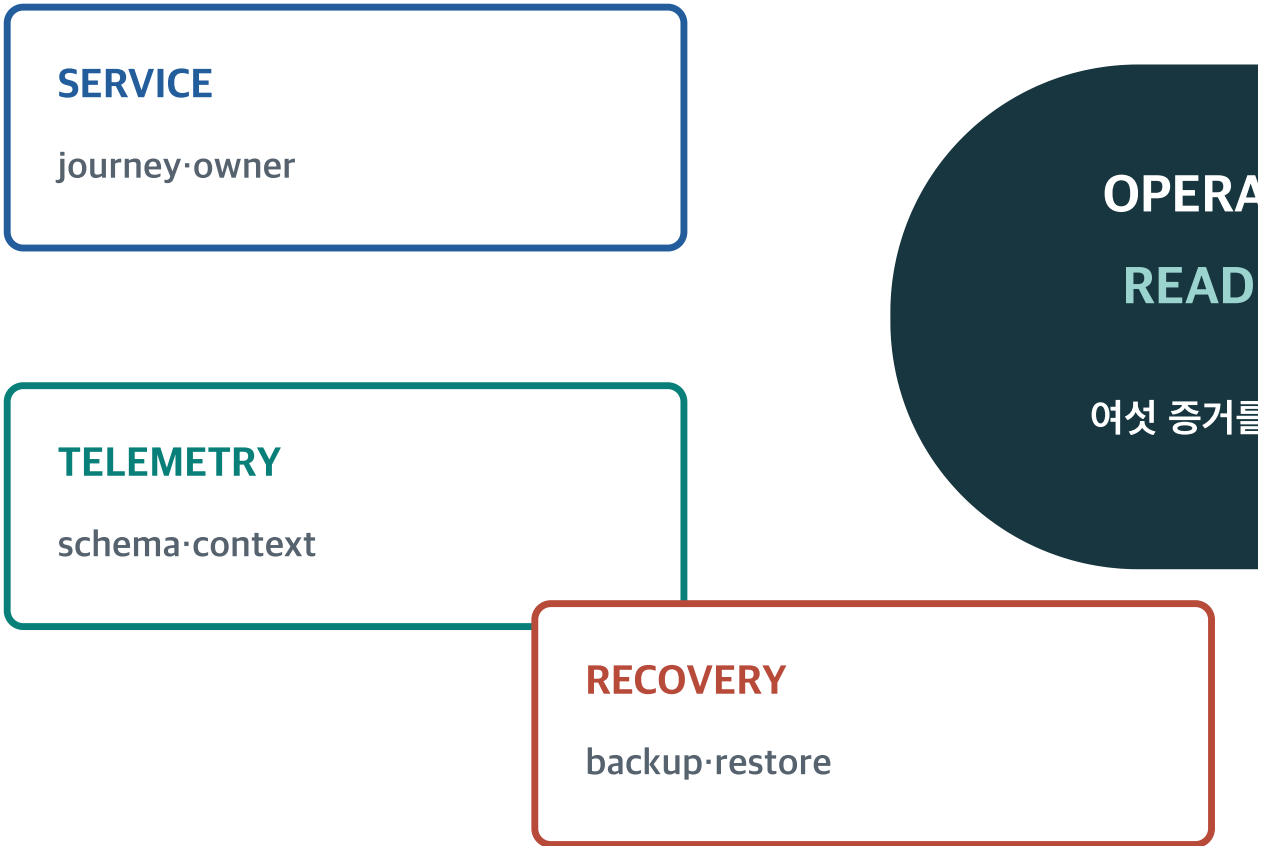
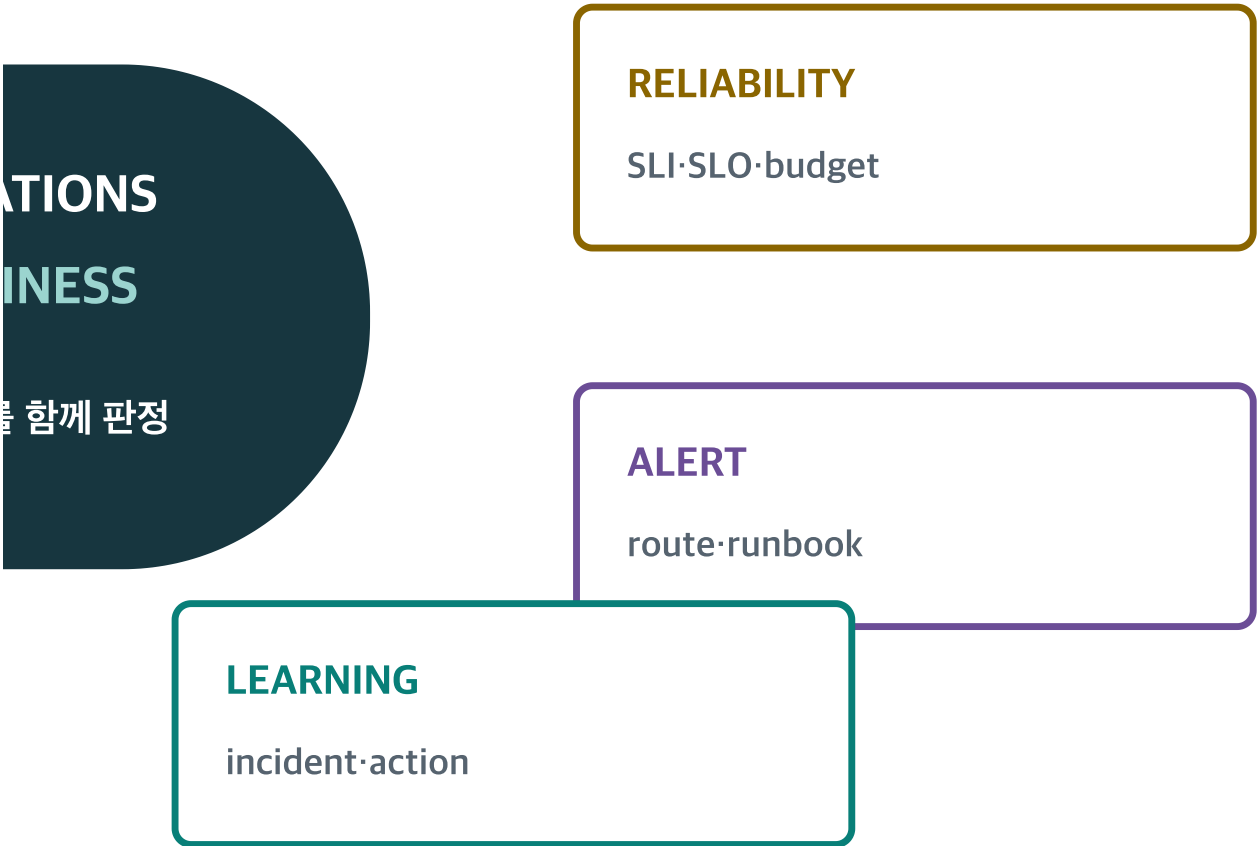


그림 2. Signal을 수집한다는 사실만으로는 운영 준비를 증명하지 못합니다. Service 범위·telemetry·reliability·alert·recovery·learning 여섯 묶음이 함께 연결돼야 합니다.

Signal 수집 성공만으로는 alert·복구·장애 지휘·학습 준비를 증명할 수 없습



늡니다.


TIONS
INESS
를 함께 판정
RELIABILITY
SLI·SLO·budget
ALERT
route·runbook
LEARNING
incident·action

난이도	그림 먼저	개념·판정	실습	셀프 테스트	최종 산출물
Level 2	35분	70분	90분	15분	운영 계획·24개 결과·Operations Gate

이 매뉴얼은 특정 observability·cloud·backup 제품의 console 사용법이 아닙니다. 제품 기능·가격·보존·불변성·region·license는 시점과 plan에 따라 바뀝니다. 먼저 provider-neutral한 질문과 evidence를 익힌 뒤 실제 작업 직전에 선택한 제품의 최신 공식 문서와 조직 절차를 확인합니다. 이 합성 실습은 실제 production 승인·보안 인증·재해 복구 인증·개인정보 영향평가를 대신하지 않습니다.

0.1 첫 두 장에서 기억할 열두 문장

```
Dashboard가 있다고 관측 가능한 것은 아니다.
Log·metric·trace는 같은 사건을 다른 각도에서 본다.
공통 context가 없으면 signal은 서로 연결되지 않는다.
Secret·token·개인정보·원문 payload는 애초에 telemetry로 보내지 않는다.
Monitoring pipeline도 별도로 monitoring해야 한다.
SLI는 측정, SLO는 목표, error budget은 행동 기준이다.
99.9%는 예시일 뿐 모든 서비스의 정답이 아니다.
Alert는 사용자 영향에서 runbook과 owner까지 이어져야 한다.
Backup job 성공은 복구 가능성의 증명이 아니다.
RPO는 잃을 수 있는 시간, RTO는 멈출 수 있는 시간이다.
Incident 대응은 지휘·실행·소통·기록을 역할로 나눈다.
Postmortem action은 owner·기한·재검 evidence가 있어야 끝난다.
```

1. 이 PDF를 공부하는 방법

1.1 1회차 · 그림 16장만 읽기 · 35분

그림 제목과 caption만 읽고 다음 순서를 소리 내어 설명합니다.

```
Observe·detect·decide·recover·learn
→ 여섯 evidence 묶음
→ log·metric·trace·release 상관관계
→ 구조화 event·민감정보 경계
→ SLI·SLO·error budget
→ dashboard·alert·monitoring-of-monitoring
→ backup protection·RPO·RTO·restore drill
→ incident command·postmortem
→ 24개 scenario·Operations Gate
```

각 그림에서 다음 문장을 완성합니다.

> 이 단계에서 사용자가 겪는 실패는 _____이고, 먼저 볼 signal은 _____이며, owner가 실행할 행동은 _____이고, 완료 evidence는 _____이다.

1.2 2회차 · 운영 준비 검수 스튜디오 · 55분

실습 생성기를 실행합니다.

```
./02_Labs/G11_Deployment_Operations/L11-03_create-operations-readiness-practice.sh
```

세 version의 차이를 직접 확인합니다.

Version	Pass	Decision
`dashboard-only-v1`	6/24	`blocked_operational_blindness`
`signals-without-recovery-v2`	16/24	`blocked_recovery_readiness`
`verified-operations-v3`	24/24	`operations_ready`

1.3 3회차 · 내 서비스 계획서 작성 · 120분

운영 모니터링·백업·장애 대응 계획서를 복제해 다음 순서로 채웁니다.

1. Service·critical journey·dependency·owner
2. Structured event·metric·trace·context
3. Telemetry 민감정보·retention·pipeline health
4. SLI·SLO·error budget policy
5. Golden signal dashboard·alert route·on-call
6. Backup inventory·schedule·retention·protection
7. RPO·RTO·dependency order·restore drill
8. Incident severity·role·timeline·status·mitigation
9. Postmortem·action·12 control·24 scenario·Operations Gate

1.4 읽다가 막힐 때

로그·모니터링·백업·장애 대응 용어집 300에서 지금 장의 20개 묶음만 읽습니다. 모든 용어를 먼저 외우지 않습니다.

2. “운영 중이다”를 다시 정의하기

2.1 다섯 문장은 서로 다르다

문장	실제 의미	아직 모르는 것
Process가 실행 중이다.	Runtime이 살아 있다.	사용자 journey·dependency
Health endpoint가 200이다.	한 내부 검사가 통과했다.	Public path·업무 결과
Dashboard가 초록이다.	선택한 query가 기준 안이다.	Signal 누락·pipeline 장애
Backup job이 성공했다.	Copy 작업이 종료됐다.	무결성·격리·복원·업무 검증
Incident가 종료됐다.	대응자가 상태를 닫았다.	Known-good·data integrity·후속 action

OPERATIONS READINESS 운영 준비는

사용자 영향을 관찰하고, 실행 가능한 경보로 owner를 부르고, 안전하게 복구하고, evidence로 검증한 뒤 system을 개선할 수 있는 상태

입니다.

2.2 운영 준비의 최소 계약

단계	질문	최소 evidence	실패 행동
Scope	무엇을 누가 언제 운영하는가	Service·journey·dependency·owner	Owner 확정 전 block
Observe	어떤 signal로 사용자 경험을 보는가	Schema·context·pipeline health	Blind spot 수정
Detect	언제 사람을 부르는가	SLI·SLO·burn·route test	Noisy·blind alert 수정
Decide	누가 우선순위를 정하는가	Severity·IC·timeline·decision	역할 배정
Recover	어느 known-good로 돌아가는가	RPO·RTO·restore·business validation	복구 재검
Learn	무엇을 system에 반영하는가	Postmortem·action·owner·due·verification	Gate block

2.3 여섯 가지 대표 실패

실패 유형	겉으로 보이는 증상	실제 위험
Telemetry 사각지대	Graph가 조용함	필요한 event가 없거나 pipeline이 멈춤
Signal 상관관계 단절	Log는 많은데 원인 불명	Request·trace·release·dependency 연결 없음
SLO·Alert 오류	Page가 너무 많거나 늦음	사용자 영향보다 symptom에 반응
Backup 공백	Job은 성공	중요 asset 누락·삭제·변조·key 부재
Recovery 실패	Copy는 존재	RPO·RTO·dependency order·업무 검증 없음
Incident 조정 실패	여러 사람이 바쁨	선언·역할·결정·소통·action owner 없음

2.4 M11-02에서 받는 Handoff

이 매뉴얼은 M11-02의 공개 endpoint를 다시 배포하지 않습니다. 다음 evidence를 입력으로 받습니다.

- Production environment·region·resource ID
- Exact release ID·artifact digest·configuration version
- Public hostname·health path
- DNS target·TLS certificate expiry
- Managed data·backup owner
- Rollback trigger·known-good release

M11-03은 이 대상을 지속적으로 관찰·복구·학습하는 계약을 만듭니다.

---

3. 운영은 닫힌 순환이다

3.1 Observe

Observe는 “모든 것을 수집”이 아닙니다. 다음 질문에 답할 최소 signal을 고릅니다.

- 사용자는 핵심 journey를 성공하는가.
- 어느 release·region·dependency에서 달라졌는가.
- 실패는 갑자기 시작됐는가, 서서히 쌓였는가.

- Monitoring pipeline 자체는 정상인가.
- 복구 뒤 public path와 업무 결과가 돌아왔는가.

3.2 Detect

Detect는 threshold를 많이 만드는 일이 아닙니다. 사용자 영향과 error budget 소진을 충분히 빨리 알아채는 일입니다.

```
user journey
→ good / total SLI
→ SLO target·window
→ error budget·burn rate
→ actionable alert
```

3.3 Decide

장애 중에는 정보가 불완전합니다. 그래서 “정답을 아는 사람”보다 역할과 의사결정 기록이 중요합니다.

결정	Owner	Evidence
Incident 선언·severity	Incident Commander	Impact·trigger·start UTC
Traffic 제한·rollback	Operations lead + approver	Release·health·stop condition
Status 공유	Communications lead	Confirmed facts·next update
Restore 선택	Recovery owner	Known-good point·RPO·RTO
종료	Incident Commander	Public path·data·minimum service

3.4 Recover

Recovery는 error graph가 내려오는 순간이 아닙니다.

```
traffic protected
+ known-good restored
+ public path verified
+ data integrity verified
+ controlled change
= recovery evidence
```

3.5 Learn

Postmortem의 목적은 멋진 문서가 아니라 다음 incident의 발생 가능성·영향·탐지·복구 시간을 줄이는 것입니다.

학습 완료 조건은 “회고 회의 완료”가 아니라 **action owner·due date·verification evidence·재검 결과**입니다.

4. Log·Metric·Trace를 한 사건으로 연결하기

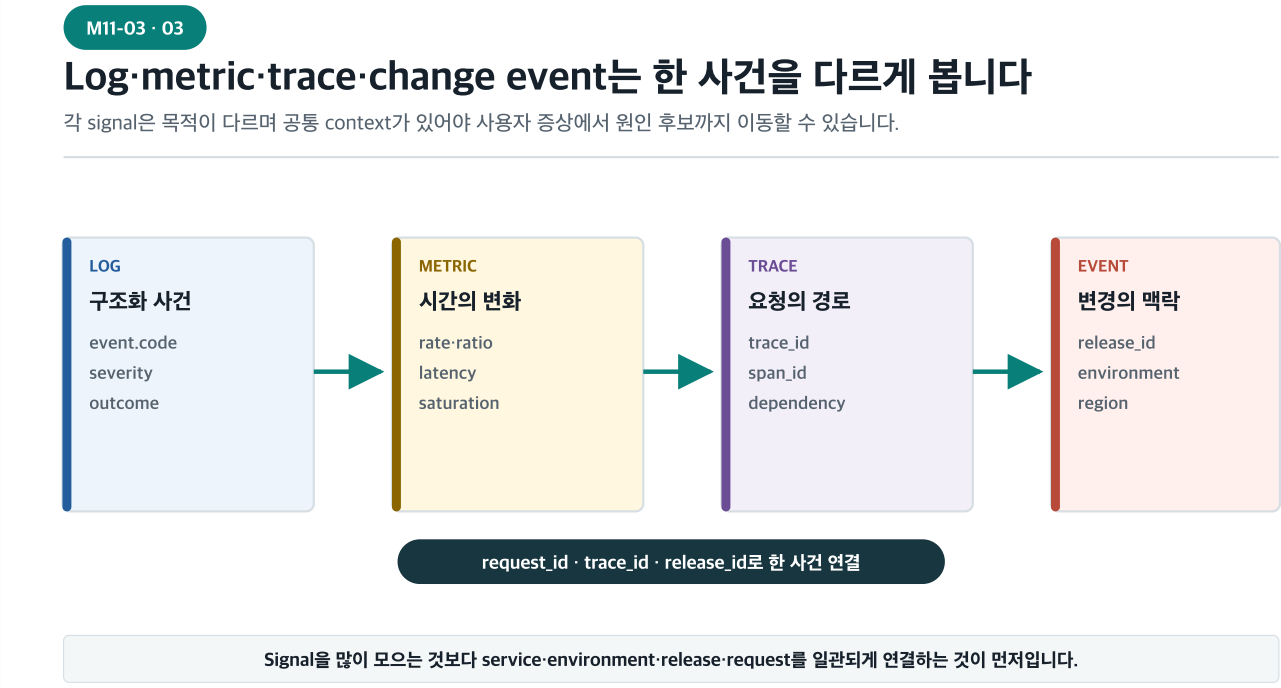
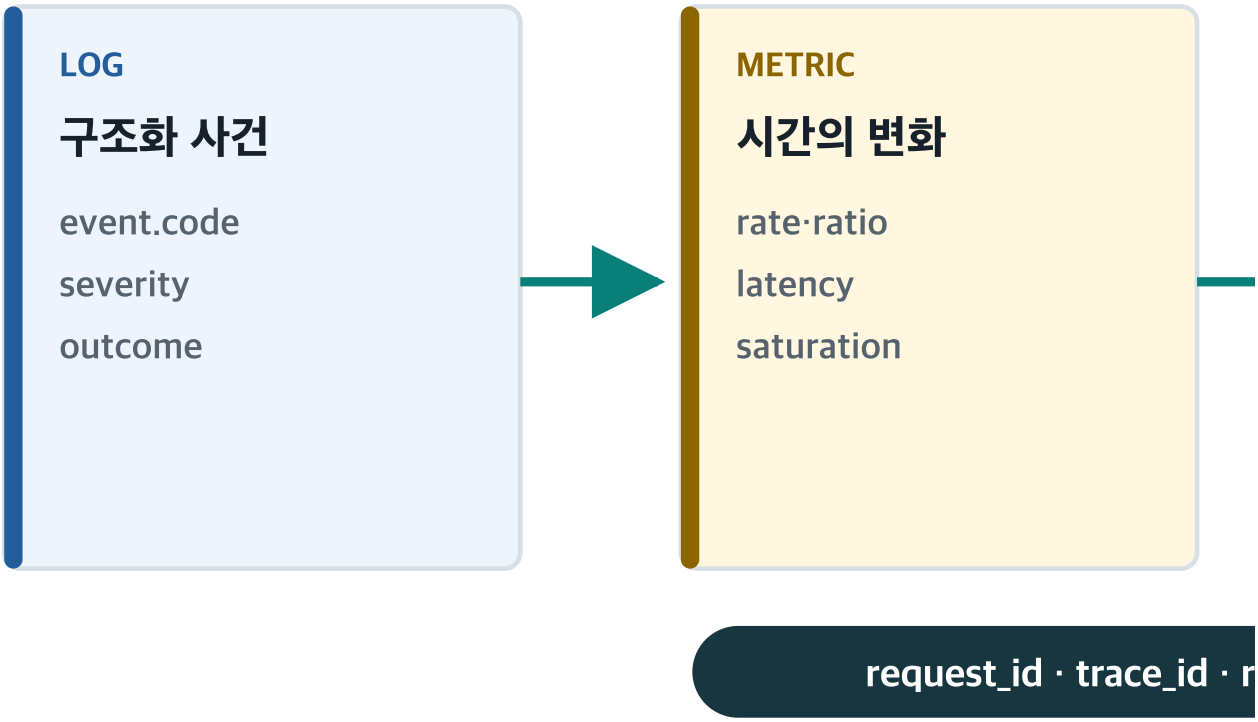


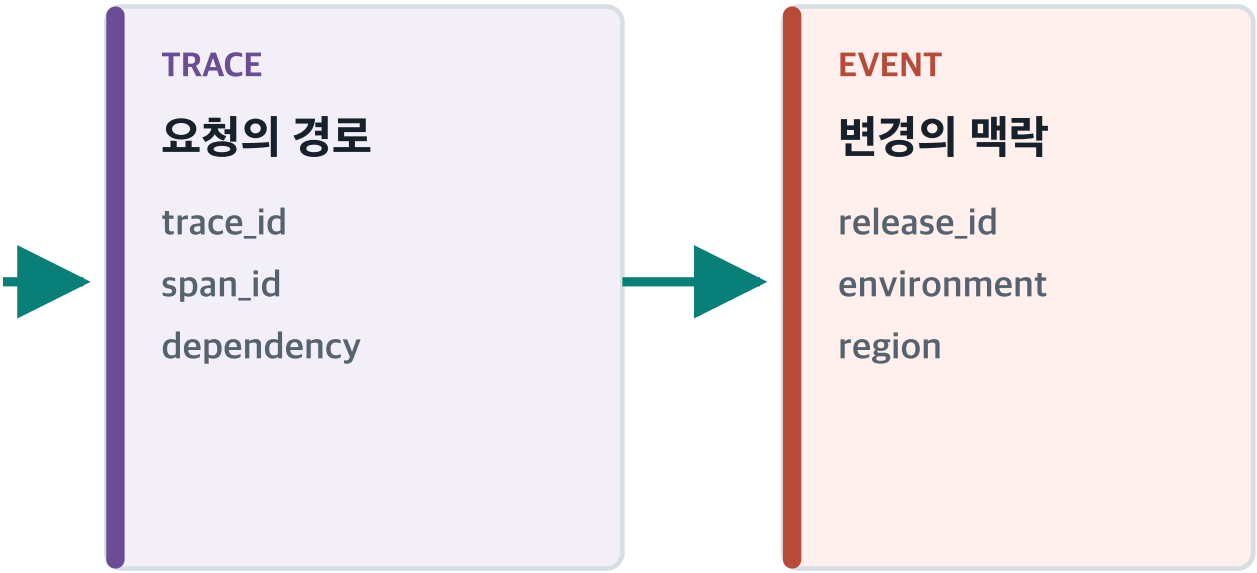
그림 3. Log는 사건, metric은 시간의 변화, trace는 요청 경로, change event는 release 맥락을 보여 줍니다. 공통 context가 있어야 네 시선을 한 incident로 묶을 수 있습니다.

Log metric trace change event로 [illegible]

각 signal은 목적이 다르며 공통 context가 있어야 사용자 증상에서 원인 후



:보까지 이동할 수 있습니다.



elease_id로 한 사건 연결

4.1 Signal마다 잘하는 일이 다르다

Signal	가장 잘 답하는 질문	약점	핵심 context
Log	정확히 어떤 사건이 일어났는가	대량·문구 drift·민감정보	event code·request·trace
Metric	언제 얼마나 변했는가	개별 사건 detail 부족	service·env·release·region
Trace	요청이 어디서 느리거나 실패했는가	Sampling·비용·전파 누락	trace·span·dependency
Change event	무엇이 바뀌었는가	영향 자체는 모름	release·config·owner

OpenTelemetry는 logs·metrics·traces를 서로 다른 signal로 설명하고 context propagation으로 distributed service 경계에서 trace 관계를 전달합니다. 이 문서는 특정 backend보다 signal 의미와 공통 context를 먼저 고정합니다.

4.2 한 요청의 상관관계

```
public request
  request_id=req-001
  trace_id=abc...
  environment=prod
  release_id=rel-2407
    ├ edge span
    ├ application span
    │    └ event.code=CHECKOUT_FAILED
    └ database span
         dependency=orders-db
```

이 연결이 있으면 dashboard의 error spike에서 candidate release를 고르고, trace에서 느린 dependency를 찾고, log에서 exact event를 확인할 수 있습니다.

4.3 Context를 넣을 곳과 넣지 않을 곳

Context	Log field	Metric label	Trace attribute	주의
Service·environment	좋음	좋음	좋음	값 집합 고정
Release ID	좋음	신중히 사용	좋음	Active release 수 제한
Route template	좋음	좋음	좋음	Raw URL 금지
Request ID	좋음	금지	좋음	High cardinality

Context	Log field	Metric label	Trace attribute	주의
User ID	원칙적으로 제거·대체	금지	원칙적으로 제거	개인정보
Secret·token	금지	금지	금지	Credential 노출

4.4 Semantic convention

OpenTelemetry Semantic Conventions는 흔한 operation과 resource의 attribute 이름·의미를 맞추는 계약입니다. 2026. 7. 16. 검토한 문서는 version 1.43.0입니다. Version은 바뀌므로 실제 적용 전 현재 문서와 사용하는 SDK·instrumentation version을 다시 확인합니다.

4.5 빠른 점검

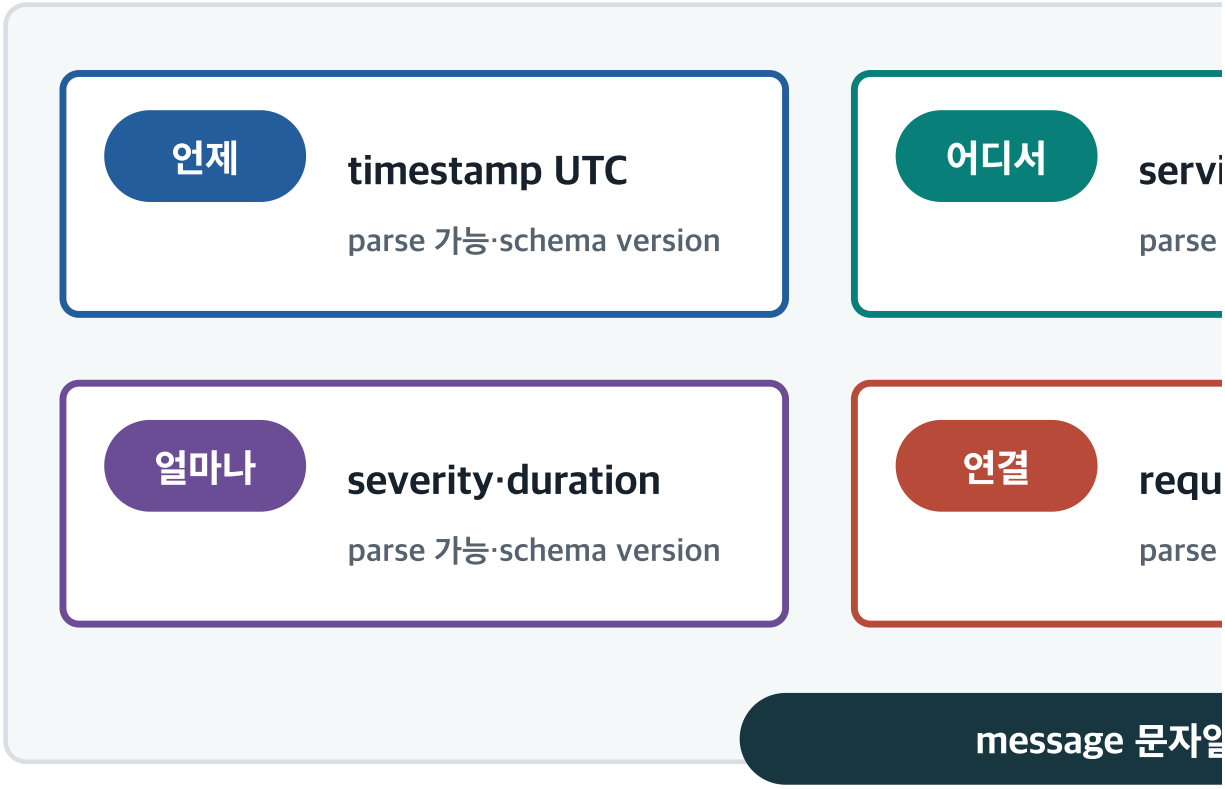
- [ ] Service·environment·release가 signal에 연결된다.
- [ ] Request와 trace를 log에서 찾을 수 있다.
- [ ] Dependency 이름이 inventory와 일치한다.
- [ ] Raw URL·user ID를 metric label로 쓰지 않는다.
- [ ] Sampling policy와 error trace 보존 원칙이 있다.

---

5. 구조화 Log는 사건 계약이다

그림 4. 좋은 log는 사람이 읽는 긴 문장이 아니라 timestamp·service·event code·outcome·correlation·sanitization field가 있는 versioned 사건 계약입니다.

UTC 시간·severity·event code·outcome·context를 기계가 안정적으로 읽



을 수 있어야 합니다.


ice·env·release
가능·schema version
무엇
event.code·outcome
parse 가능·schema version
est·trace·span
가능·schema version
안전
mask·allowlist
parse 가능·schema version
보다 field 계약

5.1 최소 Event Schema

Field	이유	좋은 예	나쁜 예
`timestamp`	사건 순서 비교	RFC 3339 UTC	Local time 문자열
`severity`	기술 중요도	INFO·WARN·ERROR	임의 단어
`event.code`	안정적 집계	`PAYMENT_DECLINED`	Message parsing
`outcome`	결과 분류	success·failure·unknown	Severity로 결과 추정
`service.name`	Source 식별	inventory exact name	Pod 임시 이름만
`environment`	경계 식별	production	누락
`release.id`	변경 연결	exact release	latest
`request.id` · `trace.id`	상관관계	format 검증 ID	사용자 email

5.2 Severity와 Outcome은 다르다

`ERROR` severity가 항상 업무 실패는 아닙니다. Retry로 최종 성공할 수 있습니다. 반대로 HTTP 200이어도 잘못된 업무 결과라면 user journey는 실패할 수 있습니다.

```
{
  "severity": "WARN",
  "event.code": "PAYMENT_RETRY_SCHEDULED",
  "outcome": "unknown",
  "attempt": 2
}
```

5.3 기록할 Event

- 중요한 user journey 시작·완료·실패
- Authentication·authorization 판단 결과
- 중요한 configuration·release 변화
- Dependency timeout·circuit open·retry exhaustion
- Backup·restore·catalog·integrity 결과
- Incident declaration·severity·role·decision

5.4 기록하지 않을 Event

- 매 loop의 반복 debug message
- Password·token·private key
- 전체 request·response body
- User가 입력한 free-form 내용을 그대로 붙인 message
- Metric으로 더 적합한 고빈도 수치 변화

5.5 Schema 변경

Consumer query와 alert가 event field에 의존할 수 있습니다.

```
새 field 추가
→ backward-compatible reader 확인
→ schema version 갱신
→ dashboard·alert query test
→ old field deprecation 기간
→ removal evidence
```

6. Telemetry도 보호해야 할 Data다

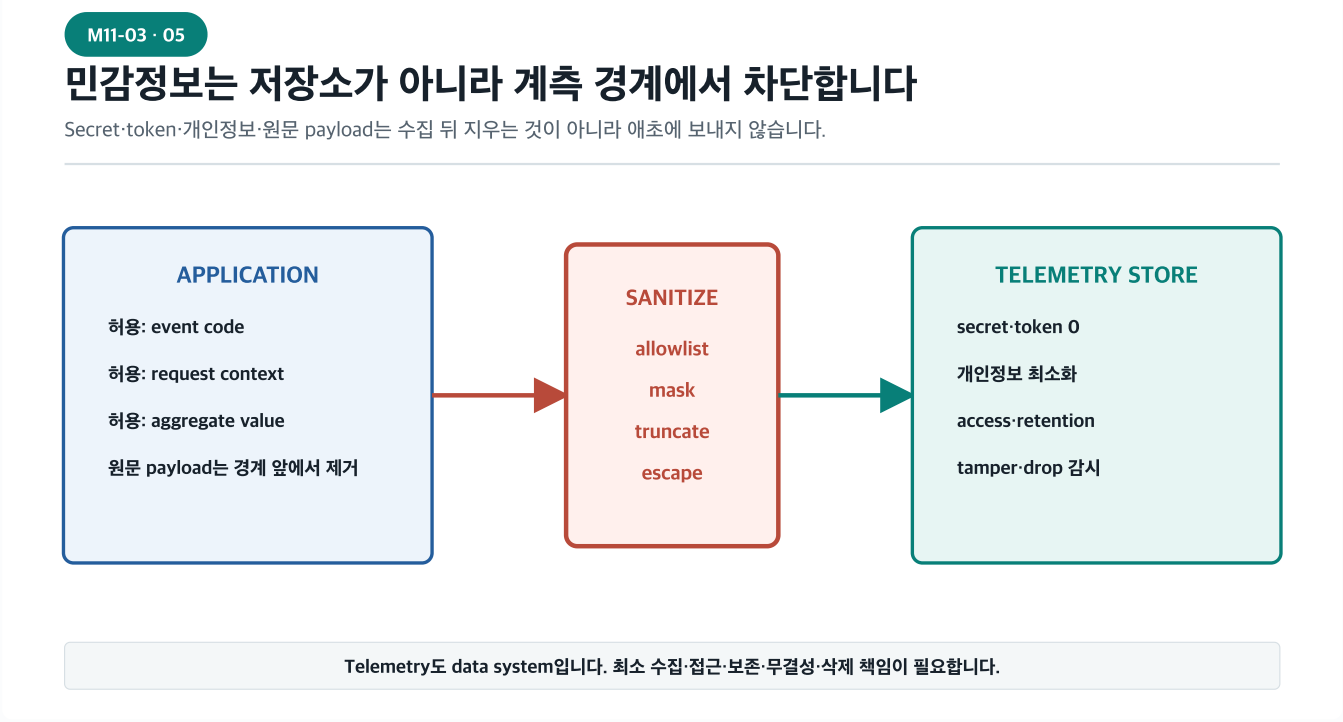
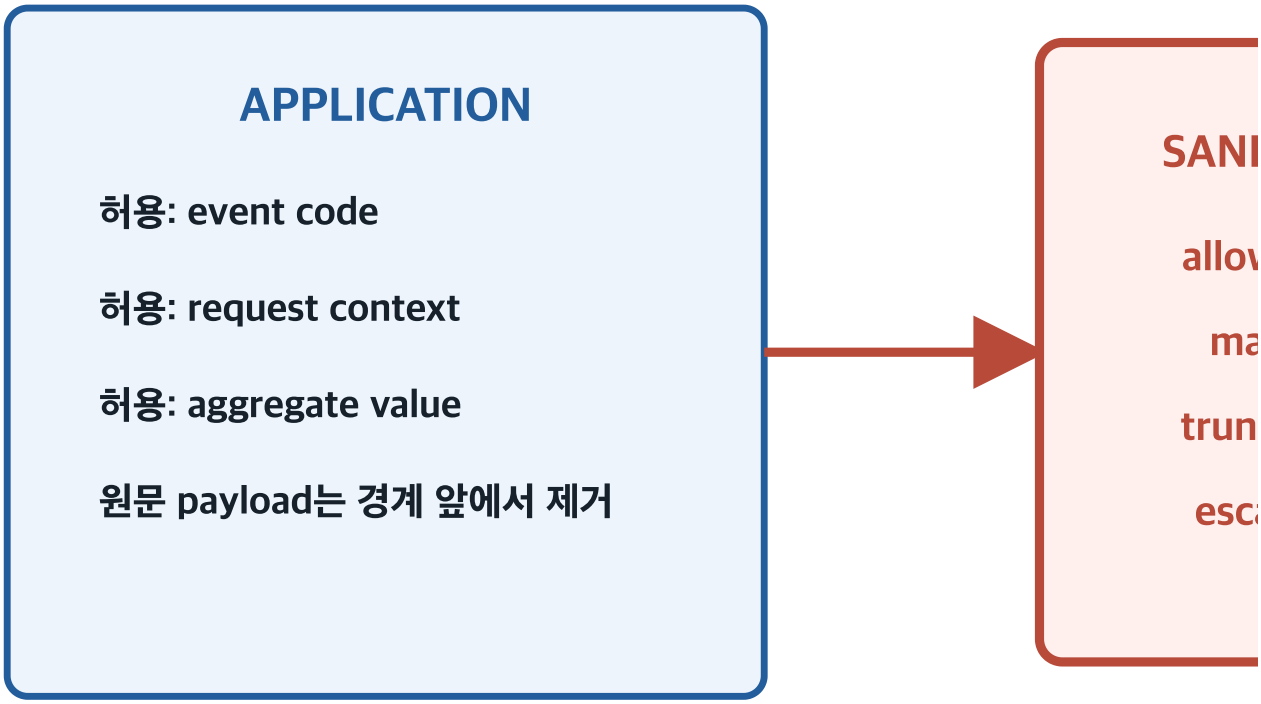


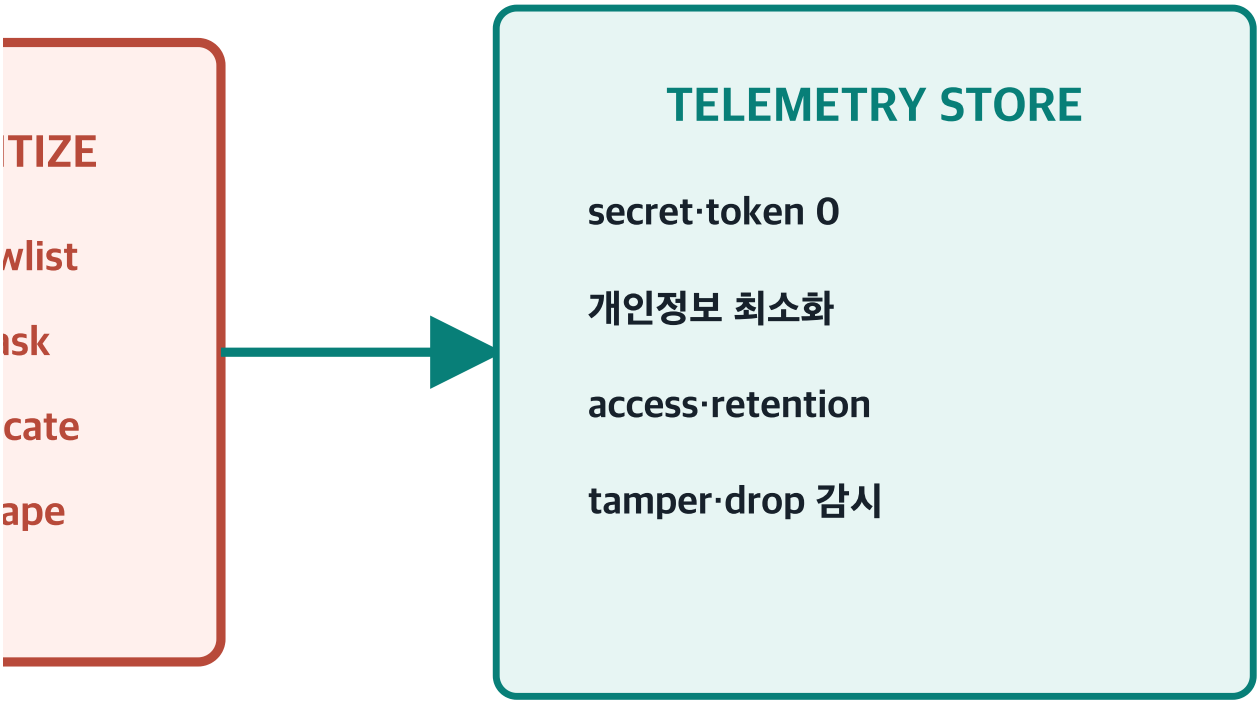
그림 5. Secret·token·개인정보·원문 payload는 저장소에서 나중에 지우지 않습니다. Application과 collector의 allowlist·mask·truncate·escape 경계에서 먼저 차단합니다.

[illegible]

Secret·token·개인정보·원문 payload는 수집 뒤 지우는 것이 아니라 애초에



에 보내지 않습니다.


TIZE
wlist
sk
cate
ape
TELEMETRY STORE
secret·token 0
개인정보 최소화
access·retention
tamper·drop 감시

6.1 왜 Log가 위험한가

Log는 많은 사람이 검색하고 여러 backend로 복제되며 긴 기간 남을 수 있습니다. Application database보다 접근자가 넓어질 수도 있습니다. OWASP Logging Cheat Sheet는 인증·권한·입력 검증 등 중요한 event를 기록하되 access token·password·민감한 개인정보 같은 data를 직접 기록하지 않도록 안내합니다.

6.2 Allowlist가 기본이다

```
승인한 field만 기록
→ type·length 검증
→ sensitive pattern 제거·mask
→ newline·control character escape
→ collector에서 2차 filter
→ 합성 canary로 누출 test
```

Blocklist만 사용하면 새 field와 변형된 이름을 놓치기 쉽습니다.

6.3 자유 입력과 Log Injection

공격자가 newline이나 control character를 넣으면 가짜 log line을 만들거나 viewer 구조를 깨뜨릴 수 있습니다.

입력	안전 처리
Free-form message	Length 제한·escape
Header	Allowlist·known format
URL	Route template·query 제거
Exception	Stack 제한·sensitive pattern 제거
Object	필요한 scalar field만 선택

6.4 Access·Retention·Integrity

통제	질문	Evidence
Access	누가 어떤 service·기간을 보는가	Role review·audit log
Retention	왜 며칠 보존하는가	Policy·lifecycle
Export	어디로 복제되는가	Destination inventory

통제	질문	Evidence
Integrity	무단 변경·삭제를 아는가	Tamper·deletion alert
Deletion	만료와 요청을 처리하는가	Lifecycle·deletion evidence

6.5 합성 누출 Test

실제 secret를 넣지 않습니다. `SYNTHETIC_SECRET_CANARY` 같은 무해한 marker를 event 후보에 넣고 sanitizer 이후 0건인지 검사합니다.

> 실제 token을 "검사 목적"으로 log에 넣지 않습니다. 누출 test도 가짜 canary만 사용합니다.

---

7. SLI·SLO·Error Budget을 사용자 여정에서 시작하기

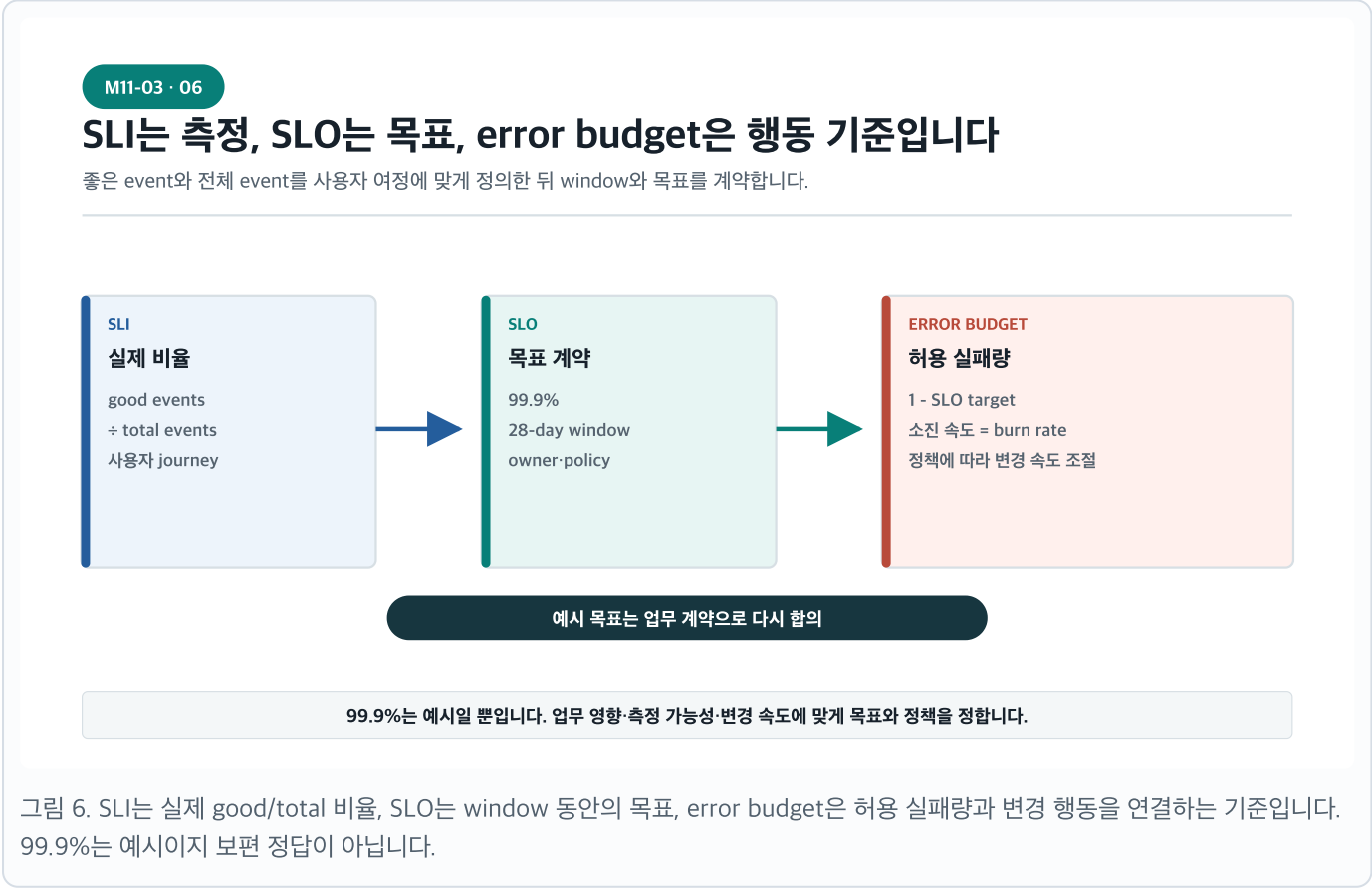
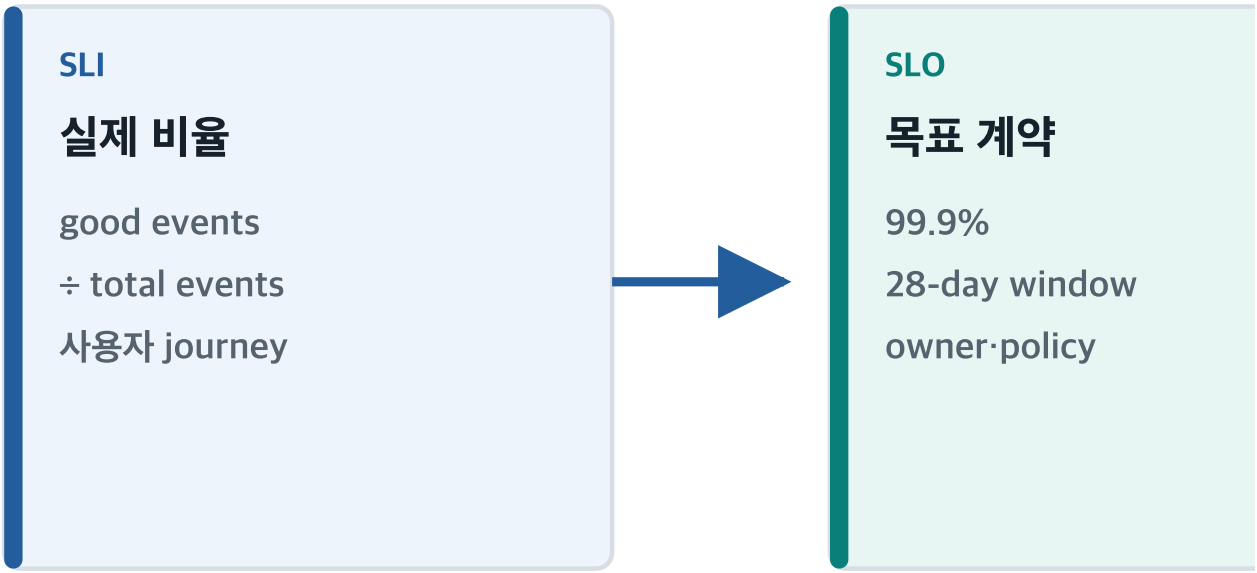


그림 6. SLI는 실제 good/total 비율, SLO는 window 동안의 목표, error budget은 허용 실패량과 변경 행동을 연결하는 기준입니다. 99.9%는 예시이지 보편 정답이 아닙니다.

SLI·SLO·Error Budget

좋은 event와 전체 event를 사용자 여정에 맞게 정의한 뒤 window와 목표



예시 목표는 업무 기

를 계약합니다.

ERROR BUDGET

허용 실패량

1 - SLO target

소진 속도 = burn rate

정책에 따라 변경 속도 조절

계약으로 다시 합의

7.1 먼저 User Journey를 고른다

Infrastructure CPU가 정상이어도 사용자가 결제를 완료하지 못하면 서비스는 신뢰할 수 없습니다.

Journey	Good event	Total event	제외
Search	결과가 1초 안에 반환	Eligible search request	명시된 bot·test
Checkout	승인된 주문 생성	유효 checkout attempt	사용자 취소
File upload	검증된 object 저장	허용 크기 upload	Client가 중단
Background job	Deadline 안에 완료	Scheduled eligible job	승인 maintenance

제외 조건은 실패를 숨기지 않도록 version과 owner를 둡니다.

7.2 계산

```
SLI = good events / total eligible events
SLO target = 목표 good ratio
Error budget = 1 − SLO target
```

예시:

```
SLO target = 99.9%
Error budget = 0.1%
Total = 100,000
허용 failed events = 100
```

7.3 Window

Window	장점	주의
Rolling 28일	현재 신뢰성 반영	날짜마다 과거 data가 빠짐
Calendar 월	보고·계약과 쉬움	월초 reset 효과
짧은 운영 window	빠른 feedback	작은 sample·noise

7.4 목표는 어떻게 정하는가

- 사용자가 실제로 느끼는 최소 품질
- Business impact와 contractual obligation
- 현재 측정 정확도
- Architecture와 팀의 복구 능력
- Reliability 투자와 변경 속도의 trade-off

무조건 높은 목표는 좋은 목표가 아닙니다. 측정할 수 없거나 달성 비용이 지나치거나 실제 사용자 기대와 무관하면 잘못된 계약입니다.

7.5 Error Budget Policy

상태	의미	행동 예
Budget 충분	목표 안	계획된 변경 진행
Burn 상승	위험 증가	고위험 변경 review 강화
Budget 거의 소진	SLO 위험	Reliability 작업 우선
Budget 소진	목표 위반·임박	예외 외 release 제한·incident 대응

Google SRE Workbook의 example error budget policy도 조직 합의가 필요한 예시 정책입니다. 그대로 복사하지 말고 우리 owner·approval·release process에 맞춥니다.

---

8. Dashboard는 질문에서 시작한다

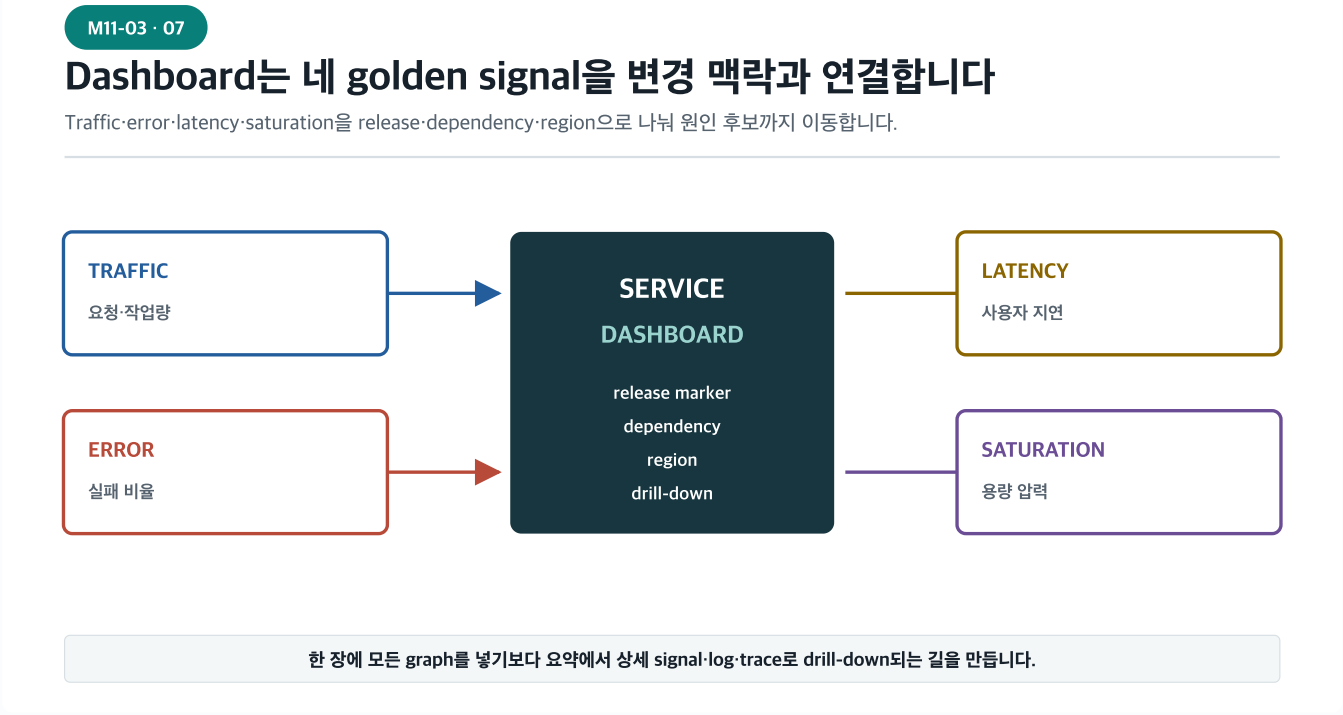
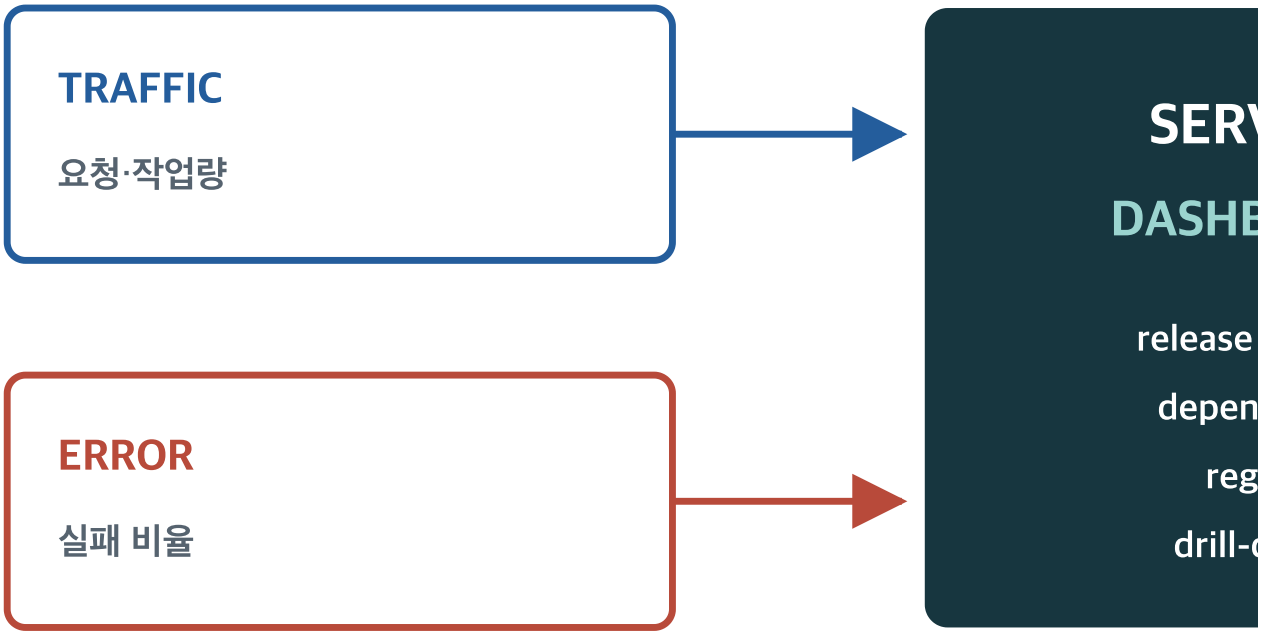


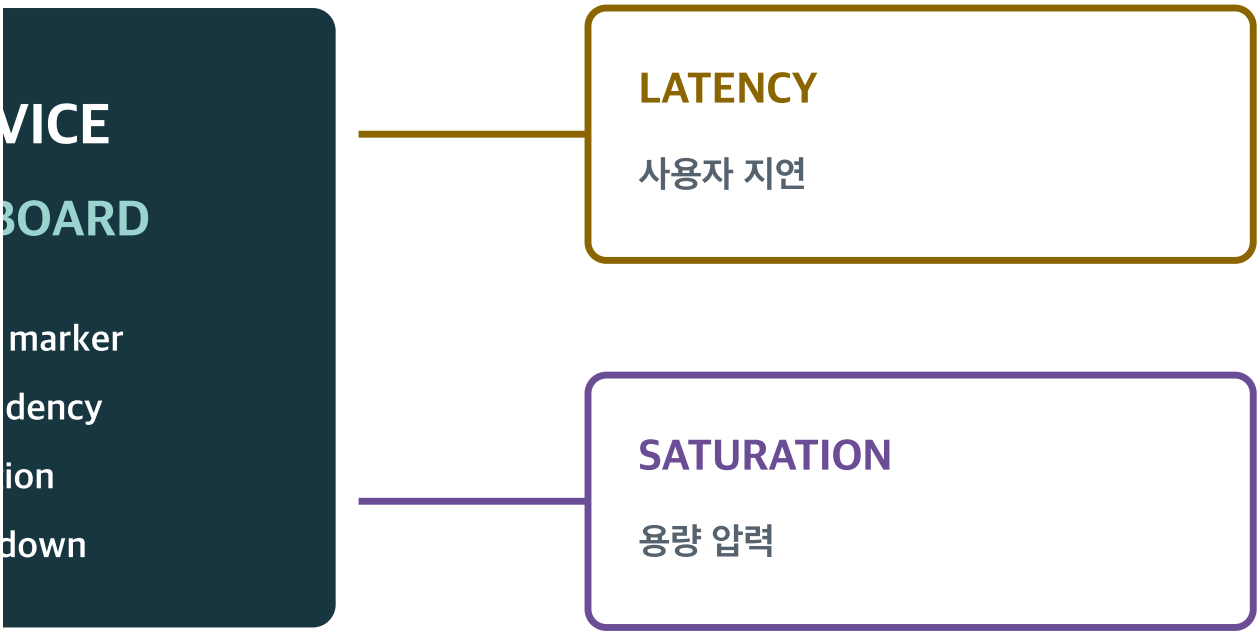
그림 7. Overview dashboard는 traffic·error·latency·saturation을 release marker·dependency·region과 연결하고, 상세 log·trace로 내려가는 길을 제공합니다.

Dashboard는 4개 golden signal을 보여

Traffic·error·latency·saturation을 release·dependency·region으로 나눠



| 원인 후보까지 이동합니다.



8.1 네 Golden Signal

Signal	질문	대표 측정	놓치기 쉬운 것
Traffic	얼마나 쓰는가	Request·job·message rate	업무량 단위
Error	얼마나 실패하는가	Error ratio·failed job	HTTP 200 업무 실패
Latency	얼마나 느린가	p50·p95·p99	성공·실패 분리
Saturation	한계에 얼마나 가까운가	CPU·memory·queue·connection	Quota·thread pool

8.2 Overview의 순서

1. User journey SLI·SLO
2. Error budget remaining·burn rate
3. Traffic·error·latency·saturation
4. Release·incident annotation
5. Dependency·region breakdown
6. Log·trace·runbook drill-down

8.3 Dashboard Anti-pattern

나쁜 화면	문제	고치기
Graph 50개 한 화면	판단 질문이 없음	Overview와 diagnostic 분리
평균 latency만	Tail 지연 숨김	Distribution·p95·p99
Instance별 CPU만	사용자 영향 단절	SLI와 연결
Release 표시 없음	변경 상관관계 단절	Annotation·release label
모든 URL label	Cardinality 폭증	Route template
Owner 없음	오래된 query 방치	Owner·review date·폐기 조건

8.4 Dashboard는 Alert가 아니다

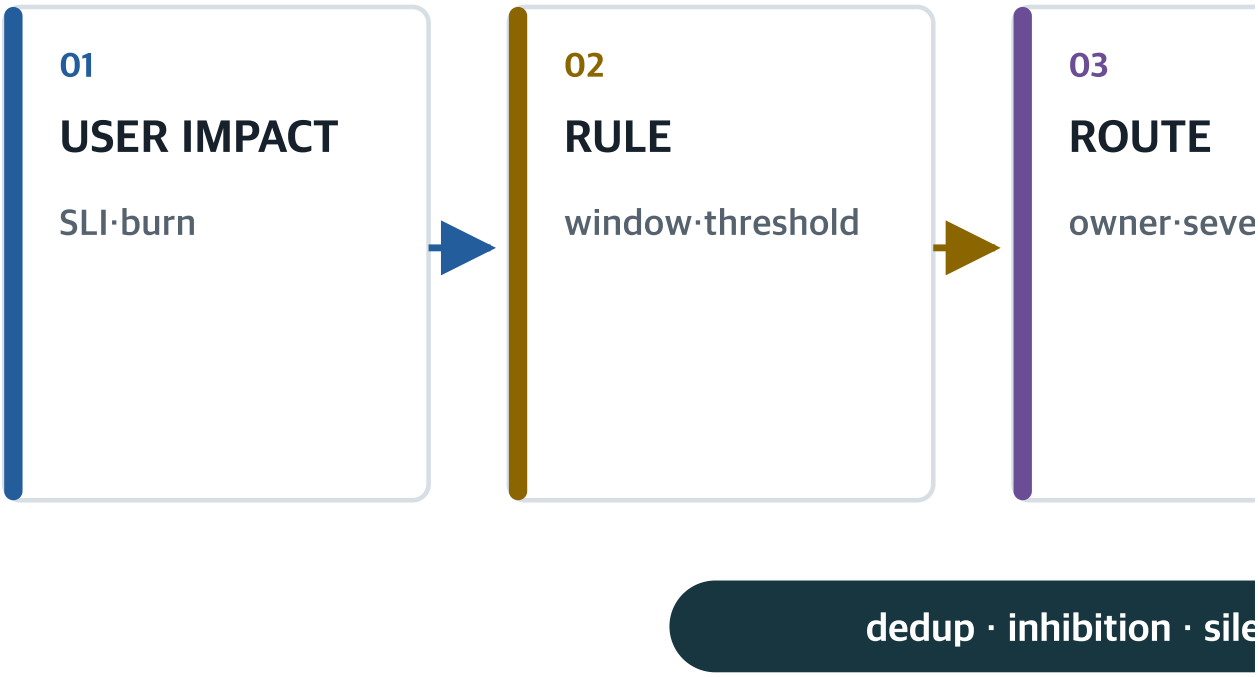
Dashboard는 사람이 보러 와야 합니다. 즉시 행동이 필요한 사용자 영향은 alert route로 연결해야 합니다. 반대로 행동이 없는 모든 warning을 page로 바꾸면 alert fatigue가 생깁니다.

9. 실행 가능한 Alert를 설계하기

그림 8. Actionable alert는 사용자 영향에서 시작해 rule·route·acknowledgement·runbook·복구 검증으로 이어집니다. Dedup·inhibition·silence expiry·route test가 noise와 누락을 줄입니다.

[illegible]

사용자 영향에서 시작해 rule·route·acknowledgement·runbook·복구 검증



[illegible]

증까지 이어집니다.

ence expiry · route test

9.1 Alert 한 건의 계약

Field	질문
User impact	어떤 journey가 얼마나 실패하는가
Signal·rule	어떤 query·window·threshold인가
Severity	즉시 page인가 업무 시간 ticket인가
Owner·route	현재 누가 받는가
Runbook	첫 5분 행동과 stop condition은 무엇인가
Ack·escalation	몇 분 안에 누가 다음으로 받는가
Recovery verify	무엇이 돌아와야 해제하는가

9.2 Fast·Slow Burn

한 window만 보면 짧은 spike에 자주 울리거나 느린 budget 소진을 놓칠 수 있습니다. Multi-window burn alert는 짧은·긴 window를 함께 사용해 즉시 대응과 지속 악화를 구분합니다.

유형	짧은 window	긴 window	행동
Fast burn	높음	높아지는 중	즉시 page
Slow burn	중간	높음	조기 조사·변경 조절
Transient spike	높음	낮음	Noise 확인

9.3 Noise Control

- **Deduplication:** 같은 alert 반복을 하나로 묶습니다.
- **Grouping:** 같은 service·incident의 관련 alert를 묶습니다.
- **Inhibition:** 상위 dependency 장애 때 파생 page를 억제합니다.
- **Silence:** 승인 maintenance 동안 일시 중지하되 owner·이유·만료를 둡니다.

Prometheus Alertmanager configuration은 routing·grouping·inhibition·silence 같은 기능을 제공합니다. 제품 기능을 켜는 것보다 label·owner·rule 관계를 설계하고 합성 test로 검증하는 것이 먼저입니다.

9.4 Route Test

실제 사람을 깨우지 않는 합성 channel에서 다음을 확인합니다.

```
rule fires
→ label complete
→ expected route
→ template safe
→ synthetic acknowledgement
→ escalation simulation
→ resolution notification
```

실제 notification을 보내지 않아도 route logic은 검증할 수 있습니다.

9.5 On-call의 지속 가능성

Google SRE Workbook의 on-call 안내는 업무량과 대응 역량·훈련·escalation을 함께 봅니다. On-call은 이름을 schedule에 넣는 일이 아닙니다.

- 충분한 training과 shadow rotation
- 명확한 service ownership
- 감당 가능한 page 수
- 최신 runbook
- Handoff와 active risk 공유
- Incident 뒤 회복 시간과 개선

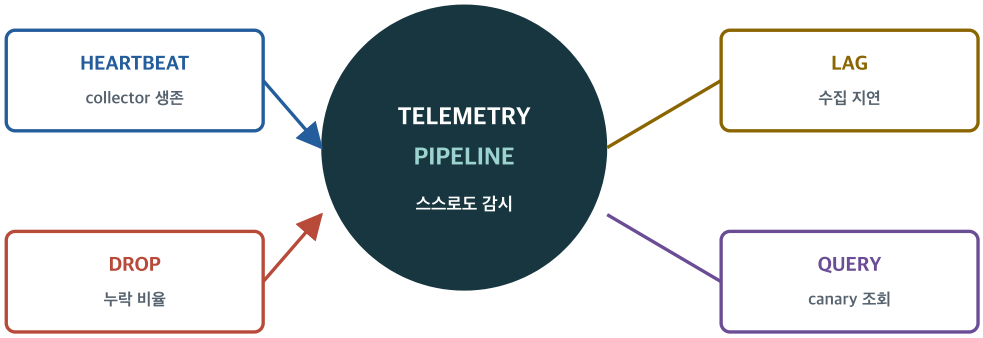
---

10. Monitoring을 Monitoring하기

M11-03 · 09

Monitoring이 멈추면 서비스가 조용해진 것처럼 보일 수 있습니다

Collector heartbeat·drop·lag·query canary를 별도 경로로 감시해 관측 사각지대를 드러냅니다.

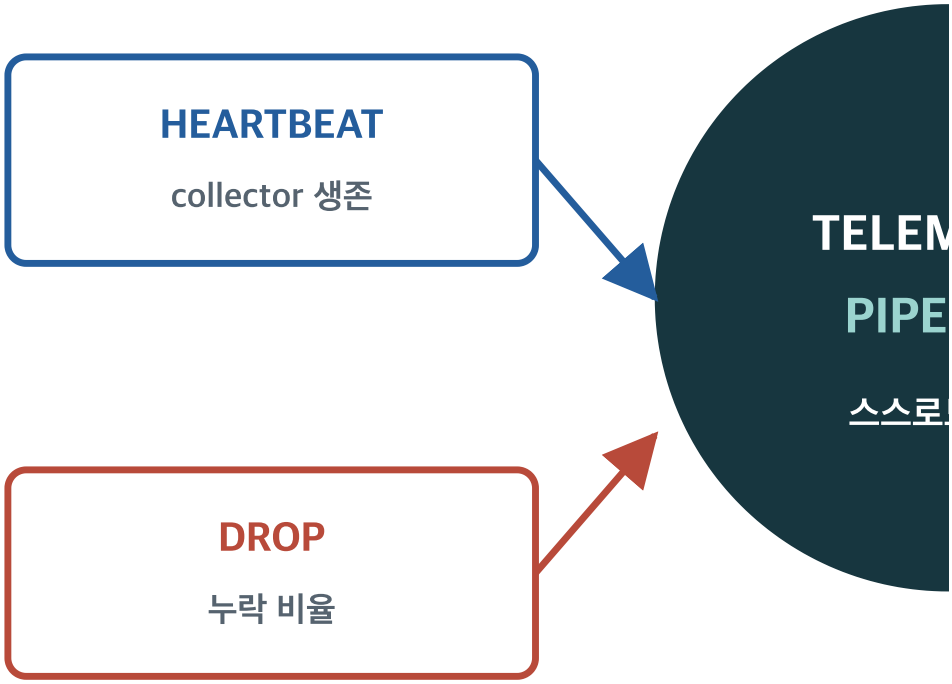


'신호 없음'과 '문제 없음'을 구분하려면 관측 pipeline 자체의 SLI와 alert가 필요합니다.

그림 9. Collector heartbeat·drop rate·ingestion lag·query canary가 있어야 "신호 없음"과 "문제 없음"을 구분할 수 있습니다.

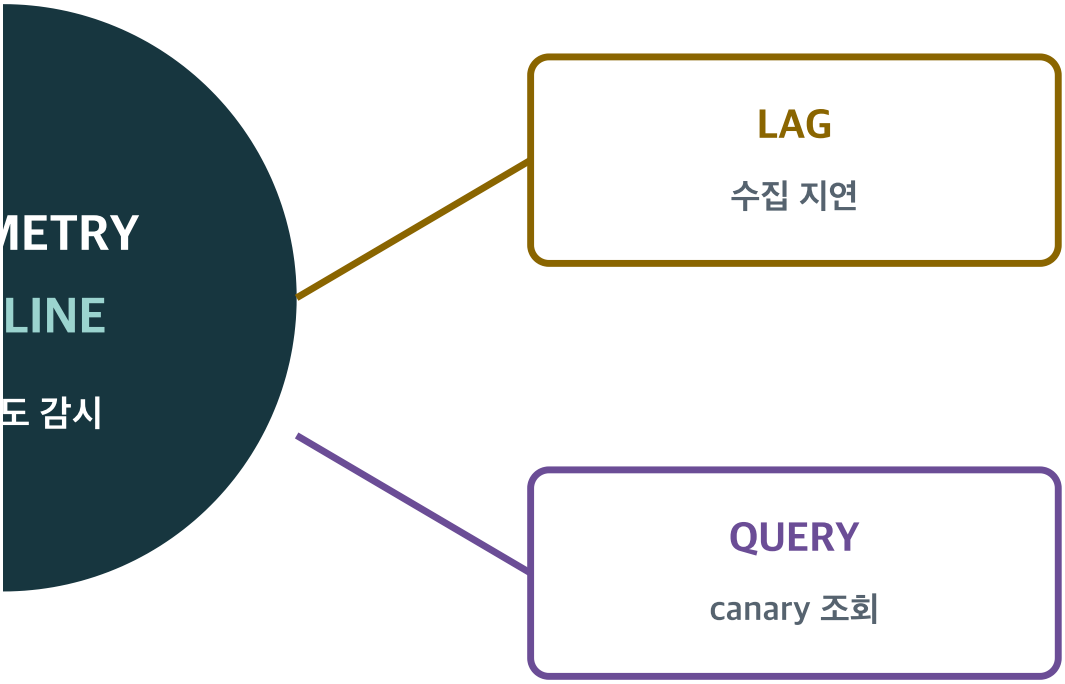
Monitoring 수집부터 대시보드까지

Collector heartbeat·drop·lag·query canary를 별도 경로로 감시해 관측 사



은 것까지 드러낼 수 있습니다

사각지대를 드러냅니다.



10.1 조용함의 두 의미

```
Case A: 서비스가 정상이라 error event가 없다.
Case B: collector가 멈춰 error event가 도착하지 않는다.
```

Dashboard만 보면 둘 다 0처럼 보일 수 있습니다.

10.2 Pipeline SLI

구간	Signal	질문
SDK·agent	Export error·queue	생성했지만 보내지 못했는가
Collector	Heartbeat·CPU·memory	Process가 살아 처리하는가
Export	Retry·drop·failure	Backend로 전달되는가
Ingestion	Lag·accepted count	언제 query 가능한가
Storage	Retention·deletion	기대 기간과 무결성인가
Query	Canary success·latency	실제로 찾을 수 있는가

10.3 별도 경로

가능하면 monitoring pipeline health는 같은 실패 domain에만 의존하지 않습니다. 예를 들어 collector heartbeat와 query canary가 같은 collector 하나를 모두 거치면 그 collector 전체 실패를 놓칠 수 있습니다.

10.4 Black-box와 White-box

방식	보는 것	장점	한계
Black-box	DNS·TLS·HTTP·업무 결과	사용자 관점	원인 detail 적음
White-box	Internal metric·log·trace	진단 detail	사용자 경로와 다를 수 있음

둘을 함께 사용합니다. Public health는 내부 CPU만으로 증명되지 않고, black-box 실패의 원인은 내부 signal 없이 찾기 어렵습니다.

10.5 Synthetic 안전 경계

- 전용 합성 identity·data
- Production state를 바꾸지 않는 read-only 또는 cleanup 가능한 step
- Rate·timeout 제한
- 실제 고객 notification 0
- Secret·개인정보 0
- 본 실습은 외부 network도 0

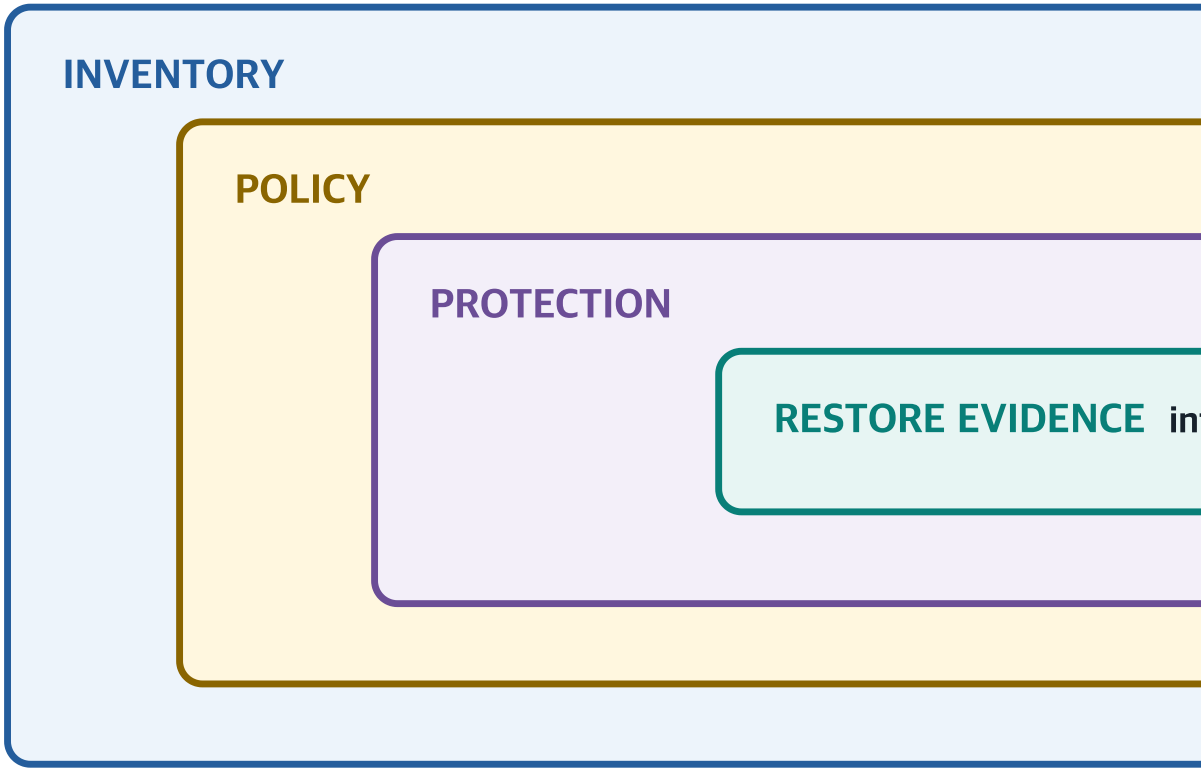
---

11. Backup은 사본보다 복구 계약이다

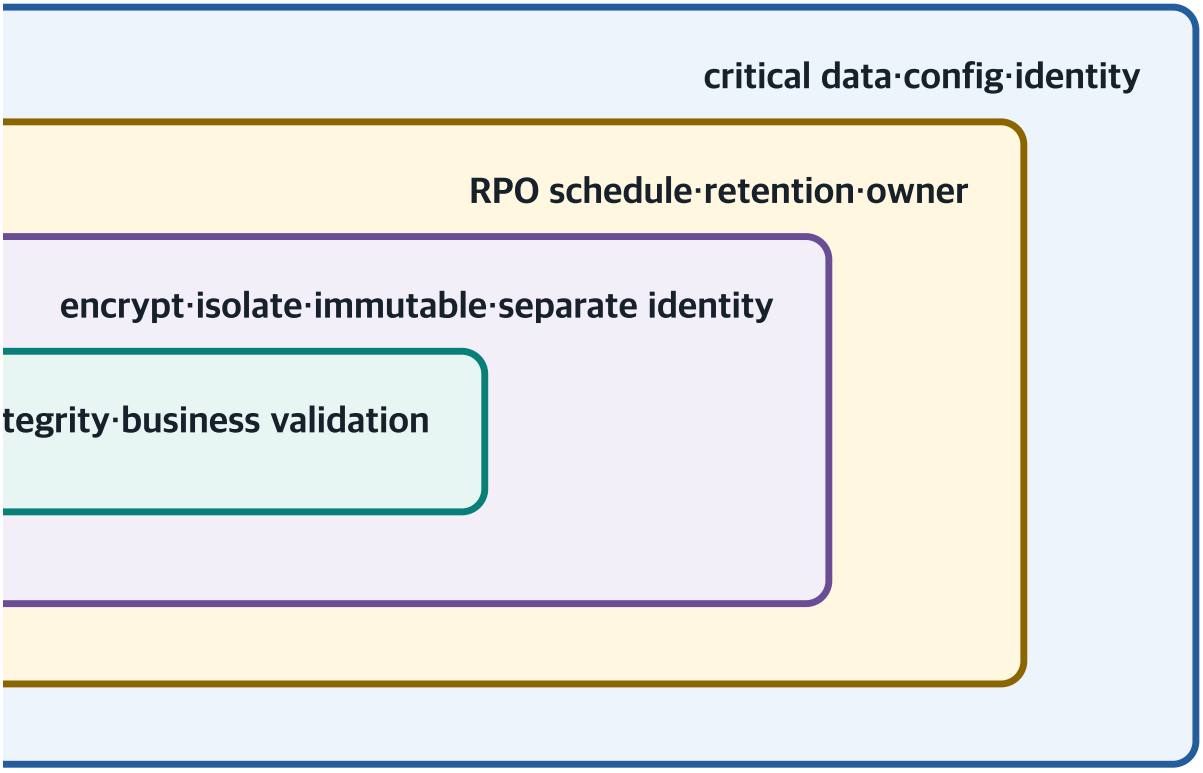
그림 10. Backup은 critical asset inventory에서 시작해 RPO schedule·retention·encryption·isolation·immutability·separate identity를 거쳐 restore evidence로 완성됩니다.

Backup은 "파일 복사"가 아니라 "복구 가능성"을 관리하는 체계

중요 자산 inventory에서 시작해 schedule·retention·격리·불변성·별도 id



entity를 위험에 맞게 설계합니다.



11.1 Backup의 네 질문

1. **무엇을:** Data뿐 아니라 configuration·identity material·catalog도 포함하는가.
2. **언제까지:** RPO를 만족하도록 얼마나 자주 만드는가.
3. **어떻게 보호:** Production 침해·오삭제·key 손실이 copy로 번지지 않는가.
4. **실제로 복원:** 격리 target에서 integrity와 업무 결과를 검증했는가.

11.2 Critical Asset Inventory

Asset	빠질 때 영향	Backup 형태	특별 검증
Primary database	Transaction 손실	Snapshot·log	Schema·record 관계
Object·file	사용자 content 손실	Versioned copy	Checksum·sample open
Configuration	Service 재현 실패	Version control·export	Exact version
Identity material	접근·복호화 실패	보호된 config·key plan	Separate recovery
Queue·event	처리 누락·중복	Service별 전략	Idempotency·replay
Backup catalog	Recovery point 탐색 실패	별도 inventory	Search·owner

11.3 Backup 종류를 목적에 맞춘다

방식	장점	주의
Full backup	단순한 recovery chain	시간·storage 큼
Incremental	빠르고 storage 절약	Chain 의존·복원 복잡
Differential	Full 이후 변화 묶음	시간이 갈수록 증가
Snapshot	빠른 point-in-time	같은 failure domain 여부
Transaction log	세밀한 RPO	연속성·catalog·replay 검증

Provider feature 이름보다 recovery objective와 dependency를 먼저 씁니다.

11.4 Schedule과 RPO

```
RPO 15분
Backup interval 60분
= 정상 schedule만으로도 목표 미달 가능
```

Interval만 짧다고 충분하지 않습니다. Job duration·queue·replication lag·catalog update·failure alert를 함께 봅니다.

11.5 Retention

Daily·weekly·monthly tier는 관습이 아니라 recovery scenario와 법적·비용 요구에서 나옵니다.

질문	기록
가장 오래 뒤 발견되는 오류는 무엇인가	Retention 근거
잘못된 data가 backup에 포함될 수 있는가	여러 시점 필요
삭제 의무가 있는가	Lifecycle·legal review
오래된 format을 복원할 수 있는가	Tool·schema compatibility

11.6 Job Success의 함정

다음 상태는 실패입니다.

```
job_success = true
checksum_verified = false
catalog_updated = false
unexpected_deletion = 2
restore_evidence = false
```

Backup 성공 기준은 최소한 scope·copy·catalog·integrity·protection·restore evidence를 포함합니다.

12. Backup Copy를 Production 실패에서 분리하기

12.1 보호 층

보호	막으려는 위험	확인
In-transit encryption	전송 중 노출·변조	TLS·endpoint
At-rest encryption	Storage 노출	Key·policy
Separate identity	Production admin 탈취 확산	Role·approval
Isolation	Network·account 침해 확산	Path·account boundary
Immutability	오삭제·ransomware 변조	Retention lock·권한
Offline·air gap	항상 연결된 공격 경로	Copy lifecycle
Key recoverability	장애 뒤 decrypt 불가	Recovery procedure

12.2 불변성은 마법이 아니다

Immutable copy는 정한 기간 변경·삭제를 어렵게 하지만 다음을 자동 해결하지 않습니다.

- 이미 오염된 data를 backup한 경우
- 잘못된 retention 기간
- 복호화 key 손실
- 복원 tool·format 불일치
- 업무 validation 부재
- 비용과 legal deletion conflict

위험과 규제·비용에 맞춰 선택하고 실제 provider의 mode·권한·retention behavior를 현재 공식 문서로 확인합니다.

12.3 Separate Identity

Production 관리자가 backup을 즉시 삭제할 수 있으면 같은 credential 침해가 원본과 copy를 함께 파괴할 수 있습니다.

```
production identity
  └ application·database operation

backup identity
  └ create·catalog·restore limited operation
       + separate approval
       + audit
```

12.4 CISA의 복구 관점

CISA StopRansomware Guide와 Cross-Sector Cybersecurity Performance Goals는 offline·encrypted backup과 복구 test 같은 기본 통제를 강조합니다. 실제 적용은 조직 위험·asset·법적 요구·provider 기능에 맞춰 구체화합니다.

---

13. RPO·RTO와 Recovery 순서

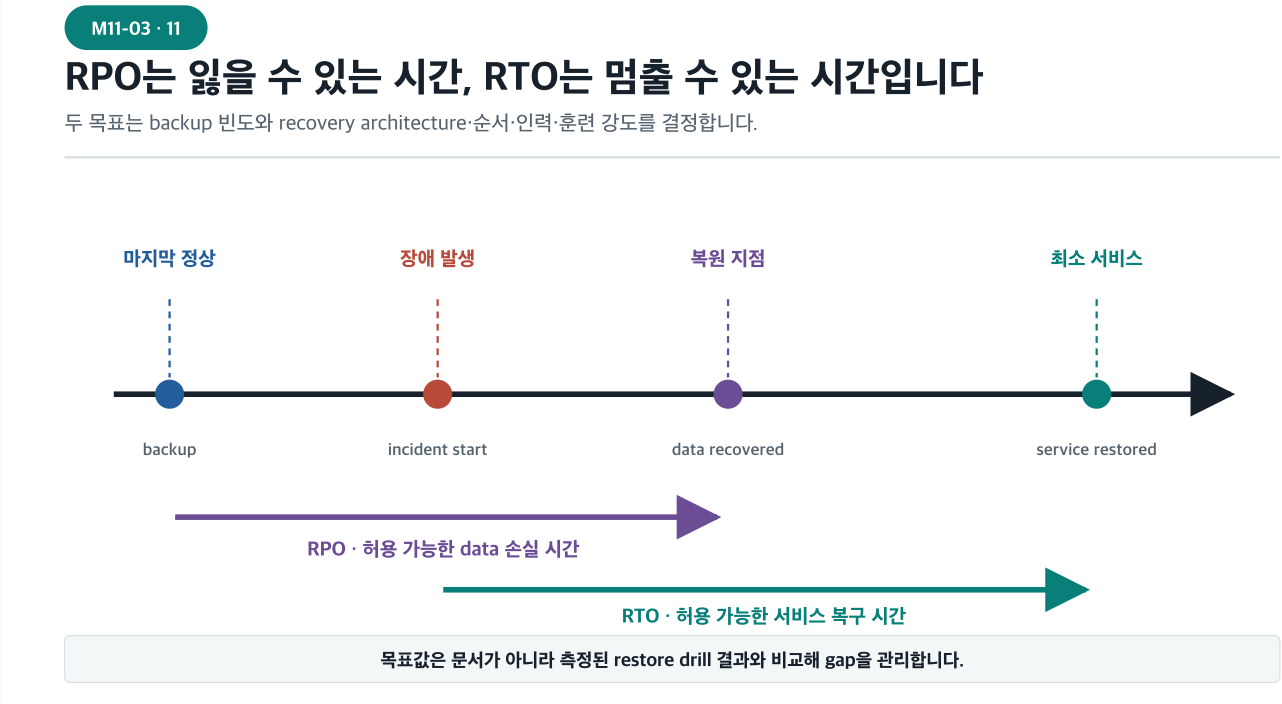


그림 11. RPO는 허용할 data 손실 시간, RTO는 허용할 service 중단 시간입니다. 목표는 문서 숫자가 아니라 restore drill에서 측정한 값과 비교해야 합니다.

RPO는 얼마까지 잃어도 되는가, RTO는 얼마

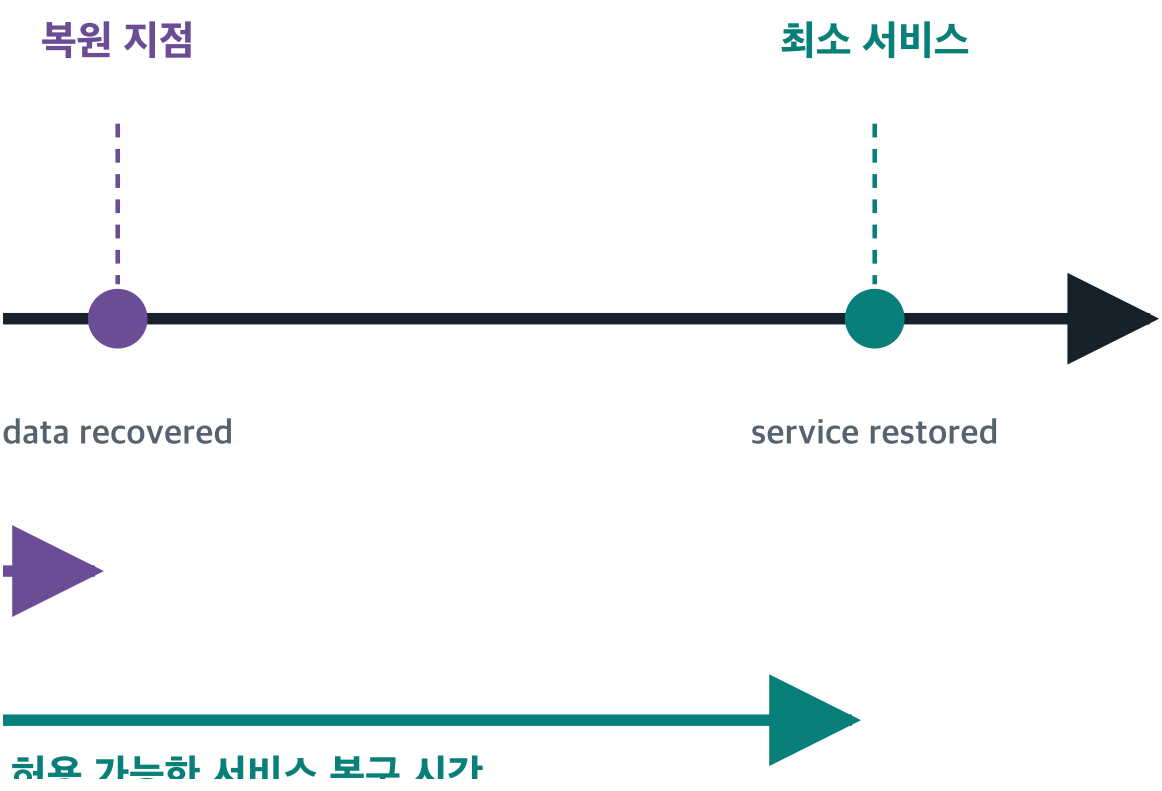
두 목표는 backup 빈도와 recovery architecture·순서·인력·훈련 강도를 결

RPO · 허용 가능한 data 손실 시간

RTO ·

있는 [illegible]

[illegible]정합니다.



13.1 두 목표의 차이

목표	질문	설계에 미치는 영향
RPO	몇 분의 data를 잃을 수 있는가	Backup·replication·log 빈도
RTO	몇 분 안에 최소 service를 복구할까	Architecture·automation·인력·순서

RPO 15분과 RTO 60분은 “15분 안에 복구”라는 뜻이 아닙니다.

13.2 Minimum Service

완전 복구보다 먼저 제공할 최소 기능을 정합니다.

Priority	기능	Minimum service 예
1	기존 주문 조회	Read-only·최근 일자
2	새 주문 접수	제한된 payment method
3	추천·분석	일시 중지

13.3 Dependency Order

일반적 예:

```
identity·key
→ network·name
→ primary data
→ queue·object
→ application
→ public path
→ background·optional feature
```

실제 순서는 service graph에 맞게 작성합니다. Application을 먼저 켜도 identity와 data가 준비되지 않으면 retry storm과 추가 손상이 생길 수 있습니다.

13.4 Objective를 정하는 질문

- 이 journey가 중단되면 시간당 영향은 무엇인가.
- Data를 몇 분 잃으면 되돌릴 수 없는가.

- 수동 복구에 필요한 사람과 권한이 실제로 있는가.
- 야간·휴일에도 같은 목표를 지킬 수 있는가.
- Dependency provider의 복구 목표와 우리 목표가 맞는가.
- 목표를 검증한 마지막 drill은 언제인가.

14. Restore Drill은 다섯 단계다

그림 12. Known-good point를 선택하고 production과 격리된 target에 dependency 순서대로 복원한 뒤 integrity·business result를 검증하고 훈련 자원을 정리합니다.

운영 데이터를 건드리지 않는 격리 target에서 dependency 순서와 busine

01

SELECT

known-good point

time·owner

02

ISOLATE

recovery target

time·owner

03

RESTORE

dependency

time·owner

RPO·RTO 측정 + evid

ess validation을 함께 검증합니다.

ence + action owner

14.1 Select

- Scenario와 복구 목표를 정합니다.
- Backup catalog에서 recovery point를 고릅니다.
- 왜 known-good인지 근거를 기록합니다.
- 필요한 key·tool·version을 확인합니다.

14.2 Isolate

Production과 분리된 account·network·namespace·target을 사용합니다.

- 실제 고객 traffic 0
- 실제 notification 0
- Production write path 0
- 전용 credential·만료
- Cleanup owner

14.3 Restore

Runbook의 dependency order를 실제로 따릅니다. 사람 머릿속에서 생략하지 않습니다.

기록	예
Start UTC	09:00
Recovery point	08:48
Tool·version	Approved restore tool
단계별 owner	Identity·DB·app
오류·retry	Timeline

14.4 Validate

Integrity validation과 **business validation**은 다릅니다.

Integrity	Business
Checksum	핵심 journey 성공

Integrity	Business
Schema	업무 규칙 일치
Record count	Reference 관계 정상
Constraint	예상 결과 반환
File open	사용자가 content 이용

14.5 Clean

훈련 target은 민감 data의 새로운 사본이 될 수 있습니다.

- Temporary access revoke
- Restored data·resource 삭제 evidence
- Network route 제거
- Tool output·log의 민감정보 확인
- Cost·inventory 정리

14.6 측정

```
measured RPO = incident point − restored data point
measured RTO = incident start − minimum service restored time
```

목표를 넘으면 “훈련 성공”이 아니라 recovery gap입니다. Owner·due·재검을 붙입니다.

15. Incident는 역할로 조정한다

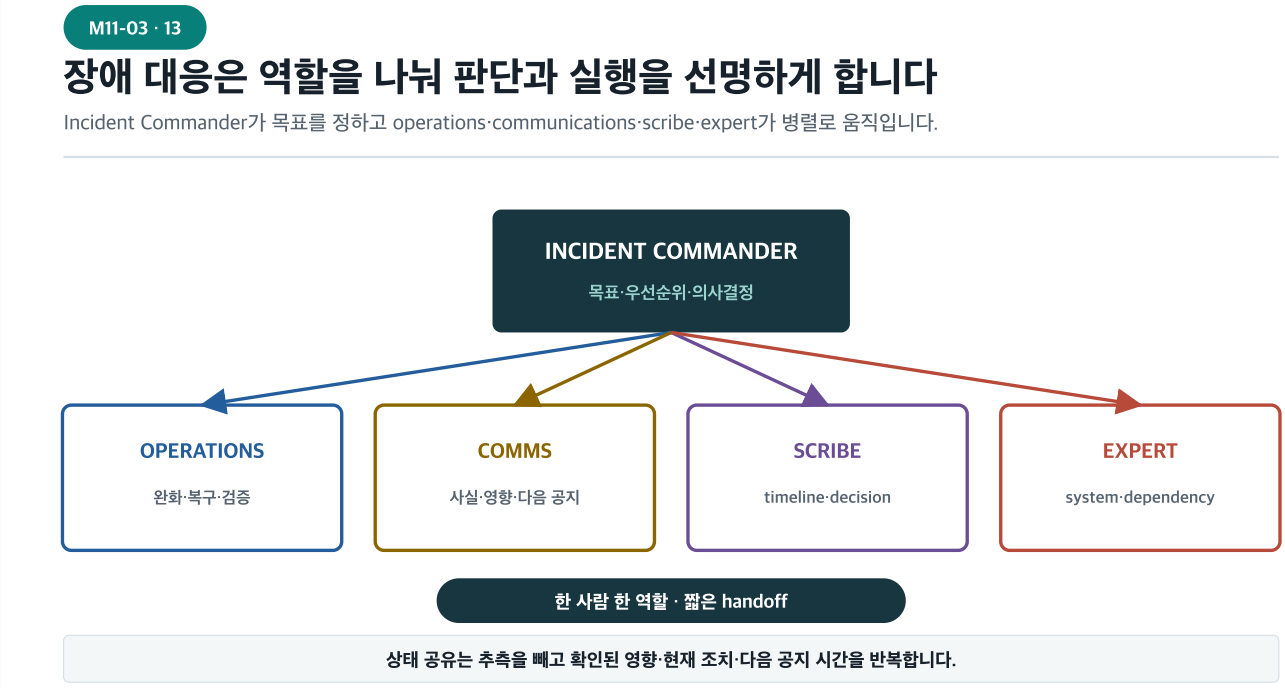
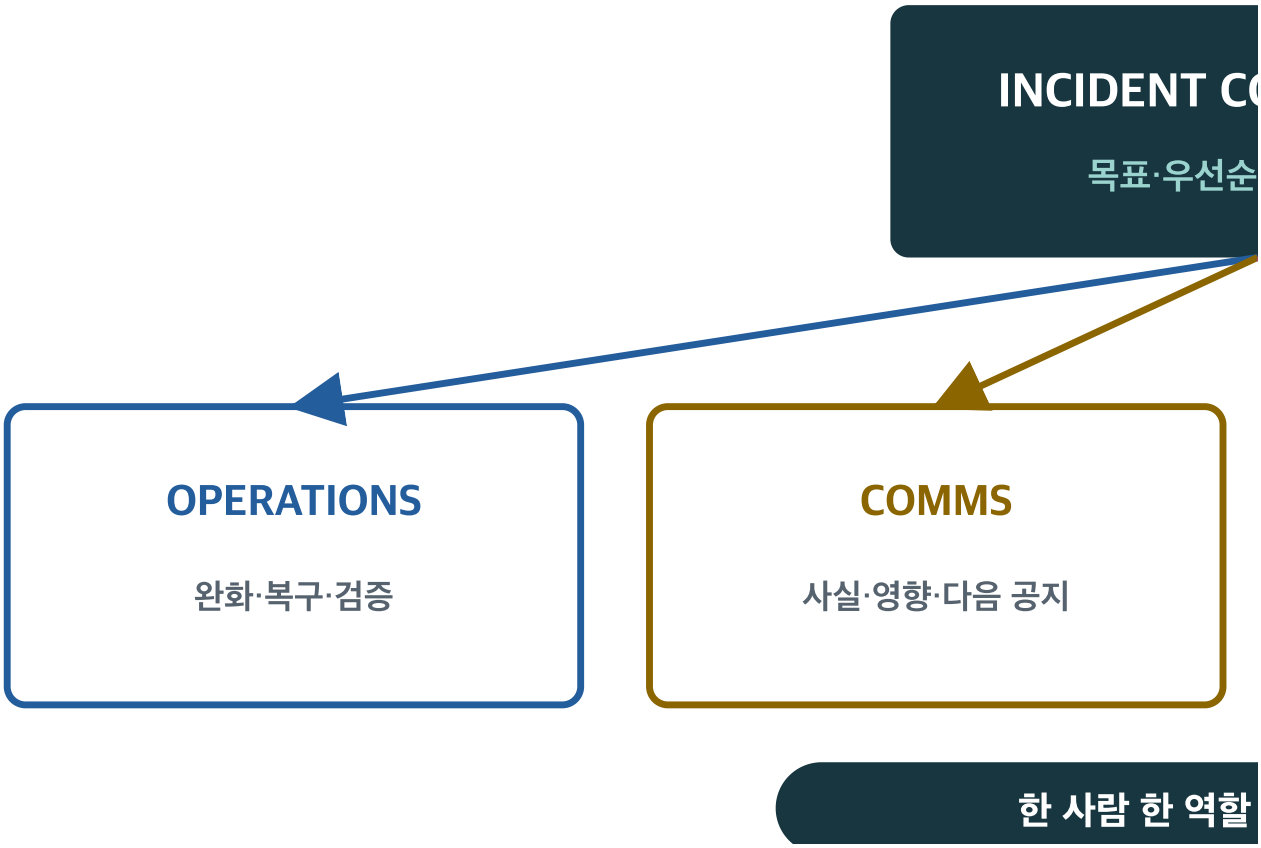


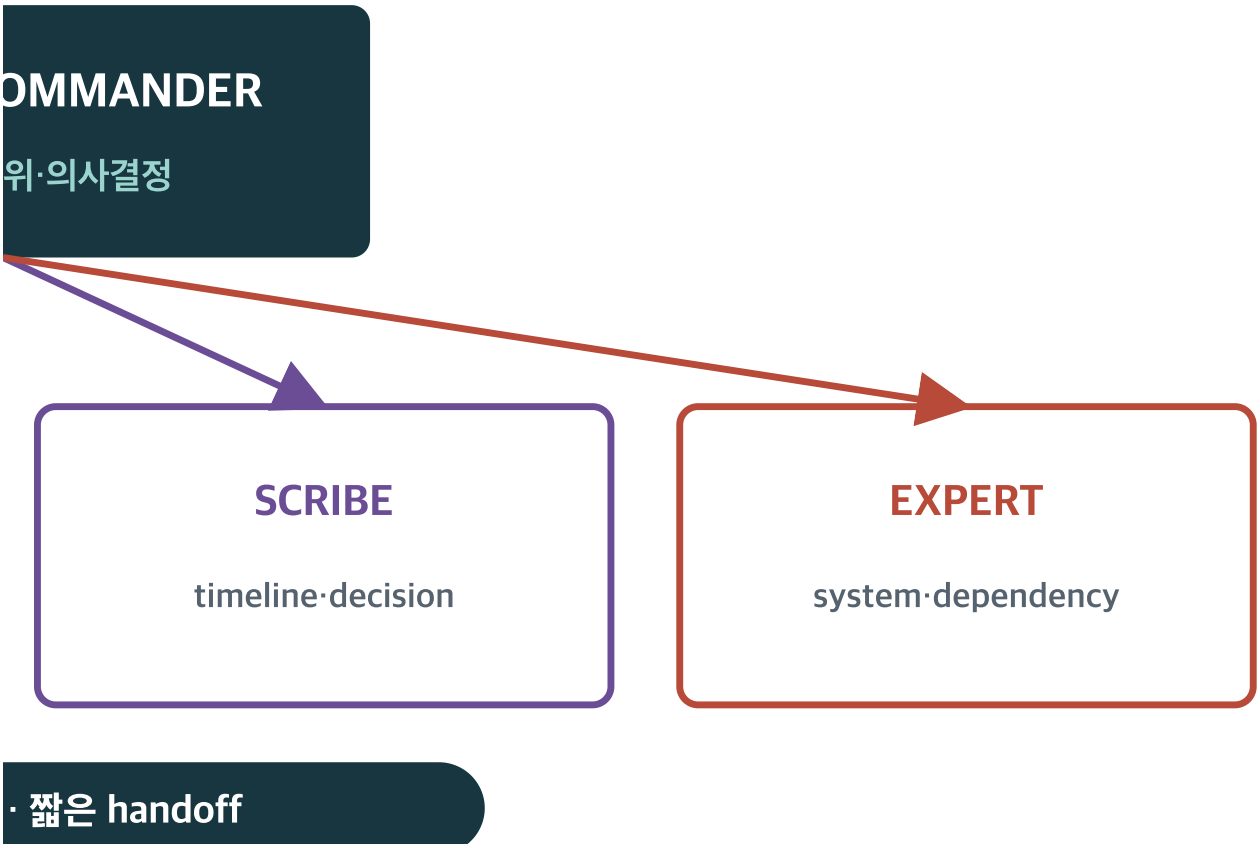
그림 13. Incident Commander가 목표와 우선순위를 정하고 operations·communications·scribe·expert가 병렬로 움직입니다. 한 사람이 모든 역할을 쥐면 판단·실행·기록이 서로 방해합니다.

장애 대응은 혼돈 대신 역할 분담으로

Incident Commander가 목표를 정하고 operations·communications·scri



be·expert가 병렬로 움직입니다.



· 짧은 handoff

15.1 언제 Incident를 선언하는가

다음 trigger를 미리 정합니다.

- Critical journey fast burn
- 광범위한 availability 저하
- Data integrity·security 의심
- Recovery objective 위반 위험
- 여러 팀·provider 조정 필요
- 고객 communication 필요

선언이 늦으면 각 팀이 다른 방에서 다른 목표로 움직입니다.

15.2 Severity

Severity는 기술 error message의 크기가 아니라 사용자·업무·data·안전 영향과 복구 긴급성으로 정합니다.

질문	예
누가 영향을 받는가	일부 사용자 / 전체 사용자
어떤 journey인가	Optional / critical
Data·security 위험이 있는가	없음 / 의심 / 확인
우회가 있는가	검증된 우회 / 없음
얼마나 지속됐는가	분 / 시간

15.3 역할

역할	책임	집중해야 할 질문
Incident Commander	목표·우선순위·결정·escalation	지금 가장 중요한 결과는 무엇인가
Operations lead	완화·복구·검증	가장 안전하고 되돌릴 수 있는 행동은 무엇인가
Communications lead	사실 기반 status	확인된 영향과 다음 공지는 무엇인가
Scribe	Timeline·decision·action	언제 무엇을 보고 왜 결정했는가
Expert	System·dependency 분석	어떤 조건이 증상을 설명하는가

작은 팀은 한 사람이 두 역할을 할 수 있지만 역할 이름과 전환 시점을 명시합니다.

15.4 첫 15분

```
incident ID·start UTC·severity
→ IC·operations·comms·scribe
→ confirmed impact
→ recent change·release·dependency
→ traffic protection·stop condition
→ next update time
→ timeline·decision log
```

15.5 Handoff

교대 때 전달할 것:

- Current confirmed impact
- Active hypothesis와 반증 evidence
- 실행한 change와 결과
- 금지된 또는 위험한 action
- Next step·owner·deadline
- Next stakeholder update
- Current residual risk

---

16. Status와 Mitigation을 안전하게 운영하기

16.1 사실 기반 Status

나쁜 예:

```
아마 database bug 같습니다. 곧 고치겠습니다.
```

좋은 예:

```
09:00 UTC부터 일부 checkout 요청이 완료되지 않았습니다.
현재 traffic을 보호하고 known-good release 복구를 검증 중입니다.
다음 상태는 09:30 UTC에 공유합니다.
```

16.2 Status의 다섯 field

1. Confirmed impact
2. Affected scope·start time
3. Current action
4. Verified workaround가 있으면 안내
5. Next update time

추측·확정되지 않은 root cause·개인정보·token·내부 공격 detail을 넣지 않습니다.

16.3 Mitigation 후보

행동	장점	위험	Stop condition
Traffic 제한	추가 피해 감소	사용자 접근 감소	SLI·capacity
Known-good rollback	빠른 회복	Schema 비호환	Data·health
Feature disable	Critical path 보호	기능 축소	Minimum service
Dependency 우회	연쇄 장애 완화	Data consistency	Validation
Restore	Data 회복	오점 선택·덮어쓰기	Isolated verify

16.4 Controlled Change

Incident라고 change control이 사라지지 않습니다. 더 짧아질 뿐입니다.

```
hypothesis
→ one change
→ owner·approver
→ expected signal
→ stop condition
→ execute
→ verify·record
```

동시에 여러 변경을 하면 무엇이 효과를 냈는지 알 수 없고 rollback도 어려워집니다.

16.5 Recovery Verify

- [ ] Public path가 정상이다.
- [ ] Critical journey SLI가 돌아왔다.
- [ ] Data integrity와 업무 결과가 정상이다.
- [ ] Known-good release·config가 확인됐다.
- [ ] 임시 access·silence·traffic rule에 owner·expiry가 있다.
- [ ] 일정 observation window를 지났다.
- [ ] Stakeholder status가 실제 상태와 일치한다.

---

17. Postmortem은 비난이 아니라 System 학습이다

17.1 기록할 것

항목	질문
Impact	누가 무엇을 얼마나 겪었는가
Timeline	언제 탐지·선언·완화·복구했는가
Trigger	무엇이 잠재 조건을 드러냈는가
Root cause	어떤 system 조건이 핵심이었는가
Contributing factors	무엇이 확대·지연시켰는가
Detection gap	왜 더 일찍 몰랐는가
Response gap	역할·runbook·권한이 왜 늦었는가
Recovery gap	RPO·RTO·business validation은 어땠는가
Worked well	어떤 통제가 피해를 줄였는가
Action	무엇을 누가 언제 어떻게 검증할까

17.2 Blameless의 의미

Blameless는 책임이 없다는 뜻이 아닙니다. 당시 정보·tool·pressure에서 왜 그 판단이 합리적으로 보였는지 이해하고, 개인의 주의력보다 system이 더 안전한 선택을 만들도록 바꾸는 것입니다.

나쁜 action:

- 더 주의한다.
- 다시는 실수하지 않는다.
- 교육한다.

좋은 action:

- Production target mismatch를 preflight에서 block한다.
- Missing release label을 contract test로 실패시킨다.
- Restore drill을 분기마다 실행하고 RTO를 자동 기록한다.
- Alert에 owner·runbook이 없으면 merge를 막는다.

17.3 Action 완료 조건

Field	예
Action	Query canary 추가
Risk reduction	Monitoring blind spot 탐지
Owner	Observability owner
Due	YYYY-MM-DD
Verification	Collector 중단 합성 test
Revalidation	SC-06 PASS

Google SRE Workbook의 postmortem culture는 학습과 조직적 개선을 강조합니다. 실제 조직은 legal·security·HR 요구와 함께 운영해야 합니다.

---

18. Operations Readiness Gate

M11-03 · 14

Operations Gate는 네 영역의 증거를 동시에 요구합니다

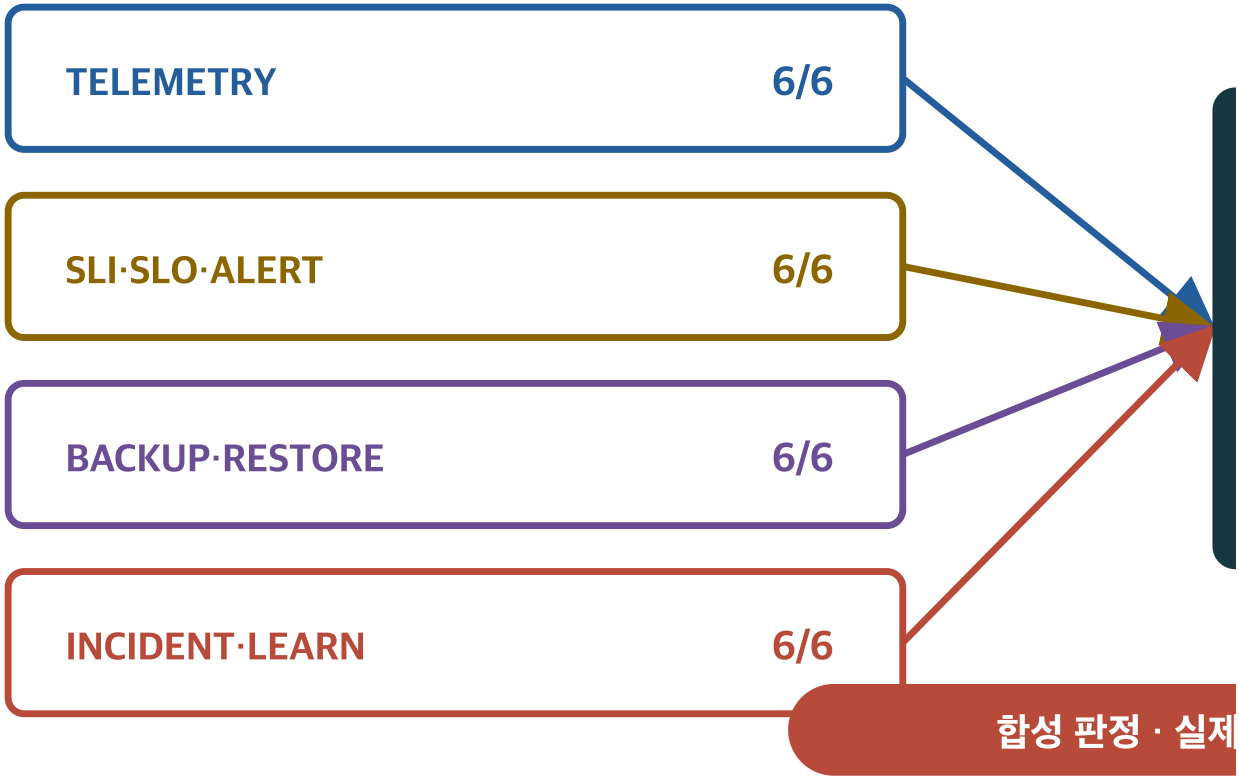
전체·critical·lane·control coverage가 100%이고 실제 side effect가 0일 때만 합성 후보를 수락합니다.

복원 evidence나 action owner 하나라도 없으면 준비 완료가 아니라 residual risk와 재검 계획을 남깁니다.

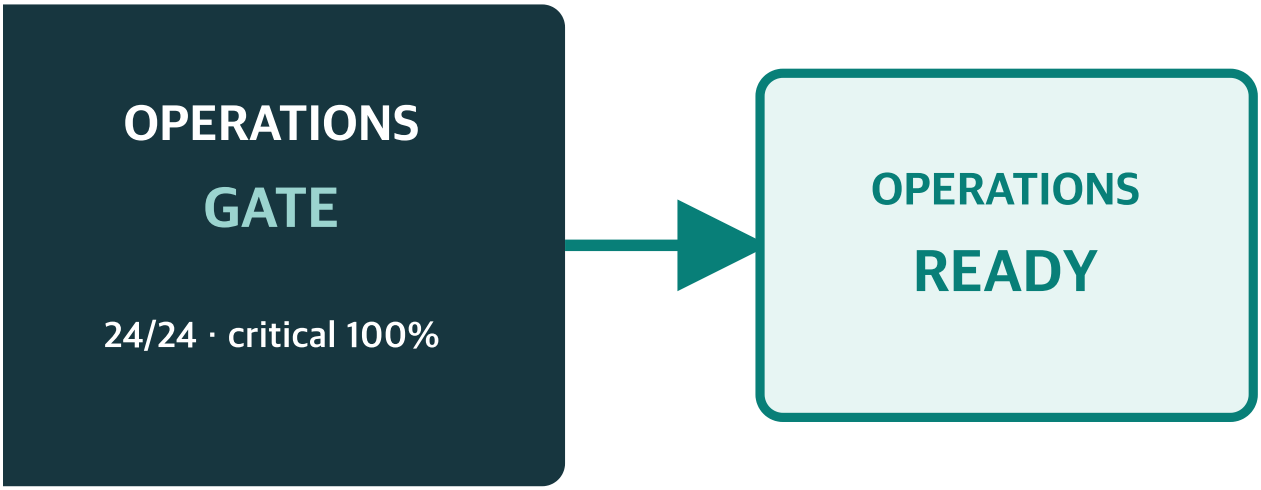
그림 14. Telemetry·SLI/SLO/Alert·Backup/Restore·Incident/Learn 네 lane가 각각 6/6이고 critical·control·safety gate가 모두 통과할 때만 합성 `operations_ready`를 판정합니다.

Operation Gate로 배포 가능 여부 판단

전체·critical·lane·control coverage가 100%이고 실제 side effect가 0일



때만 합성 후보를 수락합니다.



운영 승인 아님

18.1 네 Lane

Lane	Scenario	핵심 결과
Telemetry 계약	SC-01~06	Scope·schema·context·data boundary·pipeline health
SLI·SLO·Alert	SC-07~12	User SLI·budget·dashboard·route·black-box
Backup·Restore	SC-13~18	Inventory·protection·RPO·RTO·restore evidence
Incident·학습	SC-19~24	Declaration·role·status·mitigation·postmortem

18.2 정량 Gate

```
scenario pass rate = 100%
critical pass rate = 100%
lane coverage = 100%
control coverage = 100%
orphan controls = 0
orphan scenarios = 0
real side effects = 0
```

18.3 Safety Gate

- 실제 개인정보·secret 0
- 외부 network·cloud account 0
- Production telemetry·resource access 0
- 실제 alert·notification 0
- 실제 backup write·restore 0
- 파괴적 recovery 0
- Live side effect·cost 0
- Raw payload 0
- 보안·운영 인증 주장 0

18.4 `operations_ready`의 정확한 의미

정의한 합성 계약에서 24개 scenario와 모든 gate가 통과해 실제 운영 검토를 시작할 후보가 됐다는 뜻입니다.

다음을 뜻하지 않습니다.

- 실제 production 승인
- Security certification
- Disaster recovery certification
- 무장애 보증
- Provider SLA 보증
- 실제 backup 복원 완료

---

19. 운영 준비 검수 스튜디오

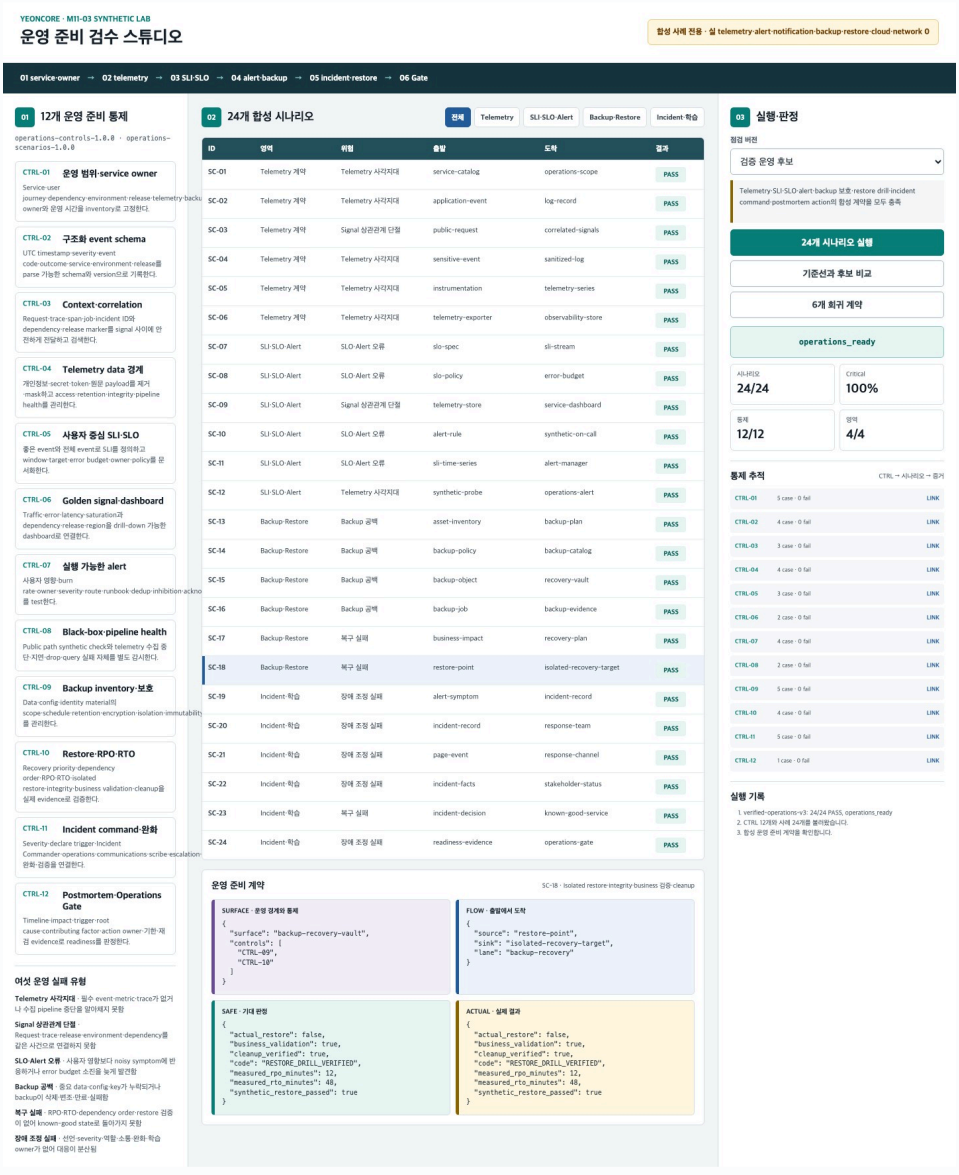


그림 15. 왼쪽 12개 control, 가운데 24개 scenario와 expected·actual, 오른쪽 variant·decision·coverage를 한 화면에서 연결합니다.

19.1 생성

```
./02_Labs/G11_Deployment_Operations/L11-03_create-operations-readiness-practice.sh
```

생성기는 다음을 자동 실행합니다.

Evidence	결과
Unit·contract·frontend·server test	333개 PASS
Contract audit	57/57 PASS
Regression contract	6/6 PASS
Safety boundary	Real·live side effect 0
Runtime	Python 표준 라이브러리 only

19.2 실행

```
cd gibalja-operations-readiness-practice/app
python3 app.py --host 127.0.0.1 --port 4395
```

Browser:

```
http://127.0.0.1:4395
```

19.3 화면 읽기

왼쪽: Control을 눌러 연결 scenario만 봅니다.

가운데: Scenario의 source·sink·risk·result를 보고 행을 선택해 expected와 actual을 비교합니다.

오른쪽: Variant를 실행해 pass·critical·control·lane coverage와 decision을 봅니다.

19.4 Mobile

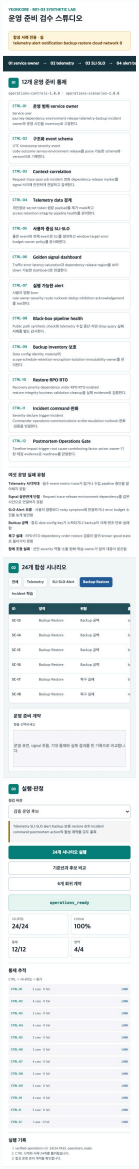


그림 16. 좁은 화면에서는 control·scenario·execution이 위에서 아래로 이어집니다. Lane filter로 6개씩 집중 학습할 수 있습니다.

---

20. 세 Version으로 배우기

20.1 Dashboard만 있음 · 6/24

이 version에는 inventory·SLI·backup scope·recovery objective·incident declaration·role의 골격만 있습니다.

대표 실패:

- Event schema 없음

- Signal correlation 없음
- Secret·개인정보 노출
- Metric cardinality·sampling 부재
- Telemetry pipeline blind
- SLO budget policy 없음
- Alert route·runbook 없음
- Backup protection·restore evidence 없음
- Incident handoff·safe status·mitigation·postmortem 미검증

Decision:

```
blocked_operational_blindness
```

20.2 Signal은 있으나 복구 미검증 · 16/24

Structured telemetry·SLO·dashboard·backup schedule·incident workflow가 생겼습니다. 그러나 다음 8건이 남습니다.

ID	결함
SC-04	Telemetry sensitive data boundary
SC-06	Pipeline health
SC-10	Alert route test
SC-12	Black-box·heartbeat
SC-15	Backup protection
SC-18	Restore drill
SC-22	Safe status
SC-24	Full Gate

Decision:

```
blocked_recovery_readiness
```

20.3 검증 운영 후보 · 24/24

모든 scenario의 expected와 actual이 일치합니다.

```
operations_ready
24/24 PASS
critical 100%
control 12/12
lane 4/4
```

20.4 비교

```
dashboard-only-v1
→ verified-operations-v3
= fixed 18
+ regressed 0
= accept_candidate
```

21. 12개 Control

ID	Control	최소 Evidence
CTRL-01	운영 범위·service owner	Service·journey·dependency·owner·coverage
CTRL-02	구조화 event schema	UTC·severity·code·outcome·release
CTRL-03	Context·correlation	Request·trace·span·dependency·release link
CTRL-04	Telemetry data 경계	Secret·개인정보·raw payload 0·access·retention
CTRL-05	사용자 중심 SLI·SLO	Good·total·window·target·budget·policy
CTRL-06	Golden signal·dashboard	Traffic·error·latency·saturation·drill-down
CTRL-07	실행 가능한 alert	Owner·route·runbook·ack·noise control
CTRL-08	Black-box·pipeline health	Public synthetic·heartbeat·drop·lag·canary
CTRL-09	Backup inventory·보호	Scope·schedule·retention·encryption·isolation

ID	Control	최소 Evidence
CTRL-10	Restore·RPO·RTO	Dependency order·isolated restore·validation
CTRL-11	Incident command·완화	Severity·role·timeline·status·mitigation
CTRL-12	Postmortem·Operations Gate	Action owner·due·verification·residual risk

21.1 Control과 Tool을 구분한다

“Log platform 도입”은 control이 아닙니다. Control은 위험·owner·statement·scenario·evidence가 있어야 합니다.

```
위험: collector 중단을 서비스 정상으로 오인
control: pipeline heartbeat와 query canary를 별도 감시
owner: observability owner
scenario: SC-06·SC-12
evidence: 합성 heartbeat 중단·alert·query result
```

22. 24개 Scenario

22.1 Telemetry 계약 · SC-01~06

ID	Scenario	Expected 핵심
SC-01	Service·journey·dependency·owner inventory	4 dependencies·owners·24x7
SC-02	UTC·severity·event code	Schema 1.0·parse 가능
SC-03	Request·trace·span·release	Cross-signal links 3
SC-04	Secret·개인정보·raw payload	Sensitive field 0
SC-05	Metric·cardinality·sampling	Unit·bounded label·histogram·policy
SC-06	Drop·lag·retention·access·tamper	Pipeline healthy

22.2 SLI·SLO·Alert · SC-07~12

ID	Scenario	Expected 핵심
SC-07	User journey good·total	Versioned exclusion·owner
SC-08	SLO·error budget	Target·28d·calculation·policy
SC-09	Golden signal dashboard	4 signal·release·dependency·region
SC-10	Alert route test	Owner·runbook·route·ack·real delivery 0
SC-11	Multi-window·noise control	Fast·slow·dedup·inhibit·expiry
SC-12	Public path·heartbeat	DNS·TLS·HTTP·business·canary

22.3 Backup·Restore · SC-13~18

ID	Scenario	Expected 핵심
SC-13	Critical asset inventory	Data·config·identity·owner
SC-14	RPO schedule·retention	Interval 10m < RPO 15m·alert
SC-15	Backup copy protection	Encrypted·isolated·immutable·separate identity
SC-16	Job·checksum·catalog·deletion	Integrity evidence
SC-17	Recovery objective·order	RPO 15m·RTO 60m·minimum service
SC-18	Isolated restore·validation	Synthetic pass·RPO 12m·RTO 48m·cleanup

22.4 Incident·학습 · SC-19~24

ID	Scenario	Expected 핵심
SC-19	Declare·severity·incident ID	Trigger·UTC start
SC-20	IC·operations·comms·scribe	Role·runbook
SC-21	Ack·escalation·handoff·timeline	Ack 120s·8 entries·decision
SC-22	Safe stakeholder status	Impact·30m·speculation 0·sensitive 0
SC-23	Mitigation·recovery verify	Traffic·known-good·public·data·controlled

ID	Scenario	Expected 핵심
SC-24	Postmortem·Operations Gate	24/24·critical 100%·action·risk

23. 실제 프로젝트 적용 순서

Phase 0 · Scope

산출물:

- One-sentence service outcome
- Critical journey·dependency graph
- Service·observability·on-call·backup·recovery·incident owner
- Production environment·region·release handoff
- Data classification·operation window

Gate:

```
unknown journey = BLOCK
unknown owner = BLOCK
unknown dependency = BLOCK
```

Phase 1 · Telemetry Contract

- Structured event schema
- Metric type·unit·bounded label
- Trace operation·context propagation
- Release·environment correlation
- Sensitive data allowlist
- Retention·access·integrity

Phase 2 · Reliability Contract

- Good·total SLI
- SLO target·window

- Error budget calculation
- Error budget policy
- Golden signal dashboard
- Release·dependency·region drill-down

Phase 3 · Alert·On-call

- Fast·slow burn
- Owner·route·runbook
- Dedup·grouping·inhibition
- Silence owner·expiry
- Synthetic route test
- Handoff·escalation

Phase 4 · Monitoring of Monitoring

- Public path synthetic
- Collector heartbeat
- Drop·lag·queue
- Query canary
- Separate failure path

Phase 5 · Backup·Recovery

- Critical asset inventory
- RPO-based schedule
- Retention·catalog·checksum
- Encryption·isolation·immutability·separate identity
- RPO·RTO·minimum service·dependency order
- Isolated restore·integrity·business validation·cleanup

Phase 6 · Incident·Learning

- Severity·declare trigger
- IC·operations·communications·scribe
- Timeline·decision·status cadence
- Controlled mitigation·recovery verify

- Blameless postmortem
- Action owner·due·verification

Phase 7 · Gate·Revalidation

다음 변화가 있으면 다시 검증합니다.

- Critical journey·SLO 변경
- New dependency·region·runtime
- Telemetry schema·backend·retention 변경
- Alert route·on-call team 변경
- Backup policy·key·vault 변경
- Incident 또는 restore drill 실패
- Major release·migration

---

24. 문제 해결 지도

증상	먼저 볼 경계	확인	위험한 즉흥 조치
Graph가 모두 0	Pipeline health	Heartbeat·drop·lag·query	“문제 없음” 선언
Log 검색 불가	Schema·context	Service·env·release·event code	Free-text 더 늘림
Metric 비용 급증	Cardinality	Label value count	필요한 signal 전부 삭제
Trace가 끊김	Context propagation	Traceparent·sampling	User ID를 correlation으로 사용
Secret가 log에 보임	Data boundary	Allowlist·mask·export	저장소 record만 삭제
Alert가 너무 많음	Rule·noise	SLI·dedup·inhibition	모든 alert silence
Alert가 늦음	SLO·burn	Window·route·ack	Threshold 무조건 낮춤
특정 release만 오류	Correlation	Release marker·label	전체 평균만 확인
Monitoring이 조용함	Monitoring-of-monitoring	Collector·canary	Collector만 restart
Backup job 실패	Backup pipeline	Source·quota·catalog·alert	Retention 전체 삭제

증상	먼저 볼 경계	확인	위험한 즉흥 조치
Backup 성공, 복원 실패	Restore	Key·format·dependency·validation	Production에 즉시 restore
RTO 초과	Recovery plan	순서·automation·권한	목표 숫자만 늘림
Restore data 불일치	Integrity·business	Recovery point·schema·relation	일부 record 수동 수정
Incident 선언 늦음	Trigger·severity	SLI·impact·authority	개인 판단에 맡김
대응 channel 혼란	Role·handoff	IC·scribe·decision log	모두가 모든 작업 수행
Status가 자주 바뀜	Fact boundary	Confirmed impact·cadence	Root cause 조기 단정
Rollback 후 error 감소	Recovery verify	Public path·data·SLI	즉시 incident 종료
Action이 오래 미완료	Postmortem governance	Owner·due·verification	문서 close
Dashboard가 너무 큼	질문·drill-down	Overview vs diagnostic	Graph만 더 추가
비용 신호가 이상	M11-04 handoff	Volume·retention·license	Reliability control 무작정 제거

24.1 문제 해결 순서

```
사용자 영향 확인
→ signal과 pipeline health 분리
→ environment·release·dependency context 연결
→ incident trigger·role 판단
→ one controlled mitigation
→ public path·data·SLI 검증
→ timeline·evidence 기록
→ action·재검
```

25. 공식 근거와 현재성

25.1 OpenTelemetry

- OpenTelemetry Signals
- OpenTelemetry Semantic Conventions Concepts
- OpenTelemetry Semantic Conventions 1.43.0

- OpenTelemetry Context Propagation

Semantic convention과 SDK·collector version은 변합니다. 실제 구현 전 현재 stable status와 migration note를 확인합니다.

25.2 Google SRE Workbook

- Monitoring
- Implementing SLOs
- Example Error Budget Policy
- On-call
- Incident Response
- Postmortem Culture

SRE 자료의 목표와 policy 예시는 우리 업무 계약·조직 권한·risk에 맞춰 조정합니다.

25.3 Logging·Alerting

- OWASP Logging Cheat Sheet
- Prometheus Recording and Alerting Rules
- Prometheus Alertmanager Configuration

특정 제품 configuration을 그대로 옮기기보다 event·SLI·route·owner·evidence 계약을 먼저 만듭니다.

25.4 Incident·Recovery·Continuity

- NIST SP 800-61 Rev. 3 · Incident Response Recommendations and Considerations for Cybersecurity Risk Management
- NIST SP 800-184 · Guide for Cybersecurity Event Recovery
- NIST SP 800-34 Rev. 1 · Contingency Planning Guide
- CISA StopRansomware Guide
- CISA Cross-Sector Cybersecurity Performance Goals

NIST SP 800-61 Rev. 3은 2025년 4월 final입니다. 실제 incident·recovery plan은 최신 법규·산업 규정·조직 policy와 함께 검토합니다.

25.5 출처를 읽는 열 가지 질문

1. 이 문서의 update date와 version은 무엇인가.
2. 개념·recommendation·required control 중 무엇인가.

3. 우리 service·data·region·industry에 적용되는가.
4. 사용자 중심 SLI가 실제로 측정 가능한가.
5. Alert가 어떤 owner와 action으로 이어지는가.
6. Log가 금지해야 할 data는 무엇인가.
7. Backup 보호 기능은 어떤 mode·plan·region에 있는가.
8. Restore test가 실제 data와 production에 어떤 영향을 주는가.
9. Incident role과 communication에 법적 요구가 있는가.
10. 다음 review trigger와 owner는 누구인가.

---

26. 셀프 테스트 30

Q01. Dashboard가 있어도 observability가 부족할 수 있는 이유는 무엇인가.

모범 답안
Dashboard는 이미 수집되고 query 가능한 signal을 시각화합니다. 필요한 event가 계측되지 않았거나 공통 context가 없거나 collector·query pipeline이 멈추면 보기 좋은 화면이 있어도 사용자 증상과 원인·release를 연결할 수 없습니다.

Q02. Log·metric·trace가 각각 가장 잘 답하는 질문은 무엇인가.

모범 답안
Log는 정확히 어떤 사건이 일어났는지, metric은 시간에 따라 얼마나 변했는지, trace는 하나의 요청이 service와 dependency를 어떻게 통과했는지를 잘 보여 줍니다. Release event는 언제 무엇이 바뀌었는지 보완합니다.

Q03. Signal correlation에 필요한 공통 context 네 가지를 쓰라.

모범 답안
Service·environment·release ID·request 또는 trace ID가 핵심입니다. Incident·job·dependency ID도 흐름에 따라 추가할 수 있지만 metric label에는 request ID 같은 high-cardinality 값을 넣지 않습니다.

Q04. Structured log가 자유 문장 log보다 운영에 유리한 이유는 무엇인가.

모범 답안

Timestamp·severity·event code·outcome·service·release 같은 field와 type이 고정돼 문구가 바뀌어도 안정적으로 parse·집계·alert할 수 있습니다. Schema version으로 consumer 변경도 조정할 수 있습니다.

Q05. Severity와 outcome의 차이는 무엇인가.

모범 답안

Severity는 사건의 기술적 중요도나 긴급성이고 outcome은 operation의 성공·실패·불명 결과입니다. Retry warning이 최종 성공할 수 있고 HTTP 200이라도 업무 결과가 잘못되면 outcome과 user journey는 실패할 수 있습니다.

Q06. Secret·token·개인정보를 수집 뒤 삭제하는 것보다 보내기 전에 막아야 하는 이유는 무엇인가.

모범 답안

Telemetry는 collector·queue·backend·export·cache·backup에 복제될 수 있어 저장 뒤 삭제만으로 모든 사본을 통제하기 어렵습니다. Application·collector의 allowlist와 mask 경계에서 원문이 pipeline에 들어오지 않게 해야 합니다.

Q07. Log injection은 무엇이며 어떻게 줄이는가.

모범 답안

공격자가 newline·control character·가짜 field를 입력해 log line과 viewer 구조를 조작하는 공격입니다. 승인 field만 선택하고 type·길이를 제한하며 control character를 escape하고 원문 payload를 기록하지 않습니다.

Q08. Metric cardinality가 왜 중요한가.

모범 답안

각 고유 label 조합이 별도 time series를 만들어 user ID·request ID·raw URL 같은 거의 무한한 값은 storage·memory·query·비용을 급증시킵니다. Route template·status class·region처럼 bounded dimension을 사용합니다.

Q09. Head sampling과 tail sampling의 차이는 무엇인가.

모범 답안

Head sampling은 trace 시작 때 수집 여부를 결정해 단순하고 빠르지만 뒤에 나타날 error·latency를 모릅니다. Tail sampling은 trace 완료 후 결과를 보고 보존할 수 있지만 collector 자원과 지연·복잡성이 더 큽니다.

Q10. Monitoring of monitoring이 필요한 이유는 무엇인가.

모범 답안

Collector·exporter·storage·query가 멈추면 graph가 0이 돼 서비스가 정상처럼 보일 수 있습니다. Heartbeat·drop rate·ingestion lag·query canary로 “신호 없음”과 “문제 없음”을 구분합니다.

Q11. SLI·SLO·SLA의 차이는 무엇인가.

모범 답안

SLI는 실제 측정값, SLO는 정한 window 동안의 목표, SLA는 미달 때 책임·보상 등 계약상 결과를 포함할 수 있는 합의입니다. 모든 내부 SLO가 외부 SLA인 것은 아닙니다.

Q12. Good event와 total event를 사용자 journey에서 정의해야 하는 이유는 무엇인가.

모범 답안

Infrastructure 상태가 아니라 사용자가 목적을 달성했는지 측정하기 위해서입니다. Eligible total과 성공 조건·제외를 명시해야 분모를 바꿔 실패를 숨기지 않고 반복 계산할 수 있습니다.

Q13. SLO 99.9%의 error budget은 얼마이며, 왜 이 숫자가 보편 정답이 아닌가.

모범 답안

Error budget은 1 - 0.999 = 0.001, 즉 0.1%입니다. 그러나 실제 target은 사용자 기대·업무 영향·측정 정확도·architecture·팀 복구 능력·투자 trade-off로 합의해야 합니다.

Q14. Burn rate는 무엇인가.

모범 답안

허용된 error budget에 비해 실제 실패가 budget을 얼마나 빠르게 소진하는지 나타내는 비율입니다. Fast burn은 즉시 대응, slow burn은 지속 악화의 조기 개입에 사용합니다.

Q15. Golden signals 네 가지는 무엇인가.

모범 답안

Traffic·error·latency·saturation입니다. 각각 workload 양, 실패, 처리 지연, resource 한계 압력을 보고 release·dependency·region context와 함께 분석합니다.

Q16. Actionable alert의 최소 요소를 쓰라.

모범 답안

사용자 영향·signal rule·severity·owner·route·runbook·acknowledgement 목표·escalation·recovery verification이 필요합니다. 즉시 행동이 없다면 page보다 dashboard나 ticket이 적합할 수 있습니다.

Q17. Deduplication·grouping·inhibition·silence의 차이는 무엇인가.

모범 답안

Dedup은 같은 alert 반복을 하나로 만들고 grouping은 관련 alert를 한 notification에 묶습니다. Inhibition은 상위 장애가 알려졌을 때 파생 notification을 억제하며 silence는 승인 조건에서 일시 중지합니다. Silence에는 owner·이유·만료가 필요합니다.

Q18. Route test에서 실제 담당자에게 notification을 보내지 않아도 되는 이유는 무엇인가.

모범 답안

합성 channel과 acknowledgement simulator로 rule·label·route·template·escalation logic을 검증할 수 있기 때문입니다. 학습 실습의 안전 경계에서는 실제 사람 호출과 실제 notification을 0으로 유지합니다.

Q19. Backup job success가 recovery readiness를 증명하지 못하는 이유는 무엇인가.

모범 답안

Job 성공은 copy 작업 상태일 뿐 critical asset coverage·catalog·checksum·retention·encryption·isolation·key recoverability·restore·business validation을 보장하지 않습니다. 격리 복원 evidence가 있어야 복구 가능성을 주장할 수 있습니다.

Q20. Backup inventory에 database 외 무엇을 포함해야 하는가.

모범 답안

Configuration·identity material·object·file·queue 또는 event recovery 전략·backup catalog·runbook 등이 필요할 수 있습니다. Service를 재현하고 접근·복호화·업무 검증하는 데 필요한 모든 critical asset을 위험 기반으로 식별합니다.

Q21. Encryption·isolation·immutability가 각각 막는 위험은 무엇인가.

모범 답안

Encryption은 storage·전송 노출을 줄이고 isolation은 production account·network·identity 침해가 backup으로 번지는 위험을 줄입니다. Immutability는 정한 기간 무단 변경·삭제를 어렵게 합니다. 어느 하나도 오염된 data나 restore 실패를 자동 해결하지 않습니다.

Q22. Separate backup identity가 필요한 이유는 무엇인가.

모범 답안

Production 관리자 credential 침해나 오작업이 원본과 backup을 함께 삭제·변조하지 못하도록 failure domain과 권한을 나누기 위해서입니다. 별도 role·approval·audit·key recovery를 설계합니다.

Q23. RPO와 RTO의 차이를 한 문장씩 설명하라.

모범 답안

RPO는 장애 때 허용할 수 있는 최대 data 손실 시간이고 RTO는 장애 뒤 최소 또는 목표 service를 복구해야 하는 최대 시간입니다. Backup 빈도와 복구 architecture·순서·인력 요구가 각각 달라집니다.

Q24. Restore drill의 다섯 단계는 무엇인가.

모범 답안

Known-good recovery point 선택, production과 격리된 target 준비, dependency 순서에 따른 restore, integrity와 business validation, temporary access·data·resource cleanup입니다. Measured RPO·RTO와 gap action을 기록합니다.

Q25. Integrity validation과 business validation의 차이는 무엇인가.

모범 답안

Integrity validation은 checksum·schema·constraint·record 관계·file consistency를 확인합니다. Business validation은 복원 data로 핵심 user journey와 업무 규칙이 올바른 결과를 내는지 확인합니다. 둘 다 필요합니다.

Q26. Incident Commander와 operations lead를 왜 나누는가.

모범 답안

Incident Commander는 전체 목표·우선순위·role·decision·escalation을 조정하고 operations lead는 기술적 완화·복구·검증을 실행합니다. 한 사람이 모두 하면 세부 작업이 전체 판단과 소통을 방해할 수 있습니다.

Q27. 안전한 incident status에 포함하고 제외할 것을 쓰라.

모범 답안

확인된 사용자 영향·범위·시작 시간·현재 조치·검증된 우회·다음 공지 시간을 포함합니다. 추측·확정되지 않은 root cause·개인정보·token·불필요한 security detail은 제외합니다.

Q28. Incident recovery를 error graph 하락만으로 종료하면 안 되는 이유는 무엇인가.

모범 답안

Monitoring 오류나 traffic 감소로 graph가 내려갈 수 있고 data·업무 결과가 여전히 손상됐을 수 있습니다. Public path·critical SLI·known-good release·data integrity·controlled change·stakeholder status를 함께 검증합니다.

Q29. Blameless postmortem의 의미는 무엇인가.

모범 답안

책임을 없애는 것이 아니라 당시 정보와 system 조건에서 왜 판단이 합리적으로 보였는지 이해하고 개인 주의력보다 더 안전한 tool·control·process를 만드는 방식입니다. Action에는 owner·due·verification이 필요합니다.

Q30. `operations_ready`의 정확한 의미와 한계를 설명하라.

모범 답안

정의한 합성 범위에서 24개 scenario·critical·12 control·4 lane·safety gate가 모두 통과해 실제 운영 검토 후보가 됐다는 판정입니다. 실제 production 승인·보안 인증·재해 복구 인증·무장애 보증·실제 restore 완료를 뜻하지 않습니다.

27. 마지막 한 장 요약

운영 순환

```
service·journey·owner
→ structured log·metric·trace·release context
→ user SLI·SLO·error budget
→ golden signal·actionable alert
→ incident declare·role·mitigation
→ backup·known-good·restore·validation
→ postmortem·action·revalidation
```

운영 전 여섯 질문

1. **무엇을 운영하는가:** Critical journey·dependency·owner가 있는가.
2. **무엇을 보는가:** Signal schema·context·pipeline health가 있는가.
3. **언제 행동하는가:** SLO·burn·route·runbook이 있는가.
4. **무엇으로 복구하는가:** Protected backup·key·known-good point가 있는가.
5. **얼마나 빨리 복구하는가:** RPO·RTO·minimum service·order가 검증됐는가.
6. **무엇을 바꾸는가:** Postmortem action·owner·due·verification이 있는가.

Gate

```
Telemetry 6/6
+ SLI·SLO·Alert 6/6
+ Backup·Restore 6/6
+ Incident·Learn 6/6
+ Critical 100%
+ Control 12/12
+ Real·live action 0
= operations_ready
```

실제 운영 전 추가 확인

- 실제 provider observability·backup 기능과 현재 version·region·plan
- 실제 production data classification·privacy·retention·access review

- 실제 SLO business approval·SLA·support contract
- 실제 on-call schedule·연락·훈련·노동 정책
- 실제 backup·restore drill·key recovery·cleanup evidence
- 실제 incident legal·security·communications process
- 실제 capacity·load·cost·license·TCO
- 실제 change management·approval·stakeholder communication

다음 매뉴얼

M11-04에서는 M11-03의 telemetry volume·retention·active series·trace sampling·backup storage·restore compute·on-call tool seat를 입력으로 받아 **클라우드·AI 비용과 라이선스**를 계산합니다. Reliability control을 비용 때문에 무작정 제거하지 않고, 사용자 영향과 evidence를 유지하며 비용 구조를 비교합니다.