CNF

2026.03

# SearXNG 기술 백서:
# 기업 AI 시스템을 위한 엔터프라이즈 검색 인프라

SearXNG의 등장은 단순한 프로젝트의 계승이 아니라, 오픈소스 메타검색 엔진의 발전과정에서 중요한 전환점을 의미합니다. **SearX에서 SearXNG로 이어진 이 계보를 이해**하기 위해서는, SearX가 지향했던 프라이버시 중심의 설계 철학, 그리고 그 과정에서 드러난 기술적·운영적 한계와 이를 극복하기 위한 커뮤니티의 노력을 살펴볼 필요가 있습니다.

hello@cncf.co.kr

02-469-5426

www.cncf.co.kr

# Contents

# 1장. SearXNG 프로젝트의 시작과 배경

## 1.1 SearX → SearXNG로 이어진 계보("왜 fork가 필요했나?")

SearXNG의 등장은 단순한 프로젝트의 계승이 아니라, 오픈소스 메타검색 엔진의 발전과정에서 중요한 전환점을 의미합니다. SearX에서 SearXNG로 이어진 이 계보를 이해하기 위해서는, SearX가 지향했던 프라이버시 중심의 설계 철학, 그리고 그 과정에서 드러난 기술적·운영적 한계와 이를 극복하기 위한 커뮤니티의 노력을 살펴볼 필요가 있습니다. 특히, 왜 기존 프로젝트를 그대로 유지하지 않고 fork라는 방식을 선택했는지, 그리고 이러한 선택이 실무적으로 어떤 의미를 갖는지에 대한 이해가 중요합니다. SearXNG는 단순한 코드 분기가 아니라, 실질적인 서비스 안정성과 신뢰성 확보를 위한 실무적 대안으로 자리매김하게 되었습니다.

### 1.1.1 SearX의 목표와 한계(프라이버시 중심 메타검색의 출발점)

SearX는 2014년 Adam Tauber(asciimoo)에 의해 시작된 오픈소스 메타검색 엔진으로, 사용자 프라이버시 보호를 최우선 가치로 내세웠습니다. 기존의 상용 검색 엔진들은 검색 쿼리, IP 주소, 브라우저 정보 등 다양한 사용자 데이터를 수집하고 이를 기반으로 광고 타겟팅이나 사용자 프로파일링을 수행해 왔습니다. 이에 반해 SearX는 사용자의 검색 데이터를 저장하지 않으며, 검색 결과를 여러 엔진에서 취합하여 제공함으로써 단일 엔진에 의존하지 않는 구조를 채택하였습니다. 이로 인해 SearX는 개인 사용자뿐만 아니라, 기업과 기관에서도 내부망에서 직접 운영할 수 있는 self-hosting 솔루션으로 각광받았습니다.

기술적으로 SearX는 다양한 검색 엔진(예: Google, Bing, DuckDuckGo, Wikipedia 등)과의 커넥터를 제공하며, 사용자가 원하는 엔진을 선택하여 검색 결과를 집계할 수 있도록 설계되었습니다. 설정 파일을 통해 엔진 목록, 출력 포맷, 프라이버시 관련 옵션 등을 세밀하게 조정할 수 있었고, 플러그인 구조를 통해 기능 확장도 가능했습니다. 그러나 이러한 유연성에도 불구하고, SearX는 몇 가지 근본적인 한계에 직면했습니다.

**첫째, 메타검색 엔진의 특성상 외부 검색 엔진의 API나 웹 인터페이스 변화에 매우 민감합니다.** 예를 들어, Google이나 Bing 등 주요 엔진이 API 정책을 변경하거나, CAPTCHA, rate-limit, bot protection을 강화하면 SearX의 커넥터가 정상적으로 동작하지 않는 경우가 빈번하게 발생합

니다. 이러한 엔진 오류는 전체 검색 서비스의 품질 저하로 이어지며, 사용자는 검색 결과를 제대로 받지 못하는 상황을 겪게 됩니다.

**둘째, SearX 프로젝트의 유지보수는 소수의 개발자에 의존하는 구조**였고, 시간이 지남에 따라 주요 개발자의 관심이 줄어들면서 버그 수정이나 엔진 커넥터 업데이트가 지연되는 문제가 나타났습니다. 특히, 엔진 커넥터의 신속한 개선이 필요한 상황에서 커뮤니티의 대응 속도가 느려지면, 서비스 안정성에 심각한 영향을 미칠 수밖에 없습니다.

**셋째, SearX 저장소가 아카이브되면서 공식적인 유지보수와 지원이 중단되었습니다.** 이로 인해 기존 사용자들은 새로운 엔진 오류에 대응할 방법이 없어졌고, 기관이나 기업에서는 장기적인 서비스 운영에 대한 불확실성을 느끼게 되었습니다. 이러한 상황은 SearX를 기반으로 한 새로운 fork 프로젝트의 필요성을 촉진하는 계기가 되었습니다.

실무적으로 SearX를 도입했던 기관들은 self-hosting의 장점(프라이버시 보호, 내부망 운영, 데이터 경계 설정 등)을 높이 평가했으나, 엔진 커넥터의 잦은 장애와 느린 유지보수 속도, 그리고 공식 지원의 부재로 인해 운영 리스크가 커졌습니다. 특히, 검색 서비스가 AI 에이전트, 내부 정보 탐색, 감사 시스템 등과 연계되는 경우에는 검색 엔진의 신뢰성과 안정성이 더욱 중요해집니다.

이러한 한계점은 SearXNG의 등장 배경과 직결되며, SearXNG가 기존의 프라이버시 중심 메타검색 철학을 계승하면서도, 엔진 오류 수정과 개선 속도를 높이고, 서비스 안정성을 강화하는 방향으로 발전하게 된 핵심 동인으로 작용하였습니다. SearXNG는 단순한 기능 추가가 아니라, 메타검색 엔진의 실질적인 운영 리스크를 해소하기 위한 실무적 대안으로 자리잡게 되었습니다.

SearX의 한계를 좀 더 구체적으로 살펴보면, 예를 들어 Google이나 Bing의 검색 결과 페이지 구조가 변경될 때마다 SearX의 커넥터가 제대로 동작하지 않는 사례가 빈번하게 발생했습니다. 이러한 변화는 예고 없이 이뤄지는 경우가 많아, 실시간으로 대응하지 않으면 사용자 경험이 크게 저하될 수밖에 없습니다. 또한, CAPTCHA나 bot protection이 강화되면, 자동화된 검색 요청이 차단되어 결과를 받아올 수 없는 문제가 발생합니다. 이로 인해 SearX를 도입한 기관에서는 검색 서비스의 신뢰성이 떨어지고, 내부 사용자들의 불만이 커지는 상황이 반복되었습니다.

또한, SearX 프로젝트의 유지보수 체계가 소수의 핵심 개발자에 지나치게 의존한 점도 문제였습니다. 오픈소스 프로젝트 특성상 커뮤니티의 활발한 참여가 중요하지만, 실제로는 주요 개발자의 이탈이나 관심 저하로 인해 버그 수정이나 커넥터 업데이트가 지연되는 경우가 많았습니다. 이로 인해, 실무 현장에서는 엔진 장애가 발생해도 신속하게 대응할 수 없는 구조적 한계가 드러났습니다.

마지막으로, SearX 저장소가 아카이브되면서 공식적인 지원이 완전히 중단되었고, 이는 기존

사용자들에게 큰 불안 요소로 작용했습니다. 특히, 장기적으로 검색 서비스를 안정적으로 운영해야 하는 기관이나 기업 입장에서는, 더 이상 공식적인 버그 수정이나 보안 패치를 기대할 수 없다는 점이 치명적인 리스크로 인식되었습니다. 이러한 상황은 SearXNG와 같은 새로운 fork 프로젝트의 필요성을 더욱 부각시키는 계기가 되었습니다.

### 1.1.2 SearXNG는 누가, 언제, 왜 만들었나(핵심 요약)

SearXNG는 SearX 프로젝트의 유지보수 공백과 엔진 오류 수정/개선 속도의 한계를 극복하기 위해 2021년 중반에 fork된 오픈소스 메타검색 엔진입니다. 공식 위키에는 2021-03-27 릴리스가 최초로 기록되어 있으며, SearXNG의 개발은 SearX의 주요 유지보수 인력과 커뮤니티가 분기하여 시작되었습니다. SearX의 원 저자인 Adam Tauber(asciimoo)는 프로젝트 초기 프라이버시 중심 설계를 주도했으나, 시간이 지나면서 공식 저장소가 아카이브되고, 유지보수 공백이 발생하였습니다.

SearXNG의 개발자들은 SearX의 유지보수에 참여했던 경험을 바탕으로, 기존의 프라이버시 보호 철학을 계승하면서도, 엔진 커넥터의 신속한 개선과 더 빠른 upstream 통합, 엔진 신뢰성 강화에 초점을 맞추었습니다. 메타검색 엔진은 외부 검색 엔진의 API 변화, bot protection, rate limit 등 다양한 장애 요인에 노출되기 때문에, 유지보수 속도와 엔진 신뢰성이 서비스 안정성의 핵심이 됩니다. SearXNG는 이러한 실무적 요구를 반영하여, 커뮤니티 중심의 빠른 이슈 대응, 커넥터 업데이트, 플러그인 개선 등을 적극적으로 추진하였습니다.

프로젝트의 분기 과정에서는 기존 SearX 사용자들이 SearXNG로 자연스럽게 이동하는 흐름이 나타났으며, 특히 기관 및 기업에서 self-hosting 메타검색 솔루션을 필요로 하는 경우, SearXNG가 사실상 표준 대안으로 자리잡게 되었습니다. SearXNG는 기존 SearX의 설정 파일 구조, 엔진 연동 방식, 프라이버시 옵션 등을 계승하면서도, Docker, Kubernetes 등 최신 배포 환경에 맞춘 컨테이너화와 자동화 기능을 강화하였습니다.

실무적으로 SearXNG는 엔진 커넥터 장애 발생 시 신속한 패치 제공, 커뮤니티 기반의 이슈 트래킹, 그리고 다양한 검색 엔진과의 연동 테스트를 통해 서비스 안정성을 높이고 있습니다. 예를 들어, Google, Bing, DuckDuckGo 등 주요 엔진의 API 정책 변경에 대응하여 커넥터 업데이트를 빠르게 진행하고, CAPTCHA, rate-limit, bot protection 등 장애 요인에 대한 리미터 및 프록시 설정을 지원합니다.

한편, SearXNG의 등장으로 인해 메타검색 엔진을 도입하는 기관 및 기업은 유지보수 속도와 서비스 안정성에 대한 신뢰를 확보할 수 있게 되었으며, AI 에이전트, 내부 정보 검색, 감사 시스템 등 다양한 활용 시나리오에서 안정적인 검색 인프라를 구축할 수 있게 되었습니다. SearXNG는 단순한 fork가 아니라, 메타검색 엔진의 실무적 요구를 반영한 진화된 프로젝트로서, 프라이버시 보호와 서비스 안정성이라는 두 가지 핵심 가치를 동시에 실현하고 있습니다.

SearXNG의 개발 배경을 좀 더 구체적으로 살펴보면, SearX 프로젝트가 공식적으로 아카이브 된 이후에도 여전히 많은 기관과 개인 사용자들이 메타검색 엔진을 필요로 했다는 점이 있습니다. SearXNG는 이러한 사용자들의 요구에 부응하기 위해, 기존 코드베이스를 기반으로 하되, 유지보수 체계를 강화하고 커뮤니티의 참여를 확대하는 방향으로 발전하였습니다. 개발 초기에는 SearX의 주요 유지보수자들과 활발한 커뮤니티 기여자들이 중심이 되어 프로젝트를 이끌었으며, GitHub 이슈 트래킹과 PR(풀 리퀘스트) 관리가 신속하게 이루어졌습니다.

특히, SearXNG는 Docker 및 Kubernetes와 같은 현대적인 배포 환경을 적극적으로 지원함으로써, 기관이나 기업이 손쉽게 자체 인스턴스를 구축하고 운영할 수 있도록 하였습니다. 이는 기존 SearX에서 부족했던 부분으로, 실제로 많은 기관들이 SearXNG의 컨테이너화 기능을 활용하여 신속하게 서비스를 배포하고 있습니다. 또한, SearXNG는 플러그인 시스템과 커스터마이징 옵션을 강화하여, 각 조직의 요구에 맞는 맞춤형 검색 서비스를 구현할 수 있도록 지원합니다.

또한, SearXNG는 엔진 커넥터의 신속한 업데이트와 장애 대응을 위해, 커뮤니티 기반의 이슈 대응 프로세스를 체계화하였습니다. 예를 들어, Google이나 Bing의 API 정책이 변경될 경우, 커뮤니티 내에서 관련 이슈가 즉시 공유되고, 개발자들이 빠르게 패치를 제공함으로써 서비스 중단 시간을 최소화하고 있습니다. 이러한 운영 방식은 기존 SearX와 비교할 때, 훨씬 높은 수준의 서비스 안정성과 신뢰성을 보장합니다.

결과적으로, SearXNG는 단순한 코드 fork가 아니라, 오픈소스 메타검색 엔진의 실질적인 발전을 이끈 프로젝트로 평가받고 있습니다. 기관이나 기업 입장에서는 SearXNG를 도입함으로써, 프라이버시 보호와 서비스 안정성, 그리고 신속한 유지보수라는 세 가지 핵심 가치를 동시에 실현할 수 있게 되었으며, 이는 SearXNG가 오픈소스 메타검색 엔진 분야에서 사실상 표준 솔루션으로 자리잡는 데 중요한 역할을 하였습니다.

## 1.2 라이선스(기업/기관 관점에서 반드시 짚어야 할 의무)

SearXNG를 도입하려는 기업이나 기관에서는 반드시 오픈소스 라이선스의 실무적 함의를 점검해야 합니다. SearXNG는 GNU AGPL-3.0 라이선스로 배포되며, 이는 단순한 소스 코드 공개 의무를 넘어 네트워크를 통한 서비스 제공 시에도 파생 저작물의 소스 공개가 요구되는 강력한 라이선스입니다. AGPL은 SaaS 형태로 외부에 서비스를 제공할 경우, 해당 소프트웨어의 수정본이나 파생 저작물을 사용자에게 공개해야 한다는 조건을 명시하고 있습니다. 반면, 사내 온프레미스 환경에서 내부 사용자에게만 서비스를 제공하는 경우에는 공개 의무의 범위가 다소 달라질 수 있습니다. 이와 같은 라이선스 조건은 기술 도입 시 법적 리스크를 최소화하기 위해 내부 정책 체크리스트를 반드시 마련해야 하는 이유입니다. 특히, 설정 파일이나 플러그인, 엔진 커스터마이징이 파생 저작물에 해당할 수 있는지, 운영 문서화와 소스 제공 방식 등 실무적 컴플라이언스 항목을 꼼꼼히 점검하는 것이 중요합니다.

### 1.2.1 SearXNG 오픈소스 라이선스는 무엇인가

SearXNG는 GNU Affero General Public License v3.0(AGPL-3.0)으로 배포되는 오픈소스 프로젝트입니다. AGPL-3.0은 GPL 계열 라이선스 중에서도 네트워크를 통한 소프트웨어 제공에 대한 소스 공개 의무를 강화한 형태로, SaaS나 웹 서비스 형태로 소프트웨어를 운영할 때 파생 저작물의 소스 코드를 사용자에게 제공해야 함을 명시하고 있습니다.

AGPL-3.0의 핵심 조항은 다음과 같습니다. 첫째, 소프트웨어를 수정하거나 파생 저작물을 만들고 이를 네트워크를 통해 사용자에게 제공하는 경우, 해당 소스 코드를 사용자에게 접근 가능하도록 해야 합니다. 둘째, 단순히 내부에서 사용하거나, 사내망에서만 서비스하는 경우에는 공개 의무의 범위가 외부 제공과는 다소 차이가 있을 수 있습니다. 셋째, 설정 파일, 플러그인, 엔진 커스터마이징 등이 파생 저작물로 간주될 수 있으며, 이를 외부에 서비스할 경우 소스 공개 의무가 발생할 수 있습니다.

실무적으로 AGPL-3.0 라이선스는 기업이나 기관에서 SearXNG를 도입할 때 다음과 같은 체크포인트를 마련해야 합니다. 첫째, 사내 온프레미스 환경에서 내부 사용자에게만 서비스를 제공하는 경우, 소스 공개 의무는 내부 정책에 따라 해석될 수 있으나, 외부 서비스 제공(예: SaaS, 공용 검색 인스턴스 운영 등) 시에는 반드시 수정본 및 파생 저작물의 소스 코드를 사용자에게 제공해야

합니다. 둘째, 설정 파일이나 플러그인, 엔진 커스터마이징이 파생 저작물에 해당할 수 있으므로, 이를 외부에 서비스할 경우 소스 공개 범위를 명확히 정의해야 합니다. 셋째, 운영 문서화(사용한 버전, 변경점, 소스 제공 방식 등)를 통해 컴플라이언스 리스크를 최소화해야 합니다.

예를 들어, SearXNG를 내부망에서만 운영하는 기관에서는 AGPL-3.0의 공개 의무가 외부 사용자에게 미치지 않으므로, 내부 정책에 따라 소스 코드 관리와 변경점 기록을 하면 됩니다. 반면, SearXNG를 외부 서비스로 제공하거나, 공용 인스턴스를 운영하는 경우에는 AGPL-3.0의 소스 공개 의무를 반드시 준수해야 하며, 수정본이나 커스터마이징된 구성 요소의 소스 코드도 사용자에게 접근 가능하도록 해야 합니다.

AGPL-3.0은 오픈소스 생태계에서 프라이버시 보호와 사용자 권리 보장을 강화하는 라이선스이지만, 기업이나 기관에서는 기술 도입 시 라이선스 리스크를 반드시 점검하고, 내부 정책과 컴플라이언스 체크리스트를 마련하는 것이 중요합니다. 이를 통해 기술 도입이 법적 리스크로 인해 중단되는 상황을 예방할 수 있으며, SearXNG의 안정적인 운영과 서비스 제공이 가능합니다.

AGPL-3.0의 실무적 함의를 좀 더 구체적으로 살펴보면, 이 라이선스는 단순히 소스 코드를 공개하는 것에 그치지 않고, 네트워크를 통해 소프트웨어를 제공하는 모든 경우에 대해 파생 저작물의 소스 코드까지 공개해야 한다는 점이 특징입니다. 예를 들어, SearXNG를 SaaS 형태로 외부 고객에게 제공하는 경우, 단순히 원본 소스 코드뿐만 아니라, 조직에서 커스터마이징한 설정 파일, 플러그인, 엔진 커넥터 등도 모두 공개 대상이 될 수 있습니다. 이때, 소스 코드 공개 방식은 웹사이트, GitHub 저장소, 혹은 별도의 아카이브 등 다양한 방법을 사용할 수 있으며, 사용자가 쉽게 접근할 수 있도록 안내해야 합니다.

또한, AGPL-3.0은 오픈소스 커뮤니티의 투명성과 사용자 권리 보장을 강화하는 역할을 합니다. 기업이나 기관이 SearXNG를 도입할 때, 라이선스 위반으로 인한 법적 분쟁을 예방하기 위해서는, 내부적으로 소스 코드 관리 체계를 갖추고, 변경 이력을 명확히 기록하는 것이 필수적입니다. 특히, 외부 서비스 제공 시에는 소스 코드 공개 범위와 방법을 명확히 정의하여, 사용자와의 신뢰를 확보하는 것이 중요합니다.

마지막으로, AGPL-3.0은 오픈소스 소프트웨어의 발전과 확산을 촉진하는 동시에, 기업이나 기관이 기술 도입 과정에서 발생할 수 있는 법적 리스크를 사전에 관리할 수 있도록 하는 역할을 합니다. 따라서, SearXNG와 같은 AGPL-3.0 기반 오픈소스 프로젝트를 도입할 때에는, 반드시 내부 정책과 컴플라이언스 체크리스트를 마련하고, 실무적으로 소스 코드 공개 및 관리 체계를 구축하는 것이 안정적인 서비스 운영의 핵심이라고 할 수 있습니다.

### 1.2.2 배포 형태별 컴플라이언스 체크리스트

SearXNG의 AGPL-3.0 라이선스는 배포 형태에 따라 컴플라이언스 체크리스트가 달라집니다. 기업이나 기관에서는 다음과 같은 항목을 반드시 점검해야 합니다.

1. **사내망 제공(내부 사용자에게 네트워크로 제공) 시**
    - SearXNG를 사내망에서 내부 사용자에게만 제공하는 경우, AGPL-3.0의 소스 공개 의무는 외부 사용자에게 미치지 않습니다. 그러나 내부 정책에 따라 수정본, 설정 파일, 플러그인, 엔진 커스터마이징 등 파생 저작물의 소스 코드 관리와 변경점 기록을 철저히 해야 합니다.
    - 내부 Git 저장소나 아카이브를 통해 소스 코드와 변경점을 관리하며, 운영 문서화(사용한 버전, 변경점, 소스 제공 방식 등)를 통해 컴플라이언스 리스크를 최소화합니다.
2. **외부 서비스로 제공(SaaS, 공용 인스턴스 운영 등) 시**
    - SearXNG를 외부 서비스로 제공하거나, 공용 검색 인스턴스를 운영하는 경우에는 AGPL-3.0의 소스 공개 의무를 반드시 준수해야 합니다. 수정본이나 커스터마이징된 구성 요소(설정 파일, 플러그인, 엔진 커스터마이징 등)의 소스 코드도 사용자에게 접근 가능하도록 해야 합니다.
    - 소스 코드 공개 방식은 웹사이트, GitHub 저장소, 아카이브 등 다양한 형태로 제공할 수 있으며, 사용자에게 명확한 접근 경로를 안내해야 합니다.
3. **설정 파일/플러그인/엔진 커스터마이징의 파생 저작물 여부**
    - 설정 파일, 플러그인, 엔진 커스터마이징이 파생 저작물에 해당할 수 있으므로, 외부 서비스 제공 시 소스 공개 범위를 명확히 정의해야 합니다.
    - 내부 정책에서는 파생 저작물의 범위와 소스 코드 관리 방식을 명확히 규정하고, 운영 문서화(사용한 버전, 변경점, 소스 제공 방식 등)를 통해 컴플라이언스 리스크를 최소화합니다.
4. **운영 문서화 및 소스 제공 방식**
    - 사용한 SearXNG 버전, 변경점, 소스 제공 방식(내부 Git/아카이브 등)을 명확히 기록하여, 컴플라이언스 리스크를 최소화합니다.

- 외부 서비스 제공 시에는 사용자에게 소스 코드 접근 경로를 명확히 안내해야 하며, 내부 운영 시에는 정책에 따라 소스 코드 관리와 변경점 기록을 철저히 해야 합니다.

이와 같은 컴플라이언스 체크리스트를 마련함으로써, 기업이나 기관은 SearXNG 도입 시 라이선스 리스크를 최소화하고, 기술 도입이 법적 문제로 인해 중단되는 상황을 예방할 수 있습니다. 실무적으로는 내부 정책과 운영 문서화를 통해 컴플라이언스 리스크를 관리하며, 외부 서비스 제공 시에는 AGPL-3.0의 소스 공개 의무를 반드시 준수하는 것이 중요합니다. 이를 통해 SearXNG의 안정적인 운영과 서비스 제공이 가능하며, 오픈소스 생태계의 신뢰성과 투명성을 확보할 수 있습니다.

배포 형태별로 컴플라이언스 체크리스트를 좀 더 구체적으로 살펴보면, 사내망에서만 SearXNG를 운영하는 경우에는 외부 공개 의무가 상대적으로 약하지만, 내부적으로도 소스 코드와 변경 이력을 체계적으로 관리해야 합니다. 예를 들어, 내부 Git 저장소를 활용하여 모든 커스터마이징 내역을 기록하고, 운영자 간에 변경 사항을 투명하게 공유하는 것이 중요합니다. 또한, 내부 정책 문서에 AGPL-3.0의 주요 조항과 소스 코드 관리 방침을 명확히 명시함으로써, 향후 외부 서비스 전환 시에도 법적 리스크를 최소화할 수 있습니다.

반면, SearXNG를 외부에 서비스하는 경우에는, 사용자가 소스 코드에 쉽게 접근할 수 있도록 웹사이트나 별도의 저장소를 운영해야 하며, 모든 커스터마이징 내역도 함께 공개해야 합니다. 이때, 소스 코드 공개 방식과 접근 경로를 명확히 안내하는 것이 필수적입니다. 예를 들어, 서비스 홈페이지에 '소스 코드 공개' 메뉴를 별도로 마련하거나, GitHub 저장소 링크를 제공하는 방식이 일반적입니다.

설정 파일, 플러그인, 엔진 커스터마이징 등은 실제 서비스의 핵심 기능에 영향을 미칠 수 있으므로, 이들 역시 파생 저작물로 간주되어 공개 대상이 될 수 있습니다. 따라서, 내부 정책에서는 이러한 구성 요소의 관리 및 공개 방침을 명확히 규정해야 하며, 운영 문서화 과정을 통해 모든 변경 사항을 체계적으로 기록하는 것이 중요합니다.

마지막으로, 운영 문서화는 단순히 소스 코드 관리에 그치지 않고, 사용한 SearXNG 버전, 적용된 패치 내역, 소스 코드 공개 방식 등 모든 컴플라이언스 관련 정보를 포함해야 합니다. 이를 통해, 감사나 법적 분쟁 발생 시 신속하게 대응할 수 있으며, 오픈소스 생태계 내에서 신뢰받는 서비스로 자리매김할 수 있습니다. 이처럼, 배포 형태별로 세분화된 컴플라이언스 체크리스트를 마련하고, 실무적으로 이를 철저히 준수하는 것이 SearXNG의 안정적이고 합법적인 운영의 핵심이라고 할

수 있습니다.

# 2장. 왜 '사내 메타검색 엔진'이 AI Agent 운영의 필수 인프라가 되었나

## 2.1 외부 검색 API의 비용·보안 딜레마를 구조적으로 해소하는 방법

AI Agent 및 자동화 시스템을 도입하는 기업·기관에서는 외부 검색 API를 활용할 때 여러 가지 구조적 제약에 직면하게 됩니다. 대표적인 예로 Google, Bing, DuckDuckGo 등 주요 검색 서비스의 API는 비용, 쿼터, 정책 변경, 그리고 보안과 감사 요구사항에 따라 운영 환경이 크게 달라질 수 있습니다. 특히 최근 AI 기반 업무 자동화가 확산되면서, 검색 API 호출량이 급증하고, 이에 따른 비용 예측이 어려워지는 문제가 반복적으로 발생하고 있습니다. 또한 기관이나 기업의 보안 정책상, 외부 서비스로의 데이터 반출이나 접속 로그 관리가 엄격하게 요구되는 경우가 많아, 외부 API 활용이 내부 감사 체계와 충돌하는 사례가 늘고 있습니다. 이러한 배경에서 사내 메타검색 엔진은 외부 API 의존도를 줄이고, 비용과 보안 리스크를 통제할 수 있는 대안으로 부각되고 있습니다. 본 절에서는 외부 검색 API가 가진 실무적 제약과, 이를 해소하기 위한 구조적 접근법을 집중적으로 다루겠습니다.

### 2.1.1 유료 검색 API의 공통 제약(비용, 쿼터, 정책 변경, 로그/메타데이터)

외부 검색 API를 활용하는 기업·기관은 여러 가지 실무적 제약에 직면합니다. 첫째, 비용 구조의 불확실성이 가장 큰 문제입니다. Google Custom Search API, Bing Search API 등은 기본적으로 유료이며, 쿼터 제한이 존재합니다. 예를 들어, 월별 무료 쿼터를 초과하면 추가 비용이 발생하고, 환율이나 정책 변경에 따라 예산이 예측 불가능하게 변동될 수 있습니다. 최근 클라우드 회귀(Cloud Repatriation) 현상에서도 볼 수 있듯이, 퍼블릭 클라우드 및 외부 API 사용 비용이 해마다 상승하고, 예상치 못한 추가 비용이 발생하여 IT 예산 관리에 어려움을 겪는 사례가 많습니다(OPENMARU Newsletter 52호).

둘째, 쿼터와 정책 변경의 불안정성입니다. 검색 API 제공자는 정책을 수시로 변경할 수 있으며, API 호출량 제한이나 사용 조건이 강화될 때 서비스 전체가 영향을 받을 수 있습니다. 예를 들어,

Google은 API 사용량이 급증하거나 비정상적인 트래픽이 감지되면 CAPTCHA나 rate limit을 적용하여, 서비스가 일시적으로 중단되는 상황이 발생합니다. 이러한 정책 변경은 AI Agent의 자동화 파이프라인에 직접적인 장애를 유발할 수 있습니다.

셋째, 보안 및 감사 체계와의 충돌입니다. 기관·기업은 접속 로그, 데이터 반출, 메타데이터 관리 등 보안 요구가 매우 엄격합니다. 외부 API를 사용할 경우, 검색 쿼리와 결과가 외부 벤더 서버를 거치게 되며, 내부 감사 체계에서 데이터 유출이나 개인정보 보호 이슈가 발생할 수 있습니다. 특히 공공기관이나 금융권에서는 외부 서비스 사용 시 데이터 주권 확보와 내부망 감사 요구가 충돌하는 경우가 많습니다. API 호출 로그, 사용자 정보, 검색 키워드 등이 외부로 전송되는 상황은 내부 정책상 허용되지 않는 경우가 많으며, 이로 인해 외부 API 활용이 제한되는 사례가 늘고 있습니다.

실무적으로는 API 사용량 모니터링, 정책 변경 대응, 보안 감사 로그 관리 등 복잡한 운영 절차가 필요하며, 이를 자동화하거나 통제하는 데 추가적인 인력과 비용이 소모됩니다. API 호출 실패, 데이터 누락, 정책 변경에 따른 서비스 중단 등은 AI Agent 운영의 안정성을 저해하는 주요 원인입니다. 이러한 제약을 해소하기 위해서는 온프레미스 환경에서 자체 메타검색 엔진을 구축하여, 외부 API 의존도를 줄이고, 비용·보안·감사 요구를 자체적으로 통제하는 전략이 필요합니다.

실제 사례를 보면, 대형 금융기관이나 공공기관은 외부 검색 API를 최소화하고, 내부망에서 메타검색 엔진을 운영하여 데이터 반출과 접속 로그를 자체적으로 관리합니다. 이 과정에서 API 게이트웨이, 프록시, 내부 감사 시스템과 연동하여, 외부 API 호출을 엄격하게 제한하거나, 자체 검색 인프라를 구축하는 방식이 점차 확산되고 있습니다. 한편, 외부 API의 정책 변경이나 비용 상승에 대응하기 위한 예산 관리, 서비스 장애 대응 프로세스도 필수적으로 마련되어야 하며, 이러한 운영 부담이 기관·기업의 IT 전략에 큰 영향을 미치고 있습니다.

마지막으로, 외부 API 활용 시에는 서비스별 데이터베이스 전략, API 게이트웨이, 서비스 메시 등 중앙화된 관리 체계를 도입하여, 트래픽 제어, 인증, 보안, 감사 로그를 통합적으로 관리하는 것이 중요합니다. 그러나 이러한 중앙화 관리 체계도 외부 API의 근본적인 제약을 완전히 해소하지는 못하며, 온프레미스 메타검색 엔진의 필요성이 더욱 강조되고 있습니다.

### 2.1.2 메타검색이 단일 검색보다 유리한 이유("엔진 다양성 = 리스크 분산")

단일 검색 엔진에 의존하는 경우, 장애나 차단이 발생하면 전체 AI Agent 파이프라인이 멈추는 위험이 존재합니다. 예를 들어, Google이나 Bing 검색 API에 rate limit, CAPTCHA, 또는 정책

변경이 적용되면, 모든 검색 기반 에이전트가 일시적으로 중단될 수 있습니다. 이러한 상황은 자동화된 업무 처리, 정보 수집, 실시간 검색이 중요한 기관·기업에서는 심각한 운영 장애로 이어질 수 있습니다.

메타검색 엔진은 여러 검색 엔진을 통합하여 결과를 취합하는 구조를 갖고 있습니다. SearXNG와 같은 메타검색 엔진은 다양한 검색 소스(예: Google, Bing, DuckDuckGo, Wikipedia, 뉴스, 논문 등)를 동시에 연동하고, 각 엔진의 결과를 집계하여 사용자에게 제공하는 방식입니다. 이 구조는 엔진 다양성을 확보함으로써, 단일 엔진 장애 시에도 전체 서비스가 멈추지 않고, 다른 엔진을 통해 결과를 지속적으로 제공할 수 있는 리스크 분산 효과를 얻을 수 있습니다.

실무적으로는 엔진별 rate limit, bot protection, CAPTCHA 대응이 중요합니다. SearX 계열 메타검색 엔진은 서버가 검색을 대행하므로, 검색 엔진에서 봇으로 분류되어 차단될 가능성이 높습니다. 이를 방지하기 위해 IP rate limitation, 엔진 믹스 조정, 캐시/백오프, 프록시/출구 IP 전략 등이 필요합니다. 예를 들어, SearXNG는 Valkey와 같은 리미터를 활용하여 검색 요청을 조절하고, 엔진별로 호출 빈도와 IP를 분산시켜 차단 리스크를 최소화합니다.

또한, 메타검색 엔진은 엔진별 API 포맷, 응답 구조, 정책 변경에 유연하게 대응할 수 있습니다. 단일 엔진 장애 시, 다른 엔진을 통해 검색 결과를 지속적으로 확보할 수 있으며, 엔진별로 쿼터, 비용, 정책 변경을 분산하여 운영 부담을 줄일 수 있습니다. 실제로 대규모 기관에서는 SearXNG 등 메타검색 엔진을 자체적으로 운영하여, 검색 엔진 장애나 차단에 대비하고, 검색 결과의 품질과 안정성을 유지하고 있습니다.

한편, 메타검색 엔진은 검색 결과의 다양성과 통합성을 확보할 수 있습니다. 단일 엔진은 특정 알고리즘이나 인덱스에 의존하지만, 메타검색은 여러 엔진의 결과를 비교·집계함으로써, 정보의 신뢰성과 최신성을 높일 수 있습니다. 이 구조는 AI Agent가 다양한 출처에서 정보를 수집하고, 장애나 차단에 유연하게 대응할 수 있도록 지원합니다.

마지막으로, 메타검색 엔진은 내부망에서 운영할 경우, 데이터 반출과 접속 로그를 자체적으로 통제할 수 있어, 보안과 감사 요구에 효과적으로 대응할 수 있습니다. 엔진 다양성은 운영 리스크를 분산시키고, AI Agent의 안정적인 정보 수집과 자동화 파이프라인 구축에 핵심적인 역할을 합니다.

## 2.2 AI Agent 관점: “검색→수집→정제→RAG” 파이프라인에서 SearXNG의 위치

AI Agent와 LLM 기반 시스템이 본격적으로 도입되면서, 정보 검색과 수집, 정제, 그리고 RAG(**Retrieval-Augmented Generation**) 파이프라인의 중요성이 크게 부각되고 있습니다. 이 파이프라인은 외부 최신 정보의 탐색, 내부 지식 베이스의 의미 기반 재탐색, 그리고 최종 답변 생성까지의 전 과정을 포함합니다. SearXNG와 같은 사내 메타검색 엔진은 이 파이프라인에서 외부 정보 획득의 게이트웨이 역할을 담당하며, 키워드 검색과 벡터 검색의 역할 분리를 통해 정보의 신뢰성과 최신성을 보장합니다. 또한, 검색 결과를 크롤링·정제하여 구조화 데이터로 변환하고, RAG 인덱싱, 온톨로지/그래프DB 연계, 정책 기반 라우팅 등으로 확장하는 아키텍처가 점차 표준화되고 있습니다. 본 절에서는 AI Agent 관점에서 SearXNG의 위치와 역할, 그리고 파이프라인 확장 이유를 심층적으로 분석합니다.

### 2.2.1 키워드 검색과 벡터 검색(RAG)의 관계를 ‘역할 분리’로 설명

AI Agent가 정보를 획득하고 답변을 생성하는 과정에서, 키워드 검색과 벡터 검색은 명확한 역할 분리를 필요로 합니다. 키워드 검색은 외부 최신 정보의 탐색에 중점을 둡니다. 예를 들어, SearXNG를 통해 “2026년 최신 보안 트렌드”와 같은 키워드로 검색하면, 웹, 뉴스, 논문, 공식 문서 등 다양한 출처에서 최신 정보를 신속하게 확보할 수 있습니다. 이 과정은 무엇이 어디에 있는지, 어떤 정보가 최신·신뢰할 만한 출처에 있는지를 탐색하는 단계입니다.

반면, 벡터 검색은 내부 지식 베이스에서 의미 기반으로 정보를 재탐색하는 역할을 합니다. 예를 들어, 이미 수집된 문서, 내부 데이터, 과거 검색 결과를 벡터DB(예: Pinecone, Milvus, Qdrant 등)에 임베딩하여, AI Agent가 “이 내용이 무엇을 의미하는지”를 빠르게 찾아내고, 유사도 기반으로 답변을 생성할 수 있습니다. 이 과정은 정보의 의미, 맥락, 연관성을 중심으로 재탐색하는 단계이며, RAG(검색 기반 답변 생성) 파이프라인에서 핵심적인 역할을 담당합니다.

SearXNG는 외부 최신 정보의 획득 게이트웨이로 정의할 수 있습니다. 키워드 검색을 통해 신뢰할 만한 출처를 탐색하고, 검색 결과를 크롤링·정제하여 내부 지식 베이스에 적재합니다. 이후 벡터 검색을 통해 의미 기반 재탐색이 이루어지며, AI Agent는 최신 정보와 내부 지식의 조합을 통해 정확하고 근거 있는 답변을 생성할 수 있습니다.

실무적으로는, AI Agent가 SearXNG를 통해 외부 정보(예: 웹, 뉴스, 논문 등)를 획득하고, 이를 크롤링·정제하여 벡터DB에 임베딩합니다. 이후 사용자가 질문을 하면, 벡터 검색을 통해 의미 기반으로 유사한 정보를 찾아내고, 최신 정보와 내부 지식의 조합을 통해 답변을 생성합니다. 이 과정에서 키워드 검색과 벡터 검색의 역할 분리가 명확하게 이루어지며, 정보의 신뢰성과 최신성, 의미 기반 탐색의 효율성이 크게 향상됩니다.

한편, 키워드 검색만으로는 의미 기반 탐색이 어렵고, 벡터 검색만으로는 최신 정보를 반영하기 어렵다는 한계가 있습니다. 따라서 두 방식의 역할 분리가 필수적이며, SearXNG는 외부 정보 획득의 표준 인터페이스로서, AI Agent 파이프라인의 핵심 인프라로 자리잡고 있습니다.

마지막으로, 키워드 검색과 벡터 검색의 결합은 RAG 파이프라인에서 정보의 신뢰성, 최신성, 의미 기반 탐색을 동시에 보장할 수 있는 구조적 장점이 있습니다. SearXNG는 외부 정보 획득의 게이트웨이로서, AI Agent의 정보 수집과 답변 생성 파이프라인에서 핵심적인 역할을 수행합니다.

### 2.2.2 Web Scraping, 온톨로지/GraphDB까지 확장되는 이유

AI Agent가 SearXNG를 통해 검색한 결과는 일반적으로 링크 목록 형태로 제공됩니다. 이 링크 목록은 단순한 정보 탐색에 그치지 않고, 크롤링과 본문 추출 과정을 거쳐 구조화 데이터로 변환됩니다. 예를 들어, SearXNG를 통해 “2026년 보안 트렌드”를 검색하면, 관련 뉴스, 공식 보고서, 블로그 등 다양한 링크가 반환됩니다. AI Agent는 이 링크를 크롤링하여 본문을 추출하고, 텍스트 정제, 엔터티 추출, 관계 분석 등의 과정을 통해 구조화 데이터를 생성합니다.

이후, 구조화된 데이터는 RAG 인덱싱(벡터DB)에 적재되어, 의미 기반 검색과 답변 생성에 활용됩니다. 예를 들어, 본문에서 “TLS 1.3”, “OWASP Top 10”, “PCI DSS”와 같은 엔터티를 추출하고, 관계 분석을 통해 온톨로지나 그래프DB에 연동할 수 있습니다. 이 과정은 정보의 의미와 맥락, 연관성을 분석하여, AI Agent가 보다 정확하고 근거 있는 답변을 생성할 수 있도록 지원합니다.

또한, 정책·규정 기반 라우팅이 필요할 때, 내부·외부 데이터의 경계를 명확하게 설정하고, 검색 결과를 분리하여 처리할 수 있습니다. 예를 들어, 공공기관에서는 내부 정책에 따라 외부 검색 결과와 내부 데이터의 활용 범위를 구분하고, 감사·보안 요구에 맞게 데이터 라우팅을 설계합니다. 이 과정에서 SearXNG는 외부 정보 유입의 표준 인터페이스로서, 크롤링·정제·인덱싱·온톨로지 연계까지 아키텍처를 확장하는 기반을 제공합니다.

실무 사례를 보면, 대형 기관이나 연구소에서는 SearXNG를 통해 검색 결과를 수집하고, 자체 크롤러와 본문 추출기를 연동하여 구조화 데이터를 생성합니다. 이후 벡터DB와 그래프DB(예: Neo4j, TigerGraph 등)에 적재하여, AI Agent가 의미 기반 탐색과 관계 분석을 동시에 수행할 수 있도록 아키텍처를 구축합니다. 이 과정에서 엔터티 추출, 관계 분석, 정책 기반 라우팅 등 다양한 기술적 요소가 결합되어, 정보의 신뢰성, 의미, 맥락을 보장하는 구조가 완성됩니다.

한편, SearXNG가 모든 문제를 해결하는 것은 아니며, 외부 정보 유입의 표준 인터페이스로서 가치가 있다는 점이 중요합니다. 크롤링, 정제, 인덱싱, 온톨로지 연계 등은 별도의 모듈이나 시스템과 결합되어야 하며, SearXNG는 검색 결과 획득의 게이트웨이 역할을 담당합니다. 이 구조는 AI Agent 파이프라인에서 외부 정보 획득, 구조화, 의미 기반 탐색, 정책 기반 라우팅까지 확장 가능한 아키텍처를 제공하며, 기관·기업의 정보 수집과 자동화 전략에 핵심적인 역할을 수행합니다.

마지막으로, 검색 결과를 크롤링·정제하여 구조화 데이터로 변환하고, RAG 인덱싱, 온톨로지/그래프DB 연계, 정책 기반 라우팅 등으로 확장하는 아키텍처는 AI Agent 운영의 표준이 되고 있습니다. SearXNG는 외부 정보 유입의 표준 인터페이스로서, AI Agent 파이프라인의 핵심 인프라로 자리잡고 있습니다.

# 3장. Google/Bing 같은 검색 서비스와 SearXNG 메타검색의 차이

## 3.1 비교 관점 정의(기술·운영·거버넌스)

기업과 기관이 검색 인프라를 도입할 때, 단순히 검색 정확도만을 기준으로 의사결정을 내리기에는 한계가 있습니다. Google, Bing 등 상용 검색 엔진은 방대한 데이터와 고도화된 알고리즘을 기반으로 높은 정확도를 제공하지만, 실제 조직 환경에서는 기술적 통제권, 감사 가능성, 비용 구조, 프라이버시 보호, 그리고 시스템 통합성 등 다양한 운영·거버넌스 요소가 더 큰 비중을 차지하는 경우가 많습니다. 특히 검색 결과의 활용 목적이 단순 정보 탐색을 넘어 AI 에이전트, RAG 파이프라인, 내부 데이터 연계 등으로 확장되는 상황에서는 외부 서비스의 정책 변화, API 쿼터 제한, 로그 및 메타데이터 반출 이슈 등이 조직의 리스크로 작용할 수 있습니다. 반면, 온프레미스 메타검색 솔루션인 SearXNG는 조직이 직접 운영하며, 다양한 검색 엔진을 통합해 결과를 집계하는 구조를

통해 장애 대응과 리스크 분산이 가능합니다. 또한 내부망 배치와 사용자 추적 최소화 설계를 통해 데이터 경계와 프라이버시 보호를 실현할 수 있습니다. 이 장에서는 기술적 비교뿐 아니라, 운영 효율성과 거버넌스 관점에서 조직이 얻는 실질적 이점과 부담을 명확히 분석하여, 실무 의사결정에 필요한 핵심 기준을 제시합니다.

### 3.1.1 무엇을 비교해야 의사결정이 쉬워지는가

검색 인프라를 선택할 때 가장 먼저 떠오르는 비교 기준은 '검색 정확도'입니다. Google, Bing 등 상용 엔진은 방대한 크롤링 데이터와 최신 AI 기반 랭킹 알고리즘을 바탕으로 높은 품질의 검색 결과를 제공합니다. 실제로, 일반 사용자 입장에서는 특정 키워드에 대해 가장 적합한 결과를 빠르게 찾을 수 있는 점이 큰 장점입니다. 그러나 조직, 특히 기업이나 공공기관이 검색 시스템을 도입할 때는 단순한 정확도 이상의 요소가 중요해집니다.

첫째, 통제권과 감사 가능성입니다. 상용 검색 서비스는 벤더의 정책에 따라 API 사용, 데이터 접근, 로그 관리가 제한됩니다. 예를 들어, 검색 쿼리와 결과가 외부 벤더 서버로 전송되고, 그 과정에서 접속 로그나 메타데이터가 벤더 정책에 따라 저장·분석될 수 있습니다. 이는 내부 감사, 보안 점검, 데이터 반출 통제 요구가 높은 조직에서는 큰 리스크가 될 수 있습니다. 반면, SearXNG와 같은 온프레미스 메타검색 솔루션은 조직이 직접 서버를 운영하고, 네트워크 경계 내에서 데이터 흐름을 관리할 수 있습니다. 이를 통해 감사 체계와 보안 정책을 조직 내부 기준에 맞게 적용할 수 있습니다.

둘째, 비용 구조와 예측 가능성입니다. 상용 검색 API는 기본적으로 유료이며, 쿼터 제한, 정책 변경, 가격 변동이 빈번하게 발생합니다. 특히 대량 검색, 자동화된 에이전트 연계, AI 파이프라인 구축 등에서 API 비용이 급격히 증가할 수 있습니다. 반면, SearXNG는 오픈소스 소프트웨어로 제공되어 소프트웨어 자체 비용은 발생하지 않으며, 운영비는 인프라와 인력에 한정됩니다. 이로 인해 장기적 비용 예측이 가능하고, 정책 변화에 따른 갑작스러운 서비스 중단이나 추가 비용 부담을 최소화할 수 있습니다.

셋째, 프라이버시와 사용자 추적 이슈입니다. 상용 검색 엔진은 사용자 프로파일링, 광고 타겟팅 등 다양한 목적으로 데이터를 수집·분석합니다. 조직 내부에서 검색 데이터를 외부로 반출하는 것은 정보보호 관점에서 민감한 사안이 될 수 있습니다. SearXNG는 사용자 추적과 프로파일링을 하지 않는 것을 목표로 설계되어, 내부망에서 운영할 경우 프라이버시 보호와 데이터 경계 설정이

용이합니다.

마지막으로, 통합성과 엔진 다양성입니다. 상용 검색 서비스는 단일 엔진 인덱스에 기반하며, API 연계 방식이 벤더별로 상이합니다. 반면, SearXNG는 다수의 검색 엔진과 데이터베이스를 설정 기반으로 연동할 수 있으며, 표준화된 Search API를 통해 다양한 에이전트와 쉽게 통합할 수 있습니다. 장애 대응이나 엔진 차단 이슈 발생 시, 여러 엔진을 믹스해 리스크를 분산할 수 있는 구조적 강점이 있습니다.

이처럼 검색 인프라의 선택 기준은 단순한 정확도를 넘어, 통제권, 감사 가능성, 비용 구조, 프라이버시, 통합성 등 조직의 실무 요구에 맞춰 다각적으로 비교해야 의사결정이 쉬워집니다. 특히 AI 에이전트, RAG 파이프라인 등 최신 업무 환경에서는 이러한 요소들이 더욱 중요한 기준으로 작용합니다.

---

## 3.2 비교표: 검색 서비스 vs 온프레미스 메타검색(SearXNG)

검색 인프라의 선택은 조직의 기술적 요구와 운영 환경에 따라 달라질 수 있습니다. Google, Bing 등 상용 검색 엔진 서비스와 SearXNG 온프레미스 메타검색 솔루션은 제공 형태, 검색 방식, 비용 구조, 프라이버시, 보안·감사, 엔진 다양성, 차단·봇 이슈, 에이전트 연계 방식 등 여러 측면에서 뚜렷한 차이를 보입니다. 아래 비교표는 각 요소별 주요 차이점을 정리한 것으로, 조직이 실제로 얻는 이점과 부담을 명확히 파악할 수 있도록 도와줍니다.

| 구분 | Google/Bing 등 검색 엔진 서비스 | SearXNG(온프레미스 메타검색) |
|---|---|---|
| 제공 형태 | 벤더 운영 SaaS | 조직이 직접 운영(self-host) |
| 검색 방식 | 단일 엔진 인덱스 기반 | 다수 엔진/DB 결과를 **집계**하는 메타검색 |
| 비용 구조 | API 유료/쿼터/정책 종속 | 소프트웨어 자체는 OSS(운영비는 인프라/인력) |
| 프라이버시/추적 | 벤더 정책에 의존 | “사용자 추적/프로파일링을 하지 않음”을 목표로 설계 |
| 보안/감사 | 외부 반출/접속 로그 통제 어려움 | 내부망 배치로 데이터 경계 설정 용이(감사/망분리 대응) |
| 엔진 다양성 | 엔진 단일 | 수십~수백 검색 소스 연동 가능(설정 기반) |
| 차단/봇 이슈 | 사용자 직접 접근 | 서버가 대행 호출 → rate limit/bot protection 필요 |
| Agent 연계 | 벤더별 API 상이 | 표준화된 Search API(JSON 등) 제공 가능 |

조직이 SearXNG 온프레미스 메타검색을 도입할 경우 얻을 수 있는 가장 큰 이점은 통제권과 감사 가능성, 그리고 비용 예측의 안정성입니다. 내부망에 배치하면 데이터 반출과 접속 로그를 조직 정책에 맞게 관리할 수 있고, API 비용이나 쿼터 제한에 대한 걱정 없이 장기적 운영이 가능합니다. 또한, 다양한 검색 엔진을 연동해 장애 대응과 리스크 분산이 용이하며, 표준화된 Search API를 통해 AI 에이전트, RAG 파이프라인 등과 쉽게 통합할 수 있습니다. 반면, 조직이 떠안아야 하는 부담은 운영·유지보수, 엔진 차단 대응, 튜닝 작업 등입니다. 예를 들어, 검색 엔진의 변경이나 장애 발생 시 직접 대응해야 하며, rate limit/bot protection 설정 등 운영 노하우가 필요합니다. 또한, 인프라와 인력에 대한 투자와 지속적인 모니터링·최적화가 요구됩니다. 이러한 비교를 통해 조직은 실무 환경에 맞는 검색 인프라 전략을 수립할 수 있습니다.

상용 검색 엔진 서비스와 SearXNG 온프레미스 메타검색 솔루션의 차이는 단순히 기술적인 측면에만 국한되지 않습니다. 실제로, 조직의 규모와 보안 요구사항, 데이터 민감도, 그리고 장기적인 운영 전략에 따라 선택의 기준이 달라질 수 있습니다. 예를 들어, 금융기관이나 공공기관처럼 데이터 유출에 민감한 조직은 외부 서비스의 로그 관리 및 데이터 반출 통제가 어려운 상용 검색 엔진보다는, 내부망에서 직접 관리할 수 있는 SearXNG와 같은 솔루션을 선호할 수 있습니다. 반면, 신속한 도입과 최소한의 유지보수를 원하는 스타트업이나 소규모 조직은 벤더가 제공하는 SaaS 형태의 검색 엔진을 선택하는 것이 효율적일 수 있습니다.

또한, SearXNG는 다양한 검색 엔진과 데이터베이스를 연동할 수 있는 유연성을 제공하여, 특정 엔진의 장애나 정책 변화에 따른 리스크를 분산할 수 있습니다. 예를 들어, Google 검색 API가 일시적으로 차단되거나 쿼터가 초과되는 상황에서도, SearXNG는 Bing, DuckDuckGo, 자체 구축한 사내 데이터베이스 등 다양한 소스를 동시에 활용할 수 있습니다. 이로 인해 검색 서비스의 연속성과 안정성이 크게 향상됩니다.

비용 측면에서도 SearXNG는 오픈소스 기반이기 때문에, 라이선스 비용 부담이 없으며, 인프라와 인력에만 비용이 소요됩니다. 이는 장기적으로 예산 계획을 수립하는 데 있어 큰 장점이 될 수 있습니다. 반면, 상용 검색 엔진은 쿼터 초과 시 추가 비용이 발생하거나, 정책 변경에 따라 예기치 않은 비용 증가가 발생할 수 있습니다.

마지막으로, 표준화된 Search API를 제공하는 SearXNG는 AI 에이전트, 챗봇, RAG 파이프라인 등과의 연동이 매우 용이합니다. 이는 조직이 디지털 전환을 추진하거나, AI 기반의 업무

자동화를 도입할 때 중요한 경쟁력이 될 수 있습니다. 이러한 다양한 비교 요소를 종합적으로 고려하여, 조직의 목표와 환경에 가장 적합한 검색 인프라를 선택하는 것이 중요합니다.

---

## 3.3 SearXNG의 Search API가 'Agent Builder'에 중요한 이유

AI 에이전트와 RAG 기반 파이프라인이 확산되면서, 검색 결과를 자동으로 수집·정제하고 의미 기반으로 재탐색하는 작업이 필수 인프라로 자리잡고 있습니다. SearXNG는 온프레미스 환경에서 다양한 검색 엔진을 통합해 결과를 집계할 뿐 아니라, 표준화된 Search API를 제공함으로써 에이전트 빌더, RAG 서비스, 내부 업무 시스템과의 연계가 매우 용이합니다. 특히 JSON, CSV, RSS 등 다양한 출력 포맷을 지원하며, settings.yml 파일에서 허용된 포맷만 응답하도록 설계되어 있어 보안과 운영 통제 측면에서 강점을 갖습니다. 공용 인스턴스에서는 보안상의 이유로 JSON 포맷을 비활성화하는 경우가 많지만, 사내 인스턴스에서는 목적에 맞게 활성화하여 AI 에이전트와의 연동을 최적화할 수 있습니다. 이처럼 SearXNG의 Search API는 조직의 검색 인프라를 AI 중심 업무 환경에 맞게 유연하게 확장할 수 있는 핵심 도구로 평가받고 있습니다.

SearXNG의 Search API는 단순히 검색 결과를 반환하는 기능을 넘어, 조직의 업무 자동화와 AI 기반 정보 처리의 중심축 역할을 수행합니다. 예를 들어, RAG(Retrieval-Augmented Generation) 파이프라인에서는 외부 지식 소스에서 최신 정보를 검색하고, 이를 LLM(Large Language Model)이 참고하여 답변을 생성하게 됩니다. 이때 SearXNG의 Search API는 다양한 검색 엔진의 결과를 통합해 표준화된 데이터로 제공함으로써, RAG 파이프라인의 데이터 수집 효율성과 품질을 크게 향상시킵니다. 또한, 에이전트 빌더가 SearXNG의 API를 통해 실시간 검색 결과를 받아와, 내부 업무 시스템이나 챗봇, 자동화된 보고서 생성 등 다양한 업무 시나리오에 활용할 수 있습니다.

특히, SearXNG는 settings.yml 파일을 통해 API 응답 포맷, 허용 엔진, 인증 방식, 트래픽 제한 등 세밀한 운영 정책을 설정할 수 있습니다. 이를 통해 조직은 보안과 데이터 경계를 유지하면서도, 필요한 업무에 맞는 최적화된 검색 인프라를 구축할 수 있습니다. 예를 들어, 특정 부서만 API 접근을 허용하거나, 특정 시간대에만 대량 검색을 허용하는 등 유연한 정책 적용이 가능합니다.

실제 사례로, 대형 금융기관에서는 SearXNG를 내부망에 배치하여, 외부 인터넷과의 직접 연결 없이도 다양한 검색 엔진의 결과를 통합적으로 활용하고 있습니다. 이를 통해 외부 데이터 반출

위험을 최소화하면서, AI 기반 업무 자동화와 정보 검색의 효율성을 동시에 달성하고 있습니다. 또한, SearXNG의 표준화된 API 덕분에, 신규 AI 에이전트나 챗봇을 도입할 때 별도의 검색 엔진 연동 개발 없이 손쉽게 시스템을 확장할 수 있습니다.

이처럼 SearXNG의 Search API는 조직이 AI 중심의 업무 환경으로 전환할 때, 핵심적인 연결고리 역할을 하며, 보안, 확장성, 운영 효율성 측면에서 탁월한 가치를 제공합니다.

### 3.3.1 JSON 포맷 활성화와 보안(기본 비활성인 이유 포함)

SearXNG의 Search API는 AI 에이전트, RAG 파이프라인, 내부 업무 시스템 등 다양한 활용 시나리오에서 중요한 역할을 합니다. 특히 검색 결과를 JSON, CSV, RSS 등 구조화된 데이터 포맷으로 받을 수 있어, 자동화된 데이터 수집·정제, 의미 기반 인덱싱, 근거 링크 유지 등 다양한 업무에 직접 활용할 수 있습니다. 운영자는 settings.yml 파일을 통해 허용할 출력 포맷을 세밀하게 제어할 수 있으며, 미허용 포맷 요청 시 403 Forbidden 응답을 반환하도록 설계되어 있습니다. 이러한 구조는 보안과 운영 통제 측면에서 큰 장점을 제공합니다.

공용 SearXNG 인스턴스에서는 JSON 포맷을 기본적으로 비활성화하는 경우가 많습니다. 그 이유는 구조화된 데이터가 자동화된 봇, 크롤러, 외부 에이전트에 의해 대량 수집될 가능성이 높기 때문입니다. 실제로 JSON 포맷은 API 연동, 자동화된 데이터 수집에 최적화되어 있어, 악의적 사용이나 과도한 트래픽 발생 시 서비스 안정성에 영향을 줄 수 있습니다. 또한, 공용 인스턴스에서는 보안상의 이유로 검색 결과의 대량 반출이나 자동화된 접근을 제한해야 할 필요가 있습니다.

반면, 사내 인스턴스에서는 목적에 맞게 JSON 포맷을 활성화하는 것이 일반적입니다. 예를 들어, AI 에이전트가 SearXNG를 통해 검색 결과를 수집하고, 이를 RAG 파이프라인이나 내부 업무 시스템에 연계할 때 JSON 포맷이 필수적입니다. 운영자는 settings.yml에서 JSON 출력 활성화, 엔진별 접근 제어, rate limit/bot protection 설정 등 세밀한 운영 정책을 적용할 수 있습니다. 이를 통해 보안과 데이터 경계, 서비스 안정성을 동시에 확보하면서, AI 기반 업무 환경에 최적화된 검색 인프라를 구축할 수 있습니다.

실무적으로는 다음과 같은 체크리스트를 권장합니다. 첫째, 사내 인스턴스에서 JSON 출력 활성화 여부를 반드시 확인하고, 목적에 맞는 포맷만 허용하도록 설정합니다. 둘째, 엔진별 접근 제어와 rate limit 정책을 통해 과도한 트래픽이나 봇 접근을 통제합니다. 셋째, Search API 연동 시 인증·접근 제어 정책을 적용해 외부 반출을 방지합니다. 넷째, 운영 로그와 감사 체계를 구축해

검색 결과 활용 내역을 추적·관리합니다. 마지막으로, AI 에이전트와의 연동 테스트를 통해 실제 업무 시나리오에 맞는 최적화된 운영 환경을 확보해야 합니다.

이처럼 SearXNG의 Search API는 조직이 AI 중심 업무 환경으로 전환할 때, 보안과 통제, 확장성을 동시에 실현할 수 있는 핵심 도구입니다. JSON 포맷의 활성화와 세밀한 운영 정책 적용은 조직의 검색 인프라를 미래 업무 환경에 맞게 유연하게 확장할 수 있는 기반을 제공합니다.

JSON 포맷을 활성화할 때는 추가적인 보안 고려사항이 필요합니다. 예를 들어, 인증되지 않은 접근을 차단하기 위해 API 키 또는 내부망 IP 화이트리스트를 적용할 수 있습니다. 또한, 검색 쿼리 및 결과에 포함될 수 있는 민감한 정보가 외부로 유출되지 않도록, 응답 데이터에 대한 필터링과 로깅 정책을 강화해야 합니다. 실제로 일부 조직에서는 검색 API의 접근 로그를 별도로 저장하고, 이상 트래픽이나 비정상적인 접근 패턴을 실시간으로 모니터링하여, 잠재적인 보안 위협에 신속하게 대응하고 있습니다.

또한, JSON 포맷을 통한 대량 데이터 수집이 서비스 성능에 영향을 미치지 않도록, API 호출 빈도 제한(rate limiting)과 동시 접속 제한(concurrent connection limit) 정책을 병행 적용하는 것이 바람직합니다. 이를 통해 정상적인 업무용 트래픽은 원활하게 처리하면서도, 과도한 자동화 요청이나 악의적 접근으로 인한 서비스 장애를 예방할 수 있습니다.

마지막으로, JSON 포맷 활성화는 조직의 디지털 전환과 AI 기반 업무 자동화에 있어 필수적인 요소이지만, 그만큼 보안과 운영 정책의 정교한 설계가 전제되어야 합니다. 조직의 보안 담당자와 시스템 운영자가 협력하여, Search API의 활용 목적, 접근 권한, 데이터 보호 정책을 명확히 정의하고, 정기적으로 정책의 적정성을 점검하는 것이 중요합니다. 이러한 노력이 뒷받침될 때, SearXNG의 Search API와 JSON 포맷은 조직의 혁신과 경쟁력 강화에 실질적인 기여를 할 수 있습니다.

# 4장. 구축 및 운영: 컨테이너/쿠버네티스 환경에서의 SearXNG 실전 배치

## 4.1 레퍼런스 아키텍처(온프레미스 표준)

SearXNG를 온프레미스 환경에서 안정적으로 운영하기 위해서는 표준화된 레퍼런스 아키텍처가 필요합니다. 최근 기업 및 기관에서는 자체 검색 인프라를 빠르게 배포하고, 보안과 가용성을 동시에 확보하는 것이 핵심 과제로 부상하고 있습니다. 특히 메타검색 엔진은 다양한 외부 소스와 연동되기 때문에, 네트워크 경계와 인증, 캐시, 레이트리밋 등 여러 운영 요소를 체계적으로 설계해야 합니다. SearXNG 공식 Docker Compose 패키지는 이러한 요구를 반영하여, 단일 패키지로 신속하게 배포할 수 있는 구조를 제공합니다. 이 구성은 SearXNG 본체, 리버스 프록시, 그리고 Valkey(리미터/캐시)를 아우르며, 보안 인증서 관리와 차단 대응, 운영 자동화까지 실무에 적합한 요소들을 포함하고 있습니다. 본 절에서는 각 컴포넌트의 역할과 상호작용, 그리고 실제 배포 시 고려해야 할 보안 및 장애 대응 포인트를 상세히 해설합니다.

### 4.1.1 Docker Compose 기반 “5분 배포” 구성 요소 해부

Docker Compose를 활용한 SearXNG 배포는 실무에서 가장 빠르고 간편하게 시작할 수 있는 방법입니다. 공식 searxng-docker 패키지는 SearXNG 웹앱, 리버스 프록시(Caddy), 그리고 Valkey(리미터/레이트리밋 지원)를 기본 구성 요소로 제공합니다. 각 컴포넌트는 다음과 같은 역할을 수행합니다.

**기술적 배경 및 구성 요소 설명**

SearXNG 웹앱은 검색 요청을 받아 여러 엔진에 쿼리를 전송하고 결과를 집계합니다. 이때 외부 엔진과의 통신은 보안 및 프라이버시 보호가 필수적이므로, 리버스 프록시가 중간에서 SSL 인증서 관리와 트래픽 라우팅을 담당합니다. Caddy 프록시는 자동 인증서 발급(Let’s Encrypt 등)과 HTTP/2, WebSocket 지원, 다양한 헤더 관리 기능을 갖추고 있어, 검색 서비스의 신뢰성과 보안성을 높이는 데 중요한 역할을 합니다.

Valkey는 Redis 호환 인메모리 데이터베이스로, SearXNG의 IP rate limitation, 캐시, 세션 관리 등에 활용됩니다. Valkey를 통해 검색 요청 빈도, 봇 트래픽, 엔진별 레이트리밋 정책을

세밀하게 제어할 수 있으며, 장애 발생 시 캐시를 활용한 백오프(backoff) 전략도 적용 가능합니다.

이러한 구성은 실무에서 빠른 배포와 테스트 환경 구축에 최적화되어 있습니다. 예를 들어, 개발 환경에서는 Docker Compose로 손쉽게 여러 인스턴스를 띄워 기능을 검증할 수 있으며, 운영 환경에서는 각 컴포넌트의 설정을 세밀하게 조정하여 보안과 성능을 동시에 확보할 수 있습니다. 또한, 각 컨테이너는 독립적으로 관리되기 때문에, 문제가 발생한 경우에도 전체 서비스를 중단하지 않고 부분적으로 재시작하거나 교체할 수 있습니다.

**실무 배포 시나리오**

실제 배포 시에는 다음과 같은 절차를 따릅니다. 먼저 Docker Compose 파일을 내려받아 환경 변수와 볼륨 마운트, 네트워크 설정을 점검합니다. SearXNG 컨테이너는 기본적으로 8080 포트에서 서비스하며, Caddy 프록시가 80/443 포트로 외부 트래픽을 받아 내부로 전달합니다. 인증서 자동 갱신, 프록시 헤더(X-Forwarded-Proto, X-Forwarded-For 등) 관리, WebSocket 연결 유지 등 다양한 운영 상황에 대응할 수 있도록 프록시 설정을 튜닝해야 합니다.

Valkey는 독립 컨테이너로 배포하며, SearXNG와 연결해 rate limit 및 캐시 정책을 설정합니다. 예를 들어, IP별 검색 요청 횟수를 제한하거나, 엔진별로 캐시 만료 시간을 조정하여 외부 엔진 차단(CAPTCHA, rate-limit) 대응력을 높일 수 있습니다.

운영 환경에서는 로그 및 모니터링 도구와 연동하여 각 컨테이너의 상태를 실시간으로 확인할 수 있습니다. 장애 발생 시에는 로그를 분석하여 원인을 파악하고, 필요에 따라 컨테이너를 재시작하거나 설정을 변경하여 문제를 신속하게 해결할 수 있습니다. 또한, Docker Compose의 확장성을 활용하여 트래픽 증가에 따라 컨테이너 수를 조정하거나, 특정 컴포넌트만 별도로 스케일링하는 것도 가능합니다.

**코드 예시 및 설정 방법**

아래는 대표적인 docker-compose.yml 예시입니다.

```
yamlversion:'3'services:searxng:image:
↪  searxng/searxng:latestports:-"8080:8080"volumes:-
↪  ./settings.yml:/etc/searxng/settings.ymlenvironment:-
↪  SEARXNG_SECRET_KEY=your-secret-keydepends_on:- valkey
caddy:image: caddy:latestports:-"80:80"-"443:443"volumes:-
↪  ./Caddyfile:/etc/caddy/Caddyfile- ./certs:/etc/ssl/certsdepends_on:- searxng
valkey:image: valkey/valkey:latestports:-"6379:6379"volumes:- ./valkey-data:/data
```

이 예시에서 각 서비스는 독립적으로 정의되어 있으며, 볼륨 마운트를 통해 설정 파일과 데이터

를 영구적으로 보관할 수 있습니다. 환경 변수로는 보안에 중요한 SEARXNG_SECRET_KEY를 별도로 지정하여, 외부에 노출되지 않도록 관리하는 것이 좋습니다. 또한, depends_on 옵션을 통해 서비스 간의 기동 순서를 제어할 수 있습니다.

**주의사항 및 한계점**

Docker Compose 방식은 빠른 배포와 테스트에 적합하지만, 대규모 트래픽이나 고가용성 요구에는 한계가 있습니다. 프록시 인증서 관리, Valkey 데이터 영속성, 네트워크 분리 등은 추가적인 운영 자동화 및 보안 정책이 필요합니다. 한편, 프록시 설정이 미흡할 경우 WebSocket 연결 끊김, 인증서 만료, 헤더 누락 등 장애가 발생할 수 있으므로, 공식 문서의 예시 설정을 참고하여 환경에 맞게 튜닝하는 것이 중요합니다.

또한, Docker Compose는 단일 서버 또는 소규모 환경에 적합하므로, 대규모 서비스에서는 오케스트레이션 도구(예: Kubernetes)로의 전환을 고려해야 합니다. 데이터 백업 및 복구 전략, 모니터링 및 알림 시스템 연동, 보안 취약점 점검 등도 실무에서 반드시 검토해야 할 항목입니다. 특히, Valkey와 같은 인메모리 데이터베이스의 경우 장애 발생 시 데이터 손실 가능성이 있으므로, 정기적인 스냅샷 백업과 장애 복구 시나리오를 마련해 두는 것이 좋습니다.

### 4.1.2 설정 파일(settings.yml)에서 반드시 다뤄야 할 운영 항목

SearXNG의 핵심 운영 설정은 settings.yml 파일에서 관리됩니다. 이 파일은 검색 엔진 연동, 출력 포맷, 보안 파라미터, rate limit 등 다양한 운영 정책을 정의하는데, 실무에서는 다음 항목들을 반드시 점검해야 합니다.

**기술적 배경 및 주요 설정 항목**

settings.yml 파일은 SearXNG의 동작 방식을 결정하는 가장 중요한 설정 파일입니다. 이 파일을 통해 검색 엔진의 종류, 출력 포맷, 보안 옵션, 트래픽 제어 정책 등 다양한 운영 파라미터를 세밀하게 조정할 수 있습니다. 특히, 메타검색 엔진 특성상 외부와의 연동이 많기 때문에, 각 항목의 설정이 서비스 품질과 보안에 직접적인 영향을 미칩니다.

첫째, 출력 포맷 제어는 AI 에이전트 연계, 내부 API 호출 등에서 매우 중요합니다. SearXNG는 JSON, CSV, RSS 등 다양한 포맷을 지원하며, settings.yml에서 허용된 포맷만 응답하도록 설계되어 있습니다. 예를 들어, **`output_formats: ["html", "json"]`** 설정을 통해 JSON 활성화가 가능합니다. 공용 인스턴스에서는 보안상 JSON을 비활성화하는 경우가 많으며, 사내

인스턴스에서는 목적에 맞게 활성화하는 것이 권장됩니다.

둘째, public_instance 플래그는 공용 기능(예: API 공개, 검색 제한 등)을 토글하는 역할을 합니다. **`public_instance: false`**로 설정하면 외부 접근을 제한하고, 내부망에서만 서비스할 수 있습니다.

셋째, limiter 설정과 Valkey 의존성은 봇 차단 및 레이트리밋에 직접 연결됩니다. **`limiter:`** 섹션에서 IP별, 엔진별, 사용자별 요청 제한 정책을 정의할 수 있으며, Valkey를 연결하여 실시간 트래픽 제어와 캐시 관리가 가능합니다.

넷째, secret_key 등 기본 보안 파라미터는 CSRF 방지, 세션 암호화, API 인증 등에 사용됩니다. 배포 템플릿에서는 반드시 고유한 secret_key를 설정하고, 외부에 노출되지 않도록 관리해야 합니다.

이 외에도, 검색 엔진별 커스터마이징, 사용자 인터페이스 설정, 로깅 및 모니터링 옵션 등 다양한 항목이 포함되어 있습니다. 각 항목의 설정값은 서비스 목적과 보안 정책에 따라 달라질 수 있으므로, 운영 환경에 맞게 세밀하게 조정하는 것이 중요합니다.

**실무 적용 사례 및 체크리스트**

실제 운영에서는 다음과 같은 체크리스트를 활용합니다.

- settings.yml에서 output_formats에 JSON 포함 여부 확인
- public_instance 플래그를 환경별로 설정(내부망/외부망 구분)
- limiter 정책을 Valkey와 연동하여 IP별 rate limit 적용
- secret_key를 안전하게 생성하고, 배포 자동화 시 환경 변수로 관리
- 엔진별 커스터마이징(검색 소스 추가/제거, 파라미터 튜닝) 시 변경점 문서화

이러한 체크리스트를 통해 운영자는 서비스의 보안성과 안정성을 확보할 수 있습니다. 예를 들어, 내부망에서만 사용할 경우 public_instance를 false로 설정하여 외부 접근을 차단하고, 외부에 공개하는 경우에는 rate limit을 강화하여 봇 트래픽을 효과적으로 차단할 수 있습니다. 또한, secret_key는 반드시 고유한 값을 사용하고, 환경 변수로 관리하여 소스 코드나 설정 파일에 노출되지 않도록 해야 합니다.

운영 중에는 설정 변경 이력을 주기적으로 기록하고, 변경 사항이 서비스에 미치는 영향을 모니터링하는 것이 중요합니다. 예를 들어, output_formats를 변경할 경우, 기존에 연동된 시스템이나

클라이언트에 영향이 없는지 사전에 검토해야 하며, limiter 정책을 조정할 때는 트래픽 패턴을 분석하여 적절한 한도를 설정해야 합니다.

**설정 예시**

```
yamloutput_formats:- html- json
public_instance:false
limiter:ip:limit:100window:60engine:google:limit:20window:60
secret_key:"your-very-secure-secret"
```

이 예시에서는 HTML과 JSON 두 가지 출력 포맷을 허용하고, public_instance를 false로 설정하여 외부 접근을 제한하고 있습니다. limiter 항목에서는 IP별로 1분에 100회, Google 엔진에 대해서는 1분에 20회로 요청을 제한하고 있습니다. secret_key는 반드시 고유하고 예측 불가능한 값을 사용해야 하며, 운영 환경에서는 환경 변수로 관리하는 것이 안전합니다.

**한계점 및 대안 비교**

settings.yml은 유연한 설정이 가능하지만, 복잡한 정책이나 대규모 환경에서는 중앙 관리 및 버전 관리가 어려울 수 있습니다. GitOps, CI/CD와 연계하여 설정 변경을 자동화하거나, 환경별 템플릿을 활용하는 것이 실무에서 권장되는 대안입니다. 또한, 보안 파라미터 노출 방지와 설정 변경 이력 관리에 주의해야 합니다.

대규모 환경에서는 설정 파일의 일관성을 유지하기 위해 버전 관리 시스템(Git 등)을 활용하고, CI/CD 파이프라인을 통해 변경 사항을 자동으로 배포하는 것이 효과적입니다. 또한, 환경별로 별도의 설정 파일을 관리하거나, 템플릿 엔진을 활용하여 공통 설정과 차별화된 설정을 분리하는 것도 좋은 방법입니다. 보안 측면에서는 secret_key와 같은 민감 정보는 별도의 시크릿 관리 시스템(Kubernetes Secrets, HashiCorp Vault 등)에 저장하고, 애플리케이션에서는 환경 변수나 시크릿 마운트를 통해 참조하는 것이 안전합니다.

---

## 4.2 쿠버네티스 운영 포인트(클라우드 네이티브 관점)

클라우드 네이티브 환경에서는 SearXNG의 배포와 운영이 더욱 복잡해집니다. 컨테이너 오케스트레이션, 네트워크 정책, 스토리지 관리, 관측성 등 다양한 요소가 결합되어야 하며, 특히 엔진별 egress 통제와 장애 대응 전략이 중요합니다. 쿠버네티스에서는 Deployment와 StatefulSet을

분리하여 운영 데이터와 설정을 안전하게 관리하고, 롤링 업데이트와 HPA(자동 확장) 등 고가용성 패턴을 적용할 수 있습니다. 이 절에서는 쿠버네티스 환경에서 SearXNG를 효율적으로 배포하고, 네트워크 및 보안 정책을 설계하는 실무 포인트를 상세히 다룹니다.

### 4.2.1 배포 단위: Deployment/Stateful 구성 분리

쿠버네티스에서 SearXNG를 배포할 때, 웹앱과 운영 데이터, 리미터/캐시(Valkey)를 분리하여 관리하는 것이 표준 패턴입니다. 웹앱은 stateless로 Deployment로 배포하고, 설정 파일과 Valkey는 Persistent Volume(PV)과 StatefulSet으로 분리하여 데이터의 영속성과 장애 복구력을 높입니다.

**기술적 배경과 배포 구조**

Deployment는 SearXNG 웹앱의 확장성과 장애 복구를 담당합니다. 여러 파드를 생성하여 트래픽을 분산시키고, 롤링 업데이트를 통해 무중단 배포가 가능합니다. 반면, Valkey와 설정 파일은 데이터 영속성이 필요하므로 StatefulSet과 PV를 활용합니다. StatefulSet은 파드 순서와 이름을 보장하며, 데이터 볼륨을 각 파드에 할당하여 장애 발생 시에도 데이터가 유지됩니다.

이러한 구조는 대규모 트래픽 처리와 장애 복구에 매우 효과적입니다. 예를 들어, SearXNG 웹앱을 여러 파드로 확장하면 트래픽이 자동으로 분산되어 서비스의 안정성이 높아집니다. 또한, 롤링 업데이트를 통해 새로운 버전의 SearXNG를 무중단으로 배포할 수 있으며, 문제가 발생한 경우에는 이전 버전으로 신속하게 롤백할 수 있습니다.

Valkey와 같은 상태 저장 컴포넌트는 StatefulSet으로 관리하여 데이터의 일관성과 영속성을 보장합니다. Persistent Volume을 활용하면 파드가 재시작되거나 이동하더라도 데이터가 유지되므로, 장애 발생 시에도 빠르게 복구할 수 있습니다. 설정 파일(settings.yml)은 ConfigMap이나 PV에 저장하여, 파드가 재시작되더라도 최신 설정이 유지되도록 해야 합니다.

**실무 운영 시나리오**

예를 들어, SearXNG 웹앱은 Deployment로 3개의 파드를 운영하고, Valkey는 StatefulSet으로 1개의 파드를 배포합니다. 설정 파일(settings.yml)은 ConfigMap이나 PV에 저장하여 파드 재시작 시에도 변경 내용이 유지됩니다. 장애 발생 시에는 StatefulSet 파드가 자동으로 복구되고, Deployment 파드는 HPA(Horizontal Pod Autoscaler)를 통해 트래픽 변화에 따라 자동 확장됩니다.

관측성(Observability) 측면에서는 로그와 메트릭을 수집하여 장애 원인 분석과 성능 튜닝에 활용합니다. Prometheus, Grafana 등 오픈소스 도구와 연계하여 검색 트래픽, 엔진 응답 속도, rate limit 현황 등을 모니터링할 수 있습니다.

운영 환경에서는 롤링 업데이트 전략을 활용하여 서비스 중단 없이 새로운 버전을 배포할 수 있습니다. 예를 들어, 새로운 기능이나 보안 패치가 적용된 이미지를 배포할 때, 기존 파드를 순차적으로 교체하여 서비스의 가용성을 유지할 수 있습니다. 또한, HPA를 통해 트래픽 증가에 자동으로 대응할 수 있으므로, 예기치 않은 트래픽 급증에도 안정적으로 서비스를 제공할 수 있습니다.

**코드 예시 및 설정 방법**

아래는 대표적인 쿠버네티스 배포 예시입니다.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: searxng
spec:
  replicas: 3
  selector:
    matchLabels:
      app: searxng
  template:
    metadata:
      labels:
        app: searxng
    spec:
      containers:
        - name: searxng
          image: searxng/searxng:latest
          ports:
            - containerPort: 8080
          volumeMounts:
            - name: settings
              mountPath: /etc/searxng/settings.yml
              subPath: settings.yml
      volumes:
        - name: settings
          configMap:
            name: searxng-settings
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: valkey
spec:
  serviceName: valkey
```

```
  replicas: 1
  selector:
    matchLabels:
      app: valkey
  template:
    metadata:
      labels:
        app: valkey
    spec:
      containers:
        - name: valkey
          image: valkey/valkey:latest
          ports:
            - containerPort: 6379
          volumeMounts:
            - name: data
              mountPath: /data
      volumes:
        - name: data
          persistentVolumeClaim:
            claimName: valkey-pvc
```

이 예시에서는 SearXNG 웹앱을 3개의 파드로 배포하고, 설정 파일은 ConfigMap을 통해 마운트하고 있습니다. Valkey는 StatefulSet으로 1개의 파드를 운영하며, 데이터는 Persistent Volume Claim을 통해 영구적으로 저장됩니다. 이러한 구조는 장애 발생 시에도 데이터 손실 없이 빠르게 복구할 수 있도록 도와줍니다.

**주의사항 및 한계점**

Deployment와 StatefulSet을 분리하면 데이터 무결성과 확장성이 높아지지만, 설정 변경 시 파드 재시작, PV 관리, 롤링 업데이트 전략 등 추가적인 운영 자동화가 필요합니다. HPA 설정이 미흡하면 트래픽 급증 시 서비스가 느려질 수 있으며, 관측성 도구 연계가 필수적입니다. 롤링 업데이트 시에는 파드 순서와 데이터 동기화에 주의해야 하며, 장애 복구 테스트를 정기적으로 수행하는 것이 권장됩니다.

또한, Persistent Volume의 용량 관리와 백업 정책을 사전에 마련해야 하며, 설정 파일의 변경 이력을 체계적으로 관리하는 것도 중요합니다. 대규모 환경에서는 여러 팀이 동시에 설정을 변경할 수 있으므로, GitOps와 같은 자동화 도구를 활용하여 변경 사항을 추적하고, 충돌을 방지하는 것이 좋습니다. 마지막으로, 관측성 도구와 연계하여 실시간으로 서비스 상태를 모니터링하고, 장애 발생 시 신속하게 대응할 수 있는 체계를 구축해야 합니다.

### 4.2.2 egress 통제와 엔진별 네트워크 정책

메타검색 엔진은 본질적으로 외부 검색 엔진과 통신하기 때문에, egress(외부 출구) 네트워크 정책 설계가 매우 중요합니다. 조직 보안 정책에 따라 엔진별 도메인 허용, 프록시 경유, 보안 장비 연동 등 다양한 통제 모델이 적용됩니다.

**기술적 배경 및 설계 원칙**

egress allowlist는 SearXNG가 외부 엔진에 요청을 보낼 때 허용된 도메인만 접근할 수 있도록 제어합니다. 쿠버네티스 네트워크 정책(NetworkPolicy)을 활용하여 SearXNG 파드에서 특정 엔진 도메인(IP, 포트)만 허용하고, 나머지 트래픽은 차단할 수 있습니다. 이는 데이터 반출 통제와 보안 감사에 필수적인 설계 원칙입니다.

프록시 경유는 조직 내부 프록시 서버를 통해 외부 엔진에 접근하는 방식입니다. 프록시 서버에서 트래픽 로깅, SSL 인증서 관리, IP masking 등을 수행하여 보안성을 높일 수 있습니다. 또한, 조직 보안 장비(IDS, IPS, 방화벽 등)와 연동하여 검색 트래픽을 실시간 모니터링하고, 이상 징후를 탐지할 수 있습니다.

이러한 네트워크 정책은 서비스의 보안성과 신뢰성을 높이는 데 중요한 역할을 합니다. 예를 들어, 외부 엔진 도메인만 허용하면, 내부 데이터가 외부로 유출되는 것을 방지할 수 있으며, 프록시 서버를 통해 트래픽을 통제하면, 검색 트래픽의 흐름을 중앙에서 관리할 수 있습니다. 또한, 보안 장비와 연계하여 이상 트래픽을 실시간으로 탐지하고, 필요 시 자동으로 차단할 수 있습니다.

**실무 적용 시나리오**

예를 들어, SearXNG 파드에서 Google, Bing, DuckDuckGo 등 주요 엔진 도메인만 egress allowlist에 등록하고, 나머지 도메인은 차단합니다. 프록시 서버를 설정하여 모든 외부 요청이 프록시를 경유하도록 하고, 프록시 로그를 통해 검색 트래픽을 감사합니다. 보안 장비와 연계하여 트래픽 패턴 분석, 차단 정책 자동화 등을 구현할 수 있습니다.

실제 운영 환경에서는 네트워크 정책을 주기적으로 점검하고, 엔진 도메인 변경이나 프록시 서버 장애에 대비한 대체 경로를 마련해야 합니다. 예를 들어, 엔진 도메인이 변경될 경우 allowlist를 즉시 갱신하고, 프록시 서버 장애 시에는 자동으로 대체 프록시를 사용하도록 설정할 수 있습니다. 또한, 보안 장비와 연계하여 이상 트래픽이 감지되면 즉시 알림을 받고, 필요 시 자동으로 차단 정책을 적용할 수 있습니다.

**네트워크 정책 예시**

```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: searxng-egress
spec:
  podSelector:
    matchLabels:
      app: searxng
  policyTypes:
    - Egress
  egress:
    - to:
        - ipBlock:
            cidr: 8.8.8.8/32   # Google
        - ipBlock:
            cidr: 13.107.21.200/32  # Bing
      ports:
        - protocol: TCP
          port: 443
```

이 예시에서는 SearXNG 파드에서 Google과 Bing 엔진의 IP만 허용하고, 443 포트(TCP)로만 트래픽을 허용하고 있습니다. 실제 환경에서는 엔진별로 IP나 도메인이 자주 변경될 수 있으므로, 정책을 주기적으로 점검하고 갱신하는 것이 중요합니다. 또한, 프록시 서버를 경유하는 경우에는 프록시 서버의 IP만 허용하고, 나머지 외부 트래픽은 모두 차단할 수 있습니다.

**한계점 및 대안 비교**

egress 정책은 보안성을 높이지만, 엔진 도메인 변경, 프록시 장애, 네트워크 지연 등 운영 리스크가 있습니다. 엔진별 정책 변경에 따라 allowlist를 주기적으로 갱신해야 하며, 프록시 서버 장애 시 검색 서비스 전체가 영향을 받을 수 있습니다. 대안으로는 엔진별 fallback 정책, 프록시 이중화, 네트워크 모니터링 자동화 등이 있습니다.

또한, 네트워크 정책이 너무 엄격하면 정상적인 검색 트래픽도 차단될 수 있으므로, 서비스 품질과 보안성의 균형을 유지하는 것이 중요합니다. 엔진별로 fallback 정책을 마련하여, 특정 엔진이 차단되거나 장애가 발생할 경우 자동으로 대체 엔진을 사용하도록 설정할 수 있습니다. 프록시 서버는 이중화하여 장애 발생 시에도 서비스가 중단되지 않도록 해야 하며, 네트워크 모니터링 도구를 활용하여 트래픽 패턴을 실시간으로 분석하고, 이상 징후를 신속하게 탐지할 수 있도록 해야 합니다.

## 4.3 장애/차단 대응(운영자가 실제로 겪는 문제 중심)

SearXNG 운영에서 가장 빈번하게 발생하는 장애는 외부 엔진의 CAPTCHA, rate limit, IP 차단 등입니다. 메타검색 구조 특성상 서버가 대행 호출을 하므로 봇으로 분류되어 차단되는 경우가 많습니다. 이에 대한 대응책은 리미터 튜닝, 엔진 믹스 조정, 캐시/백오프, 프록시/출구 IP 전략 등 다양한 실무적 방법이 필요합니다. 본 절에서는 장애 발생 원인과 대응 방법을 실제 운영 경험을 바탕으로 상세히 설명합니다.

### 4.3.1 CAPTCHA/차단을 유발하는 원인과 리미터의 의미

SearXNG와 같은 메타검색 엔진을 운영할 때 가장 큰 고민 중 하나는 외부 검색 엔진의 자동화 트래픽 차단 정책입니다. 특히, 구글이나 빙과 같은 대형 검색 엔진은 비정상적으로 많은 요청이 감지될 경우 봇으로 간주하여 CAPTCHA를 반환하거나, 아예 IP를 차단해버리는 경우가 많습니다. 이러한 현상은 단일 서버에서 여러 사용자의 검색 요청을 대행 처리하는 메타검색 구조의 특성상 더욱 빈번하게 발생합니다. 따라서, 운영자는 장애의 근본 원인을 이해하고, 효과적인 대응책을 마련해야 합니다.

**기술적 배경과 장애 원인**

SearXNG는 여러 검색 엔진에 대량의 요청을 보내는 구조이기 때문에, 외부 엔진에서는 이를 봇 트래픽으로 인식하여 CAPTCHA, rate limit, IP 차단 등의 보호 조치를 취합니다. 특히 Google, Bing 등은 자동화된 트래픽에 민감하게 반응하며, 일정 횟수 이상 요청이 감지되면 CAPTCHA 화면을 반환하거나 IP를 일시적으로 차단합니다.

리미터(rate limiter)는 이러한 장애를 예방하기 위한 핵심 설계 요소입니다. IP별, 엔진별, 사용자별 요청 빈도를 실시간으로 모니터링하고, 설정된 한도를 초과하면 자동으로 요청을 차단하거나 백오프(backoff) 전략을 적용합니다. SearXNG는 Valkey와 연동하여 리미터 정책을 구현하며, settings.yml에서 세부 정책을 정의할 수 있습니다.

이러한 리미터 정책은 단순히 요청을 차단하는 것을 넘어, 서비스의 품질과 안정성을 유지하는 데 중요한 역할을 합니다. 예를 들어, 갑작스러운 트래픽 급증이나 악의적인 봇 공격이 발생할 경우, 리미터가 이를 신속하게 감지하여 서비스 전체의 장애를 방지할 수 있습니다. 또한, 엔진별로 별도의 한도를 설정함으로써, 특정 엔진에 과도한 부하가 걸리는 것을 방지할 수 있습니다.

**실무 장애 사례 및 대응 전략**

실제 운영에서는 다음과 같은 장애가 빈번하게 발생합니다. 예를 들어, 사내 AI 에이전트가 SearXNG를 통해 Google 검색을 반복적으로 호출할 경우, 일정 시간 내에 100회 이상 요청이 발생하면 Google에서 CAPTCHA를 반환하거나 IP를 차단합니다. 이때 리미터 정책이 미흡하면 전체 검색 서비스가 중단될 수 있습니다.

대응책으로는 리미터 튜닝(요청 한도 조정, 윈도우 시간 변경), 엔진 믹스 조정(다양한 엔진 활용, 장애 발생 시 fallback 엔진 사용), 캐시/백오프(검색 결과 캐시 활용, 장애 발생 시 일정 시간 대기 후 재시도), 프록시/출구 IP 전략(IP 분산, 프록시 이중화) 등이 있습니다.

예를 들어, 리미터 정책을 강화하여 Google 엔진에 대한 요청 한도를 낮추고, Bing이나 DuckDuckGo와 같은 대체 엔진을 함께 활용하면, 특정 엔진이 차단되더라도 서비스 전체의 가용성을 유지할 수 있습니다. 또한, 검색 결과를 일정 시간 동안 캐시에 저장하여 동일한 요청이 반복될 경우 외부 엔진에 불필요한 트래픽을 발생시키지 않도록 할 수 있습니다. 장애 발생 시에는 일정 시간 동안 해당 엔진에 대한 요청을 중단하고, 이후에 재시도하는 백오프 전략을 적용할 수 있습니다.

프록시/출구 IP 전략은 여러 프록시 서버를 활용하여 IP 분산을 구현하거나, 장애 발생 시 대체 IP로 전환하는 방식입니다. 예를 들어, 프록시 서버를 두 개 이상 운영하여 트래픽을 분산시키고, 장애 발생 시 자동으로 대체 프록시를 사용하도록 설정할 수 있습니다.

**설정 예시 및 코드**

settings.yml에서 리미터 정책 예시:

```
limiter:
  ip:
    limit: 50
    window: 60

engine:
  google:
    limit: 10
    window: 60
  bing:
    limit: 20
    window: 60
```

이 예시에서는 IP별로 1분에 50회, Google 엔진에 대해서는 1분에 10회, Bing 엔진에 대해서는 1분에 20회로 요청을 제한하고 있습니다. 이러한 정책을 통해 특정 엔진에 과도한 트래픽이

발생하는 것을 방지할 수 있습니다.

프록시/출구 IP 전략을 적용할 때는 여러 프록시 서버를 운영하고, 장애 발생 시 자동으로 대체 프록시를 사용하도록 설정할 수 있습니다. 예를 들어, HAProxy나 Nginx와 같은 로드밸런서를 활용하여 트래픽을 여러 프록시 서버로 분산시키고, 장애 발생 시 자동으로 대체 경로를 선택할 수 있습니다.

**한계점 및 대안 비교**

CAPTCHA/차단 대응은 완벽한 해결책이 없으며, 엔진별 정책 변경에 따라 지속적인 튜닝이 필요합니다. 리미터 정책이 너무 엄격하면 서비스 품질이 저하되고, 너무 느슨하면 장애가 빈번하게 발생할 수 있습니다. 대안으로는 엔진별 fallback, 캐시 활용, 프록시 이중화, 장애 모니터링 자동화 등이 있습니다. 또한, 장애 발생 시 사용자에게 명확한 안내 메시지를 제공하고, 로그/메트릭을 통해 원인 분석을 신속하게 수행하는 것이 중요합니다.

운영자는 장애 발생 시 로그와 메트릭을 실시간으로 모니터링하여 원인을 신속하게 파악하고, 필요 시 리미터 정책을 조정하거나, 엔진 믹스를 변경하여 서비스의 가용성을 유지해야 합니다. 또한, 사용자에게는 장애 상황을 명확하게 안내하여 혼란을 최소화하고, 장애 원인과 대응 결과를 문서화하여 향후 유사한 문제가 발생했을 때 신속하게 대응할 수 있도록 해야 합니다.

마지막으로, 엔진별 정책 변경이나 프록시 서버 장애 등 예기치 않은 상황에 대비하여, 정기적으로 장애 복구 시나리오를 점검하고, 운영 자동화 도구를 활용하여 반복적인 작업을 최소화하는 것이 중요합니다. 이를 통해 SearXNG 서비스의 안정성과 신뢰성을 지속적으로 높일 수 있습니다.

# 5장. 적용 사례: AI Agent/LLM 스택에서 SearXNG를 어떻게 쓰는가(구체 사례 중심)

## 5.1 Flowise 연계: “에이전트 빌더에서 바로 쓰는 검색 도구”

Flowise는 최근 AI 에이전트 빌더 및 워크플로우 자동화 도구로 각광받고 있으며, 다양한 LLM과 검색 엔진을 조합해 실시간 정보 수집과 RAG(Retrieval-Augmented Generation) 파이프라인을 구축하는 데 활용되고 있습니다. 특히 SearXNG와의 연계는 사내 온프레미스 환경에서 프라이버시, 보안, 비용 효율성을 극대화할 수 있는 실무적 장점이 큽니다. Flowise는 캔버스 방식의

UI를 제공하여, 검색 도구(SearXNG Tool Node)를 손쉽게 배치하고, 내부망에서 직접 운영하는 SearXNG 인스턴스와 연결할 수 있습니다. 이때, 검색 결과를 JSON 포맷으로 받아야 하므로 SearXNG의 settings.yml에서 JSON 출력 허용을 반드시 활성화해야 하며, 네트워크 정책을 통해 외부 API 접근 및 데이터 반출을 통제할 수 있습니다. 이러한 연계 구조는 단순한 검색 기능을 넘어, 최신 정보의 신속한 획득과 내부 데이터베이스와의 결합을 통한 지능형 에이전트 운영에 필수적인 인프라로 자리잡고 있습니다. 본 절에서는 Flowise와 SearXNG의 실무적 결합 방식과, 실제 운영에 필요한 체크리스트를 구체적으로 다루겠습니다.

### 5.1.1 Flowise의 SearXNG Tool 노드 사용 방식

Flowise에서 SearXNG Tool 노드를 사용하는 방식은 공식 문서와 커뮤니티 사례에서 일관되게 안내되고 있습니다. Flowise 캔버스에 SearXNG Tool Node를 추가한 뒤, Base URL을 내부망에서 운영 중인 SearXNG 인스턴스 주소(예: **`http://localhost:8080`**)로 지정하는 것이 기본입니다. 이때, SearXNG의 settings.yml 파일에서 JSON 출력 포맷을 활성화해야 하며, 이는 **`output_format`** 항목에 **`json`**을 추가하는 방식으로 설정됩니다. 만약 JSON 포맷이 허용되지 않았다면, Flowise에서 검색 결과를 제대로 받아올 수 없고, HTTP 403 오류가 발생할 수 있으므로 반드시 확인이 필요합니다.

실제 운영 환경에서는 다음과 같은 체크리스트가 중요합니다.

1. **사내 인스턴스 URL**: 외부망이 아닌 내부망에서 SearXNG를 운영하여, 검색 요청이 외부로 노출되지 않도록 합니다. 이는 프라이버시 보호와 보안 감사에 필수적입니다.
2. **JSON 출력 허용**: settings.yml에서 **`output_format`**에 **`json`**을 명시하고, 필요에 따라 CSV, RSS 등도 추가할 수 있습니다. JSON 포맷은 에이전트 연계에 가장 적합하며, Flowise Tool Node와의 호환성이 높습니다.
3. **네트워크 정책**: 검색 엔진별 egress allowlist를 설정하여, SearXNG가 외부 검색 엔진에 접근할 때 조직의 보안 정책에 맞는 도메인만 허용합니다. 프록시 경유나 보안 장비 연동도 고려해야 하며, 내부망에서 운영 시 데이터 반출 경로를 명확히 통제해야 합니다.
4. **API Rate Limit 및 봇 차단 대응**: Flowise에서 반복적으로 검색 요청을 보내는 경우, SearXNG의 limiter와 Valkey 연동을 통해 rate limit을 설정하고, 엔진별 봇 차단 정책에 대응할 수 있도록 합니다.

5. **운영 로그 및 감사**: 검색 요청 및 응답에 대한 로그를 별도로 관리하여, 보안 감사 및 장애 대응에 활용할 수 있도록 합니다.

실무에서는 Flowise와 SearXNG 연계가 단순히 검색 결과를 받아오는 수준을 넘어, 검색 결과의 신뢰성과 최신성, 그리고 내부 정책에 맞는 데이터 활용을 보장하는 구조로 설계되어야 합니다. 예를 들어, 사내망에서만 접근 가능한 SearXNG 인스턴스를 운영하면 외부 API 비용, 정책 변경, 데이터 반출 리스크를 근본적으로 해소할 수 있습니다. 또한, Flowise의 워크플로우 내에서 SearXNG Tool Node를 활용하면, 다양한 LLM과 검색 엔진을 조합한 맞춤형 에이전트 구축이 가능해집니다. 이처럼, SearXNG와 Flowise의 결합은 AI 에이전트 운영의 실무적 요구를 충족시키는 핵심 인프라로 자리잡고 있습니다.

Flowise의 SearXNG Tool 노드 사용은 실제 현업에서 다양한 방식으로 확장되고 있습니다. 예를 들어, 사내 정보보안 정책에 따라 SearXNG 인스턴스에 접근 가능한 IP 대역을 제한하거나, 검색 결과에 포함되는 도메인을 화이트리스트로 관리하는 사례가 있습니다. 또한, Flowise 내에서 SearXNG Tool Node를 여러 개 배치하여, 각기 다른 검색 엔진 조합이나 카테고리별로 검색을 병렬 처리하는 방식도 활용됩니다. 이때, 각 노드의 설정을 세분화하여 특정 업무 목적(예: 기술 뉴스, 법률 정보, 사내 문서 등)에 맞는 맞춤형 검색 파이프라인을 구축할 수 있습니다.

운영 중에는 SearXNG의 엔진별 장애나 검색 차단(CAPTCHA 등)에 대비해, Flowise 워크플로우 내에서 예외 처리를 구현하거나, 검색 실패 시 대체 엔진을 자동으로 호출하는 로직을 추가하는 것이 권장됩니다. 또한, Flowise의 로그 관리 기능과 SearXNG의 감사 로그를 연동하여, 검색 요청 이력과 결과 활용 내역을 체계적으로 관리할 수 있습니다.

이처럼 Flowise와 SearXNG의 결합은 단순한 검색 도구 연계를 넘어, 실시간 정보 수집, 보안 정책 준수, 감사 대응 등 다양한 실무적 요구를 충족하는 유연한 인프라로 발전하고 있습니다. 실제로 많은 기업과 기관에서 Flowise 기반의 AI 에이전트 구축 시 SearXNG를 핵심 검색 게이트웨이로 채택하고 있으며, 내부 데이터와 외부 정보를 통합적으로 활용하는 지능형 워크플로우를 구현하고 있습니다.

### 5.1.2 Flowise 기반 RAG/Agent 운영 시 권장 패턴

Flowise와 SearXNG를 기반으로 한 RAG/Agent 운영 패턴은 최근 AI 생태계에서 표준화된 접근 방식으로 자리잡고 있습니다. 이 패턴은 크게 네 단계로 구성됩니다.

1. **SearXNG로 1차 탐색(출처 후보 수집)**

   에이전트가 사용자의 질문이나 명령을 받아, SearXNG를 통해 관련 키워드 검색을 수행합니다. 이때, 다양한 검색 엔진의 결과를 취합하여, 최신 뉴스, 공식 문서, 블로그 등 신뢰할 수 있는 출처 후보를 빠르게 수집할 수 있습니다. SearXNG의 메타검색 구조 덕분에 엔진 장애나 차단에 대한 리스크가 분산되고, 결과의 다양성과 신뢰성이 확보됩니다.

2. **크롤링/정제(본문 추출)**

   수집된 링크 목록을 바탕으로, Flowise 내 크롤러 또는 별도의 웹 스크래핑 도구를 활용하여 본문 데이터를 추출합니다. 이 과정에서 불필요한 광고, 네비게이션, 댓글 등 비정형 데이터를 제거하고, 핵심 텍스트만 정제하여 구조화된 데이터로 변환합니다. 실무에서는 BeautifulSoup, Selenium 등 Python 기반 크롤링 라이브러리가 활용되며, Flowise의 플러그인 또는 커스텀 노드로 연동이 가능합니다.

3. **임베딩/벡터DB 적재(RAG)**

   정제된 본문 데이터를 임베딩 모델을 통해 벡터화하고, 벡터DB(예: Qdrant, Pinecone, Milvus 등)에 적재합니다. 이 단계에서는 텍스트의 의미 기반 검색이 가능해지며, 기존 키워드 검색의 한계를 극복할 수 있습니다. Flowise는 LangChain과 연동하여 다양한 임베딩 모델과 벡터DB를 지원하며, 검색 결과와 임베딩 데이터의 연결고리를 유지할 수 있습니다.

4. **답변 생성 + 근거 링크 유지**

   LLM이 벡터DB에서 의미 기반으로 검색된 데이터를 바탕으로 답변을 생성합니다. 이때, 답변에 근거가 되는 링크와 출처 정보를 함께 제공하여, Explainability와 Traceability를 보장합니다. 실무에서는 답변 템플릿에 **[출처: URL]** 형식으로 링크를 삽입하거나, Flowise 내에서 답변과 근거를 명확히 구분하는 방식이 권장됩니다.

이 패턴의 장점은 검색→수집→정제→임베딩→RAG→답변 생성까지 일관된 파이프라인을 구축할 수 있다는 점입니다. 특히, SearXNG를 통한 외부 정보 획득과 내부 벡터DB의 결합은 최신성과 신뢰성을 동시에 확보할 수 있습니다. 또한, 검색 결과의 근거를 명확히 남기므로, AI 답변의 Explainability와 감사 대응이 용이합니다.

실무에서는 다음과 같은 주의사항이 있습니다.

- SearXNG 엔진 장애나 차단(CAPTCHA 등)에 대비해 엔진 믹스 조정, 캐시 활용, 프록시 경유 등을 병행해야 합니다.
- 크롤링 과정에서 robots.txt, 저작권 정책, 데이터 반출 규정을 반드시 확인해야 하며, 사내망에서만 운영하는 경우에도 내부 감사 정책을 준수해야 합니다.
- 임베딩 모델 선택과 벡터DB 구조에 따라 검색 성능과 비용이 크게 달라질 수 있으므로, 목적에 맞는 아키텍처를 설계해야 합니다.

이처럼 Flowise 기반 RAG/Agent 운영은 SearXNG를 검색 도구로 활용하는 표준적인 파이프라인을 통해, 최신 정보의 신속한 획득과 의미 기반 검색, 그리고 근거 중심 답변 생성을 실현할 수 있습니다.

Flowise 기반 RAG/Agent 운영은 실제 현업에서 다양한 확장 사례로 나타나고 있습니다. 예를 들어, 금융기관에서는 SearXNG를 통해 실시간 금융 뉴스와 공시 정보를 수집하고, Flowise의 워크플로우를 통해 해당 정보를 벡터DB에 적재한 후, LLM이 의미 기반으로 분석 및 요약하여 투자 리포트 초안을 자동으로 생성하는 사례가 있습니다.

또한, 대기업의 IT 부서에서는 사내 정책상 외부 API 사용이 제한된 환경에서, SearXNG를 내부망에 배포하고 Flowise와 연동하여, 최신 기술 동향이나 보안 이슈를 신속하게 파악하는 데 활용하고 있습니다. 이때, 검색 결과의 근거 링크와 출처 정보를 체계적으로 관리하여, 내부 보고서나 정책 문서에 활용할 수 있도록 설계합니다.

Flowise의 플러그인 생태계를 활용하면, 크롤링, 임베딩, 벡터DB 적재, 답변 생성 등 각 단계를 세분화하여 맞춤형 파이프라인을 구축할 수 있습니다. 예를 들어, 특정 업무 목적에 따라 크롤링 대상 도메인이나 임베딩 모델을 다르게 설정하거나, 답변 생성 시 근거 링크의 신뢰도를 평가하여 우선순위를 조정하는 로직을 추가할 수 있습니다.

이처럼 Flowise와 SearXNG를 결합한 RAG/Agent 운영 패턴은, 최신 정보의 신속한 획득, 의미 기반 검색, 근거 중심 답변 생성 등 AI 에이전트의 실무적 요구를 충족시키는 표준 아키텍처로 자리잡고 있으며, 다양한 산업 분야에서 빠르게 확산되고 있습니다.

## 5.2 LangChain 연계: 표준 Search API로 생태계에 붙는 방식

LangChain은 AI 에이전트 및 RAG 파이프라인 구축을 위한 대표적인 오픈소스 프레임워크로, 다양한 검색 엔진과 데이터베이스를 표준 API로 연동할 수 있는 장점을 갖고 있습니다. SearXNG와의 통합은 LangChain의 Search API를 통해 구현되며, 이는 JSON 포맷을 활용한 검색 결과 수집과, engines/categories 파라미터를 통한 맞춤형 검색이 가능합니다. 최근 공식 문서와 커뮤니티 사례에서는 SearXNG 연동을 통해 외부 정보 획득의 신뢰성과 최신성을 확보하고, 내부 벡터DB와의 결합을 통한 의미 기반 검색을 실현하는 방식이 표준으로 자리잡고 있습니다. 본 절에서는 LangChain과 SearXNG의 통합 포인트, 설정 방법, 그리고 실무적 활용 시나리오를 구체적으로 설명합니다.

### 5.2.1 LangChain의 SearxNG 통합 포인트(설정/파라미터)

LangChain에서 SearXNG Search API를 연동하는 방법은 공식 문서에 명확히 안내되어 있습니다. 기본적으로 LangChain의 Search API 모듈에서 SearXNG 인스턴스의 Base URL을 지정하고, 검색 결과를 JSON 포맷으로 받아오는 구조입니다. 이때, SearXNG의 settings.yml에서 **`output_format`**에 **`json`**을 활성화해야 하며, 사내 인스턴스에서는 이를 기본값으로 설정하는 것이 권장됩니다.

실무적으로 중요한 설정 포인트는 다음과 같습니다.

1. **Base URL 설정**

   LangChain의 Search API에서 SearXNG 인스턴스의 URL을 입력합니다. 예를 들어, **`http://localhost:8080`** 또는 사내망에서 운영하는 별도의 도메인을 사용할 수 있습니다.

2. **JSON 출력 활성화**

   SearXNG의 settings.yml에서 **`output_format: ["json"]`**을 명시해야 하며, 이는 LangChain이 검색 결과를 파싱하는 데 필수적인 조건입니다. 만약 JSON이 비활성화되어 있으면, LangChain에서 검색 결과를 받아올 수 없으므로 반드시 확인이 필요합니다.

3. **engines/categories 파라미터 활용**

LangChain에서는 검색 요청 시 engines(검색 엔진)과 categories(카테고리) 파라미터를 동적으로 지정할 수 있습니다. 예를 들어, 뉴스, 공식 문서, 블로그 등 특정 카테고리만 검색하거나, Google, Bing, DuckDuckGo 등 특정 엔진만 선택할 수 있습니다. 실무에서는 에이전트가 상황에 따라 파라미터를 조정하여, 장애 대응, 신뢰성 확보, 최신성 강화 등 다양한 목적에 맞는 검색을 수행합니다.

4. **API Key 및 인증**

   사내망에서 운영하는 SearXNG 인스턴스에는 인증 토큰이나 API Key를 설정할 수 있으며, LangChain에서 이를 헤더에 포함하여 안전하게 검색 요청을 보낼 수 있습니다.

5. **검색 결과 파싱 및 후처리**

   LangChain은 SearXNG로부터 받은 JSON 결과를 파싱하여, 링크, 제목, 본문 요약 등 필요한 정보를 추출합니다. 이후, 크롤링, 임베딩, 벡터DB 적재 등 후처리 파이프라인을 자동화할 수 있습니다.

실무 시나리오 예시:

- 기관 내부에서 최신 법령이나 규정 검색이 필요할 때, SearXNG의 법률/정부 엔진만 선택하여 검색 결과를 수집하고, LangChain을 통해 의미 기반 질의응답을 구현할 수 있습니다.
- 장애 대응을 위해 여러 엔진을 동시에 지정하여, 한 엔진이 차단되거나 장애가 발생해도 검색 결과의 신뢰성을 유지할 수 있습니다.
- 특정 카테고리(예: 뉴스, 논문, 기술 블로그)만 검색하여, 에이전트가 목적에 맞는 정보만 활용하도록 제어할 수 있습니다.

주의사항:

- SearXNG 엔진별 rate limit, CAPTCHA, 봇 차단 정책에 따라 검색 결과가 불안정해질 수 있으므로, 엔진 믹스 조정과 캐시 활용이 필요합니다.
- 검색 결과의 신뢰성, 최신성, 근거 링크 제공을 LangChain 파이프라인에서 반드시 검증해야 하며, 감사 대응을 위한 로그 관리도 병행해야 합니다.

이처럼 LangChain과 SearXNG의 통합은 AI 에이전트 및 RAG 파이프라인에서 외부 정보 획득의 표준 인터페이스로 자리잡고 있으며, 맞춤형 검색과 의미 기반 질의응답을 실현하는 핵심 기술로 평가받고 있습니다.

LangChain과 SearXNG의 통합은 실제 현업에서 다양한 방식으로 활용되고 있습니다. 예를 들어, 글로벌 컨설팅 기업에서는 LangChain의 Search API를 통해 SearXNG와 연동하여, 각국의 최신 정책 동향이나 산업별 보고서를 실시간으로 수집하고, LLM 기반 분석 결과를 내부 포털에 자동으로 제공하는 사례가 있습니다.

또한, 연구기관에서는 논문 검색 엔진만을 선택적으로 활용하여, 최신 연구 동향을 신속하게 파악하고, 벡터DB에 적재한 후 의미 기반 질의응답 시스템을 구축하고 있습니다. 이때, LangChain의 파라미터 설정을 통해 검색 엔진별 신뢰도 가중치나 카테고리별 우선순위를 동적으로 조정할 수 있습니다.

실제 운영에서는 SearXNG의 엔진별 장애나 검색 차단에 대비해, LangChain 내에서 예외 처리를 구현하거나, 검색 실패 시 대체 엔진을 자동 호출하는 로직을 추가하는 것이 권장됩니다. 또한, 검색 결과의 근거 링크와 출처 정보를 체계적으로 관리하여, 내부 보고서나 정책 문서에 활용할 수 있도록 설계합니다.

LangChain과 SearXNG의 결합은 단순한 검색 연계를 넘어, 실시간 정보 수집, 맞춤형 검색, 의미 기반 질의응답, 감사 대응 등 다양한 실무적 요구를 충족하는 유연한 인프라로 발전하고 있습니다. 실제로 많은 기업과 기관에서 LangChain 기반의 AI 에이전트 구축 시 SearXNG를 핵심 검색 게이트웨이로 채택하고 있으며, 내부 데이터와 외부 정보를 통합적으로 활용하는 지능형 파이프라인을 구현하고 있습니다.

---

## 5.3 “로컬/온프레미스 AI 검색” 스택에서의 실제 사용 예

최근 AI 에이전트 및 LLM 커뮤니티에서는 프라이버시, 보안, 비용 효율성을 극대화하기 위해 로컬/온프레미스 환경에서 SearXNG를 검색 게이트웨이로 활용하는 사례가 급증하고 있습니다. 대표적으로 Perplexica, Open WebUI 등 오픈소스 프로젝트와 커뮤니티에서 SearXNG와 LLM의 결합 패턴이 표준으로 자리잡고 있습니다. 이러한 구조는 검색(검색 결과 수집)과 요약/합성(LLM) 레이어를 명확히 분리하여, 최신 정보 획득과 의미 기반 답변 생성의 신뢰성을 동시에 확보할 수

있습니다. 본 절에서는 Perplexica와 Open WebUI 등 실제 적용 사례를 통해, 재현 가능한 구성과 실무적 운영 패턴을 구체적으로 설명합니다.

### 5.3.1 Perplexica: 'Web search powered by SearxNG'로 명시한 오픈소스 사례

Perplexica는 오픈소스 기반의 AI 검색 에이전트로, 공식 프로젝트 설명에서 웹 검색 기능을 SearXNG로 구동한다고 명확히 언급하고 있습니다. 이 구조는 검색 결과 수집과 LLM 기반 요약/합성 레이어를 명확히 분리하여, 최신 정보의 신속한 획득과 의미 기반 답변 생성을 실현합니다.

기술적 배경과 맥락:

- Perplexica는 사용자의 질의에 대해, 먼저 SearXNG를 통해 외부 웹 검색을 수행합니다. 이때, 다양한 검색 엔진의 결과를 취합하여 신뢰할 수 있는 출처를 확보합니다.
- 수집된 링크 목록을 바탕으로, 자체 크롤러 또는 외부 플러그인을 활용하여 본문 데이터를 추출하고, LLM에 입력합니다.
- LLM은 크롤링된 본문 데이터를 바탕으로, 요약, 합성, 근거 제공 등 다양한 형태의 답변을 생성합니다.

구체적인 사례와 시나리오:

- 예를 들어, 사용자가 "2024년 AI 검색 트렌드"에 대해 질의하면, Perplexica는 SearXNG를 통해 최신 뉴스, 블로그, 논문 등 다양한 출처를 수집합니다.
- 각 링크의 본문을 크롤링하여, LLM이 의미 기반으로 요약하고, 답변에 근거 링크를 명시합니다.
- 이 구조는 검색 결과의 신뢰성과 최신성을 동시에 확보하며, 프라이버시 보호와 보안 감사에도 유리합니다.

기술적 세부사항:

- Perplexica의 설정 파일에서 SearXNG 인스턴스의 Base URL을 지정하고, JSON 출력 포맷을 활성화해야 합니다.
- 크롤링 및 본문 추출은 BeautifulSoup, Selenium 등 Python 라이브러리를 활용하며, LLM과의 연동은 LangChain 등 오픈소스 프레임워크로 구현됩니다.

- 답변 생성 시 근거 링크를 명확히 남기며, 감사 대응을 위한 로그 관리도 병행됩니다.

주의사항 및 한계점:

- SearXNG 엔진 장애나 차단(CAPTCHA 등)에 대비해 엔진 믹스 조정, 캐시 활용, 프록시 경유 등이 필요합니다.
- 크롤링 과정에서 robots.txt, 저작권 정책, 데이터 반출 규정을 반드시 확인해야 하며, 사내망에서만 운영하는 경우에도 내부 감사 정책을 준수해야 합니다.
- LLM의 요약/합성 레이어는 최신성, 신뢰성, Explainability를 보장해야 하며, 근거 링크 제공을 통해 감사 대응이 용이하도록 설계해야 합니다.

이처럼 Perplexica는 SearXNG와 LLM의 역할 분리를 통해, 검색 결과 수집과 의미 기반 답변 생성을 실현하는 대표적인 오픈소스 사례로 평가받고 있습니다.

Perplexica의 실제 적용은 다양한 산업 분야에서 확인할 수 있습니다. 예를 들어, 기술 스타트업에서는 Perplexica를 활용하여 최신 기술 동향, 경쟁사 분석, 시장 뉴스 등을 자동으로 수집하고, LLM이 요약한 보고서를 내부 공유 시스템에 배포하는 사례가 있습니다.

또한, 교육기관에서는 Perplexica를 통해 논문, 학술 자료, 교육 정책 등 다양한 출처의 정보를 신속하게 수집하고, LLM이 요약 및 분석하여 교수진과 학생들에게 제공하는 시스템을 구축하고 있습니다.

Perplexica의 오픈소스 특성상, 사용자는 SearXNG 인스턴스의 엔진 조합, 크롤링 정책, LLM 모델 등을 자유롭게 커스터마이즈할 수 있습니다. 이로 인해, 각 조직의 보안 정책, 데이터 활용 목적, 비용 구조에 맞는 맞춤형 AI 검색 에이전트 구축이 가능합니다.

운영 중에는 SearXNG의 엔진별 장애나 검색 차단에 대비해, Perplexica 내에서 예외 처리를 구현하거나, 검색 실패 시 대체 엔진을 자동 호출하는 로직을 추가하는 것이 권장됩니다. 또한, 검색 결과의 근거 링크와 출처 정보를 체계적으로 관리하여, 내부 보고서나 정책 문서에 활용할 수 있도록 설계합니다.

이처럼 Perplexica는 SearXNG와 LLM의 결합을 통해, 최신 정보의 신속한 획득, 의미 기반 검색, 근거 중심 답변 생성 등 AI 에이전트의 실무적 요구를 충족시키는 표준 아키텍처로 자리잡고 있으며, 다양한 산업 분야에서 빠르게 확산되고 있습니다.

### 5.3.2 Open WebUI/로컬 LLM 커뮤니티에서의 결합 패턴(경향 사례)

Open WebUI와 로컬 LLM 커뮤니티에서는 프라이버시 우선 환경에서 SearXNG를 검색 게이트웨이로 활용하는 경향이 뚜렷하게 나타나고 있습니다. 이 구조는 외부 API 비용, 데이터 반출, 정책 변경 리스크를 근본적으로 해소하며, 최신 정보 획득과 의미 기반 검색을 동시에 실현할 수 있습니다.

기술적 배경과 맥락:

- Open WebUI는 다양한 로컬 LLM 모델(예: LLAMA, GEMMA, DeepSeek, EXAONE 등)과 결합하여, 사내망에서만 운영되는 SearXNG 인스턴스를 검색 게이트웨이로 활용합니다.
- 사용자는 Open WebUI를 통해 질의를 입력하고, SearXNG를 통해 최신 정보의 링크 목록을 수집합니다.
- 수집된 링크의 본문을 크롤링하여, LLM이 의미 기반으로 답변을 생성합니다.

구체적인 사례와 시나리오:

- 예를 들어, 사내망에서만 접근 가능한 SearXNG 인스턴스를 운영하면, 외부 API 비용, 정책 변경, 데이터 반출 리스크를 근본적으로 해소할 수 있습니다.
- Open WebUI 내에서 SearXNG와 LLM을 결합하면, 최신 뉴스, 공식 문서, 기술 블로그 등 다양한 출처의 정보를 신속하게 획득하고, 의미 기반으로 답변을 생성할 수 있습니다.

재현 가능한 구성(설정/네트워크/JSON):

- SearXNG의 settings.yml에서 **`output_format: ["json"]`**을 활성화하고, 내부망에서만 접근 가능한 도메인으로 운영합니다.
- Open WebUI의 설정 파일에서 SearXNG 인스턴스의 Base URL을 지정하고, 인증 토큰 또는 API Key를 활용하여 안전하게 검색 요청을 보냅니다.
- 네트워크 정책을 통해 SearXNG가 외부 검색 엔진에 접근할 때 egress allowlist를 설정하고, 프록시 경유나 보안 장비 연동을 병행합니다.
- 검색 결과의 근거 링크를 답변에 명확히 남기며, 감사 대응을 위한 로그 관리도 병행합니다.

주의사항 및 한계점:

- SearXNG 엔진별 rate limit, CAPTCHA, 봇 차단 정책에 따라 검색 결과가 불안정해질 수 있으므로, 엔진 믹스 조정과 캐시 활용이 필요합니다.
- 크롤링 과정에서 robots.txt, 저작권 정책, 데이터 반출 규정을 반드시 확인해야 하며, 사내망에서만 운영하는 경우에도 내부 감사 정책을 준수해야 합니다.
- LLM의 답변 생성 과정에서 Explainability와 Traceability를 보장해야 하며, 근거 링크 제공을 통해 감사 대응이 용이하도록 설계해야 합니다.

이처럼 Open WebUI와 로컬 LLM 커뮤니티에서 SearXNG와의 결합 패턴은 프라이버시 우선, 보안 강화, 비용 효율성, 최신성 확보 등 다양한 실무적 요구를 충족시키는 표준 구조로 자리잡고 있습니다. AI 에이전트 및 RAG 파이프라인에서 SearXNG와 LLM의 결합은, 검색 결과 수집과 의미 기반 답변 생성을 동시에 실현하는 핵심 인프라로 평가받고 있습니다.

Open WebUI와 SearXNG의 결합은 실제로 다양한 조직에서 활용되고 있습니다. 예를 들어, 대규모 병원이나 연구소에서는 환자 데이터와 외부 의학 정보를 통합적으로 분석하기 위해, 사내망에서만 접근 가능한 SearXNG 인스턴스를 운영하고, Open WebUI를 통해 의료진이 최신 논문, 임상 가이드라인, 신약 정보를 신속하게 검색하고 요약할 수 있도록 지원하고 있습니다.

또한, 제조업체나 공공기관에서는 외부 정책 변화, 기술 규제, 산업 동향 등 다양한 정보를 실시간으로 수집하고, LLM이 의미 기반으로 분석하여 경영진이나 실무자에게 제공하는 시스템을 구축하고 있습니다. 이때, SearXNG와 Open WebUI의 결합을 통해 외부 API 비용과 데이터 반출 리스크를 최소화하고, 내부 정책에 맞는 정보 활용이 가능하도록 설계합니다.

운영 중에는 SearXNG의 엔진별 장애나 검색 차단에 대비해, Open WebUI 내에서 예외 처리를 구현하거나, 검색 실패 시 대체 엔진을 자동 호출하는 로직을 추가하는 것이 권장됩니다. 또한, 검색 결과의 근거 링크와 출처 정보를 체계적으로 관리하여, 내부 보고서나 정책 문서에 활용할 수 있도록 설계합니다.

이처럼 Open WebUI와 SearXNG의 결합은 단순한 검색 연계를 넘어, 실시간 정보 수집, 보안 정책 준수, 감사 대응 등 다양한 실무적 요구를 충족하는 유연한 인프라로 발전하고 있습니다. 실제로 많은 기업과 기관에서 Open WebUI 기반의 AI 에이전트 구축 시 SearXNG를 핵심 검색 게이트웨이로 채택하고 있으며, 내부 데이터와 외부 정보를 통합적으로 활용하는 지능형 파이프라인을 구현하고 있습니다.

# Contact Us

hello@cncf.co.kr

02-469-5426

www.cncf.co.kr

## CNF Blog

다양한 콘텐츠와 전문 지식을 통해
더 나은 경험을 제공합니다.

## CNF eBook

이제 나도 클라우드 네이티브 전문가
쿠버네티스 구축부터 운영 완전 정복

## CNF Resource

Community Solution의 최신 정보와
유용한 자료를 만나보세요.

씨엔에프 | CNF

전화 : (02) 469 - 5426
팩스 : (02) 469 - 7247
메일 : hello@cncf.co.kr