

K8s 클러스터 분리와 네임스페이스 분리 비교

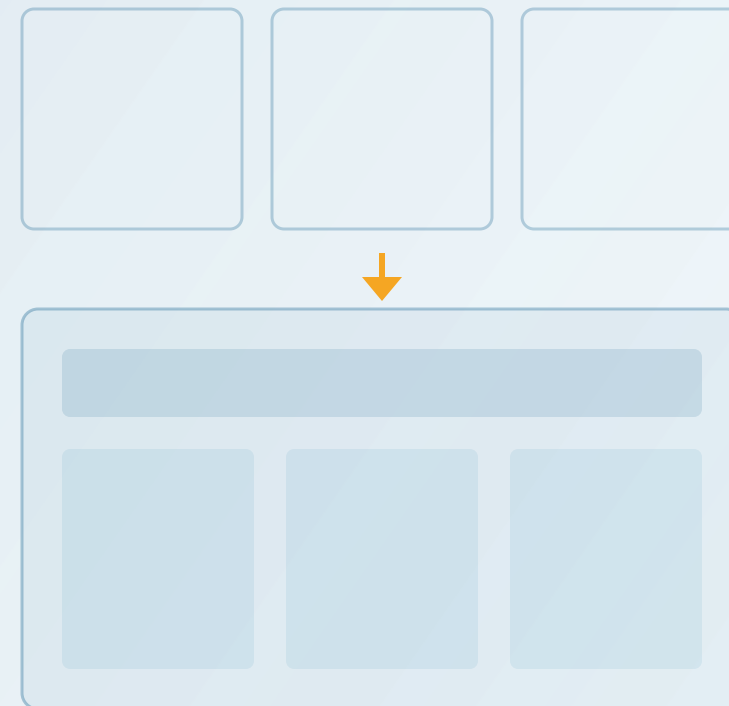
장단점과 선택 기준 — 비용·운영·개발 관점

PROJECT CODE

K8s 업무 분리

DATE

2026.10.16



Agenda

01

업무 시스템 분리 구성

업무별 마스터와 공유 마스터의 구조 차이

1장

02

비용·운영·개발 관점 비교

세 측면의 장단점과 종합 평가

4장

03

공유 마스터 클러스터의 노드 운영

워커 분리 방법, 격리 장치, 관리 한도와 사양

4장

04

분리 방식 결정 조건

택하는 이유, 필수 기준, 판단 흐름, 운영 모델

5장

KEY FOCUS

업무 분리 방식은 비용·운영·개발 비교와 클러스터 필수 기준으로 결정됨

SECTION 01

업무 시스템 분리 구성

K8s 업무 시스템을 클러스터 단위와 네임스페이스 단위로 나누는 구성 비교



KEY INSIGHT

신뢰 경계가 고객 사이라면 노드 분리 이상으로 격리 수준을 올려야 함

격리 약함 · 비용 낮음

격리 강함 · 비용 높음

01

네임스페이스로 나누기

이름 공간과 권한만 분리

커널과 노드를 그대로 공유함
신뢰하는 팀 사이 정리에는
충분하지만 보안 경계는 아님

격리 강도



02

노드로 나누기

커널까지 분리

테인트와 노드 셀렉터로
워크로드를 다른 노드에 모음
노드가 돌아도 나눠 쓸 수
없는 비용이 붙음

격리 강도



03

샌드박스 런타임

컨테이너마다 경계를 더 둠

가상화 계층을 한 겹 없어
커널 공유를 줄임
성능과 호환성을 함께
따져야 함

격리 강도



04

클러스터로 나누기

가장 확실하고 가장 비쌈

컨트롤 플레인까지 분리됨
클러스터 수만큼 운영
부담이 그대로 곱해짐

격리 강도



고르는 기준은 신뢰 경계

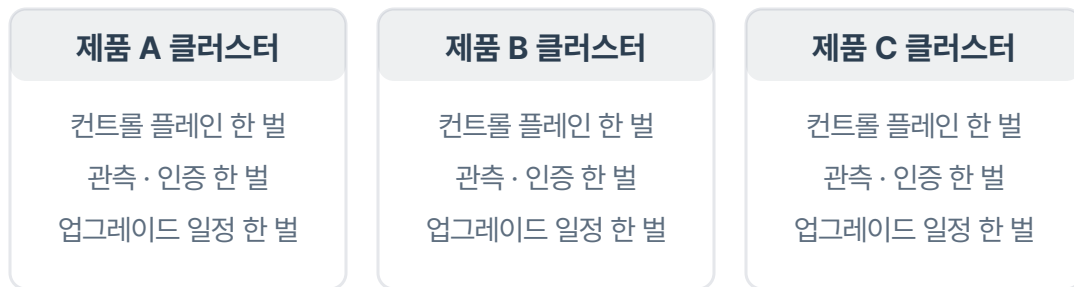
- 경계가 우리 팀 사이 — 네임스페이스로 충분함
- 경계가 서로 다른 고객 사이 — 노드 분리 아래로는 내려가지 않는 편이 안전함

Source: kubernetes.io — Multi-tenancy



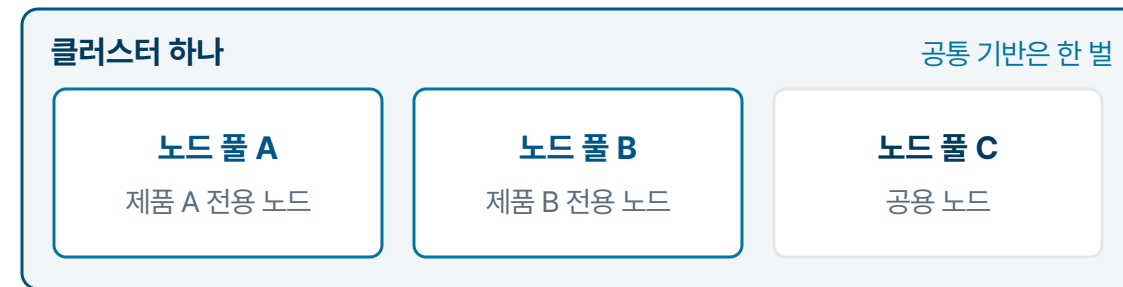
KEY INSIGHT 노드 분리로 시작 — 등급·규제·신뢰 경계가 다른 셋에 해당할 때만 클러스터를 나눔

제품마다 클러스터를 나눌 때



- 공통 기반이 제품 수만큼 복제됨
- 관리 비용이 곱하기로 늘어남
- 제품이 늘 때마다 클러스터 관리 팀의 일이 함께 늘어남

클러스터는 하나로 두고 노드를 나눌 때



- 노드 선택 규칙과 테인트로 배치를 강제함
- 자원 다툼과 배치 격리 요구는 여기서 대부분 해결됨

그래도 클러스터를 나눠야 하는 경우

컨트롤 플레인 장애를 함께 겪으면 안 되는 등급 차이

규제로 인프라 자체를 갈라야 하는 요구

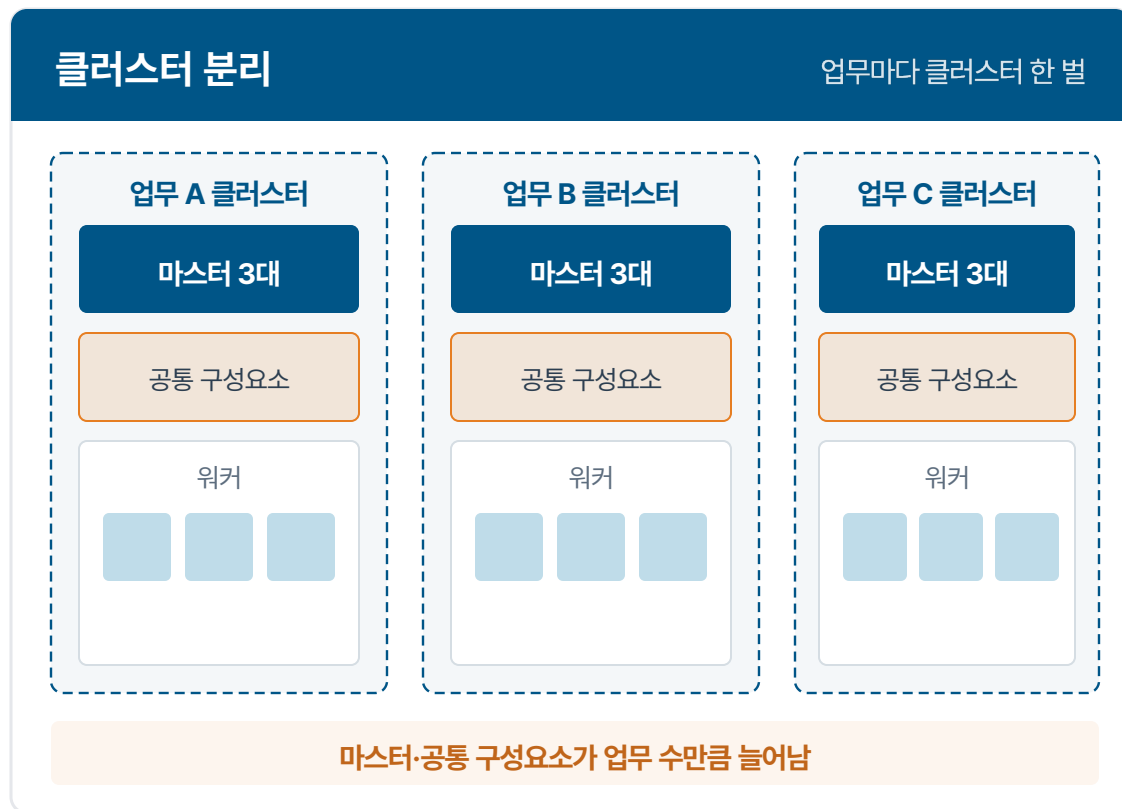
신뢰 경계가 다른 조직이 섞이는 상황

이 셋에 해당하지 않으면 노드 분리로 시작하고, 부족해질 때 클러스터를 나눔

Source: Kubernetes operations design guide, Japan Container Days (2018)

KEY INSIGHT

클러스터를 나누면 컨트롤 플레인과 공통 구성요소도 그 수만큼 늘어남 — 비용·운영 부담 차이가 여기서 생김



분리 단위

클러스터 분리는 물리 경계로 나눔 · 네임스페이스 분리는 정책 경계(권한 · 네트워크 · 자원 한도)로 나눔

SECTION 02

비용·운영·개발 관점 비교

클러스터 분리와 네임스페이스 분리를 세 관점에서 비교한 장단점

KEY INSIGHT

네임스페이스 분리는 마스터와 공통 구성요소를 한 벌로 줄임 — 클러스터 수만큼 늘던 고정 비용이 없어짐

클러스터 분리

업무마다 클러스터 한 벌

장점

- ✓ 업무별 비용이 클러스터 단위로 명확히 나뉨

단점

- ⚠ 클러스터마다 마스터 3대와 Ingress·모니터링·로깅 노드 중복
- ⚠ 클러스터 단위 구독·라이선스가 클러스터 수만큼 증가
- ⚠ 클러스터마다 여유 자원을 따로 남겨 전체 사용률 저하
- ⚠ 업그레이드·관제 인력이 클러스터 수에 비례해 증가

고정 비용

클러스터 수에 비례해 증가

네임스페이스 분리

마스터 한 벌을 공유

장점

- ✓ 마스터·공통 구성요소 한 벌로 고정 비용 절감
- ✓ 피크 시간이 다른 업무끼리 워커를 나눠 써서 빈 자원을 다른 업무가 재사용
- ✓ 구독·라이선스를 한 클러스터로 통합

단점

- ⚠ 공용 클러스터 용량을 처음부터 넉넉히 확보해야 함
- ⚠ 업무별 비용 배분은 라벨과 사용량 측정으로 따로 구성

고정 비용

업무 수와 무관하게 한 벌

KEY INSIGHT

네임스페이스 분리는 업그레이드·관제를 한 번에 처리함 — 대신 컨트롤 플레인 장애가 모든 업무에 미침

클러스터 분리

업무마다 클러스터 한 벌

장점

- ✓ 장애 영향 범위가 그 클러스터 안으로 한정됨
- ✓ 업무별로 업그레이드 일정을 따로 잡을 수 있음

단점

- ⚠ 업그레이드·보안 패치·인증서 갱신·etcd 백업을 클러스터 수만큼 반복
- ⚠ 관제·로그·정책이 흩어져 업무마다 운영 수준이 달라짐
- ⚠ 클러스터 간 통신과 인증을 따로 관리해야 함

버전 업그레이드 **클러스터마다 따로 반복**

네임스페이스 분리

마스터 한 벌을 공유

장점

- ✓ 업그레이드·보안 패치 1회로 전체 업무에 적용
- ✓ 관제·로그·추적을 한 화면에서 확인
- ✓ NetworkPolicy·Pod Security·Quota 를 한곳에서 자동 적용

단점

- ⚠ 컨트롤 플레인 장애·업그레이드 실패가 모든 업무에 영향
- ⚠ 업그레이드 전 입주 업무 전체의 호환성 확인 필요

버전 업그레이드 **한 번에 전체 업무 적용**

KEY INSIGHT

네임스페이스 분리는 새 개발 환경을 바로 제공함 — CRD·Operator 를 직접 설치해야 하는 제품은 제약을 받음

클러스터 분리

업무마다 클러스터 한 벌

장점

- ✓ 업무팀이 K8s 버전·CRD·Operator 를 직접 선택

단점

- ⚠ 개발·검증 환경마다 클러스터 구축을 기다려야 함
- ⚠ 클러스터 간 API 호출에 게이트웨이·인증 설정 추가
- ⚠ CI/CD 파이프라인이 클러스터마다 갈라짐

새 개발 환경 클러스터 구축·연동 후 제공

네임스페이스 분리

마스터 한 벌을 공유

장점

- ✓ 네임스페이스 생성만으로 새 환경 제공 (셀프서비스)
- ✓ CI/CD·GitOps·레지스트리를 하나로 통일
- ✓ 서비스 간 호출을 클러스터 내부 DNS 로 처리

단점

- ⚠ CRD·Operator·Admission Webhook 을 업무팀이 임의로 설치할 수 없음
- ⚠ K8s 버전·공통 구성요소 버전을 업무팀이 고를 수 없음

새 개발 환경 네임스페이스 생성 즉시 제공

KEY INSIGHT

비용·운영 효율은 네임스페이스 분리가 앞섬 — 장애 격리와 버전 자유도가 꼭 필요한 업무만 클러스터를 나누면 됨

측면	비교 항목	클러스터 분리	네임스페이스 분리
비용	고정 비용 (마스터·공통 구성요소)	△ 클러스터 수만큼 중복	◎ 한 벌
	자원 사용률	△ 클러스터별 여유 자원	◎ 워커 통합 사용
	업무별 비용 구분	◎ 클러스터 단위로 명확	○ 라벨·사용량 측정 필요
운영	업그레이드·패치 부담	△ 클러스터 수만큼 반복	◎ 1회
	관제·정책 일관성	△ 클러스터마다 편차	◎ 한곳에서 자동 적용
	장애 격리	◎ 클러스터 안으로 한정	△ 컨트롤 플레인 장애 공유
개발	환경 제공 속도	△ 클러스터 구축 대기	◎ 네임스페이스 생성
	버전·CRD 자유도	◎ 업무팀이 직접 선택	△ 공용 버전 사용

◎ 유리 ○ 보통 △ 불리

네임스페이스 분리 유리 5항목 · 클러스터 분리 유리 3항목

SECTION 03

공유 마스터 클러스터의 노드 운영

마스터를 함께 쓰는 클러스터에서 워커를 나누는 방법과 한 클러스터의 관리 한도

KEY INSIGHT

라벨과 Taint 를 함께 쓰면 업무별 전용 노드 풀이 됨 — 클러스터를 나누지 않고도 실행 노드를 분리할 수 있음



노드 라벨 + Node Affinity

업무 파드를 지정한 노드에 배치

- 파드가 갈 노드를 고르는 쪽
- 다른 파드의 진입은 막지 못함

설정 예

nodeSelector: team=biz-a

Taint · Toleration

전용 노드에 다른 업무 파드 진입 차단

- 노드가 파드를 거르는 쪽
- Toleration 이 있는 파드만 배치

설정 예

dedicated=biz-a:NoSchedule

전용 노드 풀

라벨 + Taint 조합으로 만든 노드 묶음

- 업무·보안 등급·GPU 별로 구성
- 실행 노드 단위로 업무를 분리

설정 예

pool=secure · pool=gpu

구성 예 — 마스터 한 벌 아래 워커 풀 셋



KEY INSIGHT

이름만 나누면 파드 간 통신과 자원 경쟁이 그대로 남음 — 생성 시점에 여섯 가지를 정책으로 함께 걸어야 업무 간 간섭이 막힘



ResourceQuota

자원

네임스페이스별 CPU·메모리·스토리지·
오브젝트 수 상한

적용 기준 **업무당 필수·증설은 신청 절차**



LimitRange

자원

컨테이너 request·limit 기본값과 상한

적용 기준 **request 없는 배포 차단**



PriorityClass

자원

자원 부족 시 우선 실행·선점 순서

적용 기준 **서비스 등급별 클래스 지정**



NetworkPolicy

보안

기본 차단 후 필요한 통신만 허용

적용 기준 **네임스페이스마다 Default Deny**



RBAC

보안

업무팀은 자기 네임스페이스 권한만 보유

적용 기준 **클러스터 범위 권한 금지**



Pod Security

보안

Admission 의 restricted 기준으로 특권
컨테이너·호스트 경로 접근 차단

적용 기준 **운영 네임스페이스 restricted**

KEY INSIGHT 공식 상한은 노드 5,000대임 — 실제 운영 한도는 마스터 사양과 etcd·API 서버 부하로 정해짐

쿠버네티스 공식 상한 (v1.37 문서 기준, 클러스터 하나)

5,000대

클러스터당 노드
워커 + 마스터 합계

110개

노드당 파드
kubelet 기본 상한

150,000개

클러스터 전체 파드
노드 수와 무관한 총량

300,000개

클러스터 전체 컨테이너
사이드카 포함

상한보다 먼저 닿는 병목 세 곳



etcd

오브젝트·이벤트가 늘면
저장 용량과 쓰기 지연 증가

대응

대규모에서는 Event 를 별도 etcd 로 분리



API 서버

노드·컨트롤러·업무 요청이
한곳으로 몰림

대응

먼저 마스터 사양을 키우고, 그다음 대수를 늘림



네트워크 · 애드온

파드 IP 대역·CNI 라우팅과
관제 애드온 수집량 증가

대응

애드온 리소스 한도를 규모에 맞춰 상향

KEY INSIGHT

252대까지 16 vCPU·64GB 로 운영됨 — 3대 중 1대가 내려가면 나머지 2대에 부하가 몰리므로 여유분을 더해 산정해야 함

마스터 1대당 사양 (마스터 3대 구성, 상용 배포판 시험값)

워커 노드	네임스페이스 밀도	마스터 CPU	마스터 메모리
24대	500	4 vCPU 	16 GB 
120대	1,000	8 vCPU 	32 GB 
252대	4,000	16 vCPU 	64 GB 

OVN-Kubernetes 네트워크를 쓰는 252대 구성은 24 vCPU · 128 GB 필요

네임스페이스 밀도: 시험에 쓴 네임스페이스 수 (Deployment·Service·Secret 등 오브젝트 포함)

! 사양 산정 시 주의

- **마스터 1대 장애 대비**
3대 중 1대가 중지·재시작되면
남은 2대의 CPU·메모리 사용량 급증
- **밀도가 사양을 좌우**
노드 수가 같아도 네임스페이스·오브젝트
수가 많으면 더 큰 사양 필요
- **운영 주기 부담**
수백 대 규모는 한 클러스터로 운영 가능,
그 이상은 업그레이드 시간이 먼저 부담

SECTION 04

분리 방식 결정 조건

네임스페이스 분리를 택하는 이유와 클러스터 분리가 필요한 조건

KEY INSIGHT

운영 인력과 비용이 클러스터 수에 비례해 늘기 때문임 — 표준을 한곳에서 적용하는 효과도 함께 얻음



운영 인력 효율

업그레이드·패치·인증서·
백업 작업을 한 번로 축소



비용 통합

마스터·공통 구성요소 중복
제거, 워커 자원 통합 사용



표준화

정책·배포·관제를 한곳에서
강제해 업무별 편차 제거



빠른 온보딩

네임스페이스·Quota·권한
배정만으로 새 업무 시작



연계 단순화

서비스 간 호출을 내부 통신과
공용 API 게이트웨이로 처리

클러스터마다 반복되는 작업

반복 작업	버전 업그레이드	인증서 갱신	etcd 백업·복구 시험	관제·로그 연동	정책 배포
클러스터 분리	클러스터 수만큼	클러스터 수만큼	클러스터 수만큼	클러스터 수만큼	클러스터 수만큼
네임스페이스 분리	1회	1회	1회	1회	1회

KEY INSIGHT

아래 여덟 가지 중 하나라도 해당하면 클러스터를 나눠야 함 — 네임스페이스 정책으로는 대신할 수 없는 경계이기 때문임

1

규제

망 분리·물리 분리 요구

법령·감독 규정상 인터넷망·업무망 분리 등이 요구되는 업무

2

배치

다른 망·데이터센터·DR 사이트

컨트롤 플레인도 지연이 짧은 한 장애 영역 묶음 안에 뒤야 함

3

장애

최상위 등급 서비스

공용 컨트롤 플레인 장애도 허용할 수 없는 업무

4

권한

클러스터 범위 권한 필수

자체 CRD·Operator·Webhook·커널 설정·특권 컨테이너가 필요한 제품

5

버전

K8s 버전·배포판 고정

공용 클러스터 업그레이드 일정을 따를 수 없는 상용 제품

6

규모

상한·사양 한계 근접

노드·파드 수가 공식 상한이나 마스터 사양 한계에 가까움

7

책임

운영 주체가 다름

외부 사업자가 클러스터 전체를 장기 운영·책임지는 계약

8

환경

운영·개발 환경 분리

업그레이드를 미리 검증하도록 운영 환경은 별도 클러스터로 둬

KEY INSIGHT

같은 보안 등급·같은 망에서 네임스페이스 권한만 쓰는 업무는 공용 클러스터로 충분함 — 성능 간섭은 Quota 와 전용 노드 풀로 해결됨



보안 등급

같은 보안 등급·같은 망

공용 클러스터와 같은 망·보안 등급에서 운영되는 업무

확인 질문 공용 클러스터와 같은 망·같은 보안 등급인가



권한

네임스페이스 범위 권한으로 구축

Deployment·Service·ConfigMap·Secret·Ingress 만으로 구성 가능

확인 질문 CRD·Operator 설치 없이 배포되는가



버전

공용 버전 사용 가능

공용 클러스터의 K8s 버전·공통 구성요소를 그대로 사용

확인 질문 공용 클러스터 업그레이드 일정을 따를 수 있는가



성능

간섭을 정책으로 해결

Quota·LimitRange·PriorityClass·전용 노드 풀로 충분한 수준

확인 질문 피크 부하를 Quota·전용 노드로 막을 수 있는가



운영

공용 관제·배포 체계 사용

공용 모니터링·로그·GitOps 배포 표준을 따를 수 있음

확인 질문 공용 관제·GitOps 표준을 따를 수 있는가



장애

공용 장애 영향 수용 가능

마스터 3대 HA 를 전제로 컨트롤 플레인 장애 영향을 받아들일 수 있는 등급

확인 질문 컨트롤 플레인 장애 시 함께 멈춰도 되는가

KEY INSIGHT 기본은 네임스페이스이고 조건에 해당할 때만 한 단계씩 올림 — 별도 클러스터는 예외 승인 대상으로 관리함



단계를 올릴 때 지키는 원칙

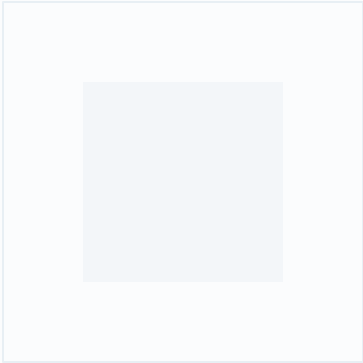
- 올림은 근거로만
판단 항목과 근거를 수용 방식 결정서로 남김
- 예외도 같은 표준
별도 클러스터도 공용 관제·로그·GitOps 표준 적용
- 주기적 재검토
사업 변경·규모 증가 시 단계를 다시 판단

3단계 비중은 작게 유지 — 별도 클러스터가 늘수록 고정 비용과 반복 작업이 다시 늘어남

KEY INSIGHT

공용 클러스터를 운영·개발 두 벌로 두고 업무는 네임스페이스로 수용함 — 예외 클러스터도 공용 관제·배포 표준을 따라야 운영이 흩어지지 않음





THANK YOU

감사합니다



K8s 업무 분리 — 네임스페이스 기본, 필수 기준에 해당할 때만 클러스터 분리

MSAP.ai

msap.ai

Web <https://www.msap.ai/>

Tel 02-6953-5427

Mail hello@msap.ai

