

금융분야 오픈소스 소프트웨어 활용·관리 안내서

2022. 12.



〈제·개정 이력〉

[illegible]

목 차

I. 개요	1
1. 배경 및 목적	2
2. 적용 대상 및 범위	3
3. 안내서 구성 및 활용	3
II. 오픈소스 소프트웨어 소개	5
1. 오픈소스 소프트웨어에 대한 이해	6
2. 오픈소스 소프트웨어 활용	8
3. 오픈소스 소프트웨어 공유 플랫폼	11
4. 오픈소스 소프트웨어 선택	13
III. 오픈소스 소프트웨어 보안성 관리	15
1. 오픈소스 소프트웨어의 보안리스크	16
2. 보안성 관리	17
IV. 오픈소스 소프트웨어 라이선스 관리	21
1. 오픈소스 소프트웨어 라이선스 특징	22
2. 오픈소스 소프트웨어 라이선스 관리	22
V. 오픈소스 소프트웨어 관리 절차	25
1. 관리 절차 개요	26
2. 세부 관리 절차	28

I

개요

1. 배경 및 목적
2. 적용 대상 및 범위
3. 안내서 구성 및 활용





개요

1 배경 및 목적

전 세계적 디지털 전환 흐름에 따라 국내 금융권에서도 오픈소스 소프트웨어 활용이 확대되면서 체계적인 활용·관리에 대한 필요성이 증대되고 있다.

해외에서도 여러 기관이 가이드라인을 발간하여 오픈소스 소프트웨어 활용·관리 시 필요한 고려사항이나 판단 기준 등을 세부적으로 제시하고 있다.

국내 금융권의 경우 금융권 IT거버넌스 특성을 반영한 오픈소스 활용·관리 관련 가이드는 부재한 상황이다.

또한, 연구·개발 분야에 대한 망분리 규제 완화 등 전자금융감독규정 개정안 시행('23.1월)에 따라 금융권의 오픈소스 소프트웨어 활용이 보다 확대될 것으로 예상된다.

따라서 금융권에서 오픈소스 소프트웨어를 안전하게 활용하기 위해서는 오픈소스 소프트웨어 활용 시 라이선스 준수 및 보안성의 체계적인 관리가 요구된다.

본 안내서는 비규제 성격으로 오픈소스 소프트웨어 활용시 고려해야 할 사항들과 관리 절차 등을 안내함으로써 안전한 오픈소스 소프트웨어 활용 문화를 조성하고 금융권의 자율 보안 체계 강화를 지원하고자 한다.

2 적용 대상 및 범위

본 안내서는 오픈소스 소프트웨어를 활용하고자 하는 금융회사 및 전자금융업자(이하 ‘금융회사 등’)를 대상으로 한다.

본 안내서에서는 오픈소스 소프트웨어 활용시 라이선스 준수와 안전한 활용과 관련하여 필요한 사항에 대해 다루며, 금융회사 등이 자체 개발한 소프트웨어를 오픈소스 방식으로 공개하는 것이나 외부 오픈소스 프로젝트에 참여하는 것과 오픈소스 소프트웨어 이용을 활성화하는 것 등에 관해서는 다루지 않는다.

국내 법규와 상충하는 부분이나 본 안내서에서 다루지 아니한 부분은 관련 법규 및 규정 등을 따른다.

3 안내서 구성 및 활용

본 안내서는 총 5개의 장으로 구성되어 있다.

제1장 개요는 본 안내서에 대한 전반적인 설명을 담고 있다.

제2장 오픈소스 소프트웨어 소개는 오픈소스 소프트웨어의 종류와 특징, 오픈소스 소프트웨어 활용·선택 방법 등에 대해 안내한다.

제3장 오픈소스 소프트웨어 보안성 관리는 오픈소스 소프트웨어 보안리스크와 보안성 관리 방법을 안내한다.

제4장 오픈소스 소프트웨어 라이선스 관리는 오픈소스 소프트웨어 라이선스의 특징과 관리 방법에 대해 안내한다.

제5장 오픈소스 소프트웨어 관리 절차는 오픈소스 소프트웨어 관리에 필요한 세부 사항을 절차별로 안내한다.

II

오픈소스 소프트웨어 소개

1. 오픈소스 소프트웨어에 대한 이해
2. 오픈소스 소프트웨어 활용
3. 오픈소스 소프트웨어 공유 플랫폼
4. 오픈소스 소프트웨어 선택



II

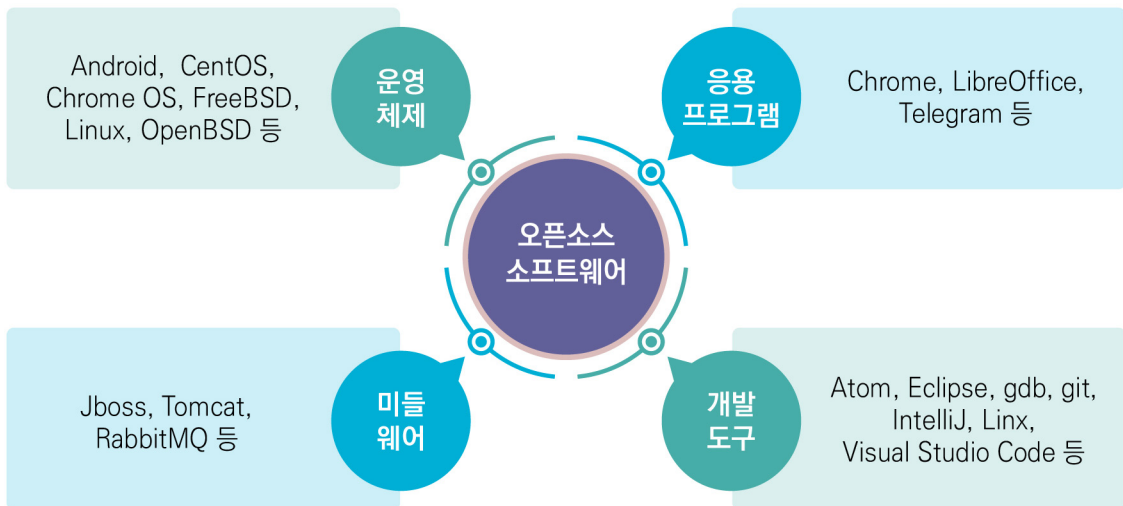
오픈소스 소프트웨어 소개

1 오픈소스 소프트웨어에 대한 이해

가. 개념

오픈소스 소프트웨어란 누구나 자유롭게 확인, 수정, 배포할 수 있는 소프트웨어로서 다수의 협업을 통해 개발·관리되며 집단지성을 통해 기능·보안 등을 최적화할 수 있다.

[그림 1] 오픈소스 소프트웨어의 종류



소프트웨어 중 운영체제, 응용프로그램, 미들웨어뿐 아니라 소프트웨어 개발에 활용되는 개발도구까지 그 유형이나 이용 방식, 개발언어 등과 관계없이 다양한 종류의 소프트웨어가 오픈소스 소프트웨어로 배포·활용되고 있다.

개발자가 오픈소스 소프트웨어를 개발하여 공개할 때는 코드 공개 의무, 특허권·지식재산권 관련 조항, 수정 시 고지 의무와 같은 활용 시 지켜야 할 의무사항 등을 표기한 라이선스와 함께 배포하는 경우가 일반적이다.

참고 소프트웨어 진흥법상 공개소프트웨어 정의

제25조(소프트웨어 연구 및 기술개발 촉진 등)

- ② 정부는 소프트웨어 분야의 국가연구개발사업을 하는 우 다음 각 호의 방법으로 소프트웨어 연구개발이 활성화되도록 노력하여야 한다.
2. 국가연구개발사업의 결과물을 공개소프트웨어(저작권자가 원시코드를 공개하여 활용·복제·수정 및 재배포가 자유로운 소프트웨어를 말한다)로 배포

나. 특징

오픈소스 소프트웨어는 전문 지식 없이도 쉽게 사용 가능한 모듈 등의 형태로 배포되는 경우가 많아 누구나 신속하고 효율적으로 개발할 수 있으며 신기술 활용에도 유용하다.

특히 많이 사용되는 오픈소스 소프트웨어는 대규모 집단지성을 통해 개발·검증되기 때문에 상대적으로 적은 노력으로 안정적 사용이 가능하고, 금융권 등 다양한 산업 분야에서도 적극적으로 활용되고 있다.

이러한 장점에도 불구하고 다른 한편으로는 누구나 코드에 대한 접근과 수정 가능하다는 특징으로 인해 취약점을 쉽게 찾거나 임의로 악의적인 코드를 추가하는 등의 공격이 가능하다는 보안 리스크도 존재한다.

개발자가 고지한 라이선스의 경우 그 의무사항이 다양하기 때문에 의무사항을 충분히 인지하지 못하는 경우 라이선스 미준수로 인해 법적 분쟁이 발생할 수 있다.

그리고 직접 개발한 코드가 아니기 때문에 기능오류, 추가개발 등의 이슈가 발생하면 자체 대응 및 적시 이슈 해결이 어려울 수 있어 이에 대한 고려도 필요하다.

2

오픈소스 소프트웨어 활용

기업은 오픈소스 소프트웨어 형태로 공개된 신기술을 신속하고 효율적으로 적용함으로써 경쟁력을 높일 수 있다. 오픈소스 소프트웨어는 전문가가 아니더라도 쉽게 사용할 수 있도록 구현되어 있으며 모듈 형태로 개발되어 다양한 프로젝트와 쉽게 결합 가능한 경우가 많다.

[표 1] 기업에서 많이 활용되는 오픈소스

오픈소스 명	설명	사용기관
numeric-keyboard	모바일 앱에서 거래 시 시스템 키보드 대신 숫자를 입력할 수 있는 가상 입력 박스	신한카드
TedPermission	안드로이드에서 앱 실행 시 필요한 권한 확인 및 요청	금융결제원
osquery	SQL 기반 운영체제 데이터 탐색, 모니터링 및 분석 프레임워크	안랩
TensorFlow	인공지능 개발에 최적화된 프레임워크로, 인공지능을 활용한 이미지·음성 분석에 활용	네이버

기업은 필요한 기술을 오픈소스 형태로 공동 개발함으로써 제품이나 서비스 출시까지의 기간을 단축하고 업무 효율을 극대화할 수 있다. 미리 개발된 프로토타입을 오픈소스로 공개한 후 추가 개발하거나, 처음부터 오픈소스 형태로 개발하는 등 다양한 방식으로 추진할 수 있다.

법적 분쟁을 방지하기 위해 지적 재산권을 기업에 귀속시키는 CLA¹⁾를 기여자에게 요청하는 사례도 있다. 기여자 라이선스 동의서의 경우 사례에 따라 다르지만 대체로 오픈소스 소프트웨어의 수정·배포·관리에 대한 권한과 라이선스 변경에 대한 권한 등을 회사에 양도하는 내용을 포함하는 경우가 많다.

1) Contributor License Agreement, 기여자 라이선스 동의서

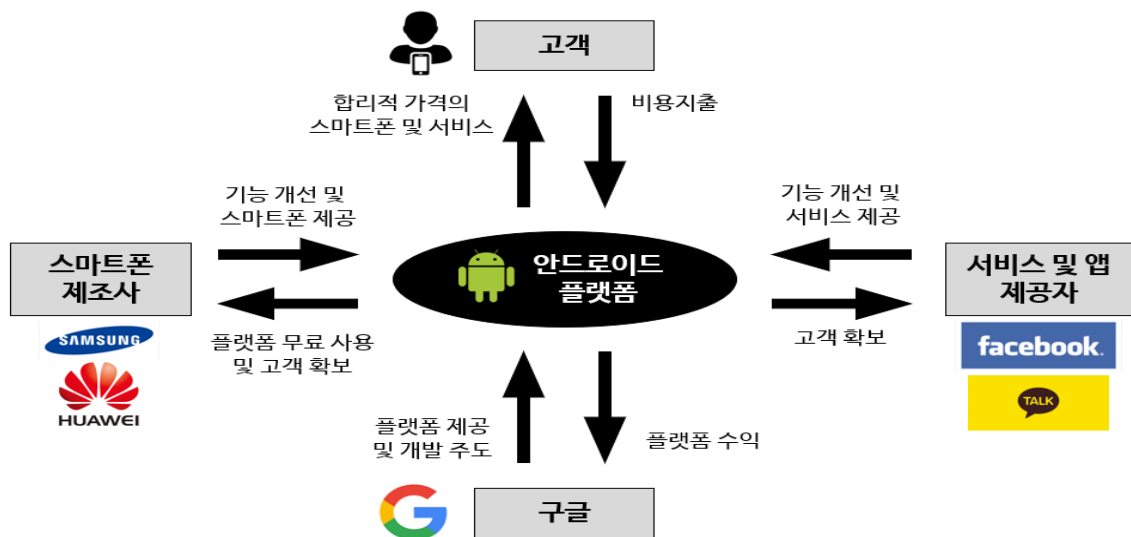
[그림 2] Golang 공동 개발



기업은 자사가 보유하고 있는 혁신적인 기술·플랫폼·인프라 등을 오픈소스 소프트웨어 형태로 공개하여 시장 장악력을 높이고 안정적인 협력 생태계를 구축할 수 있다. 구글이 모바일 운영체제인 안드로이드를 개발하여 오픈소스 형태로 공개한 후 협력 생태계를 구축하여 안정적인 수익화 모델을 구축한 것도 그런 사례 중 하나다.

일단 다수의 사용자를 바탕으로 기술 생태계가 구축되면 주도적인 기업은 생태계에 기반을 둔 다양한 비즈니스 모델을 구축할 수 있다. 또한, 분야에 따라서는 소수의 기업이 기업·플랫폼·인프라를 독점할 가능성도 존재한다.

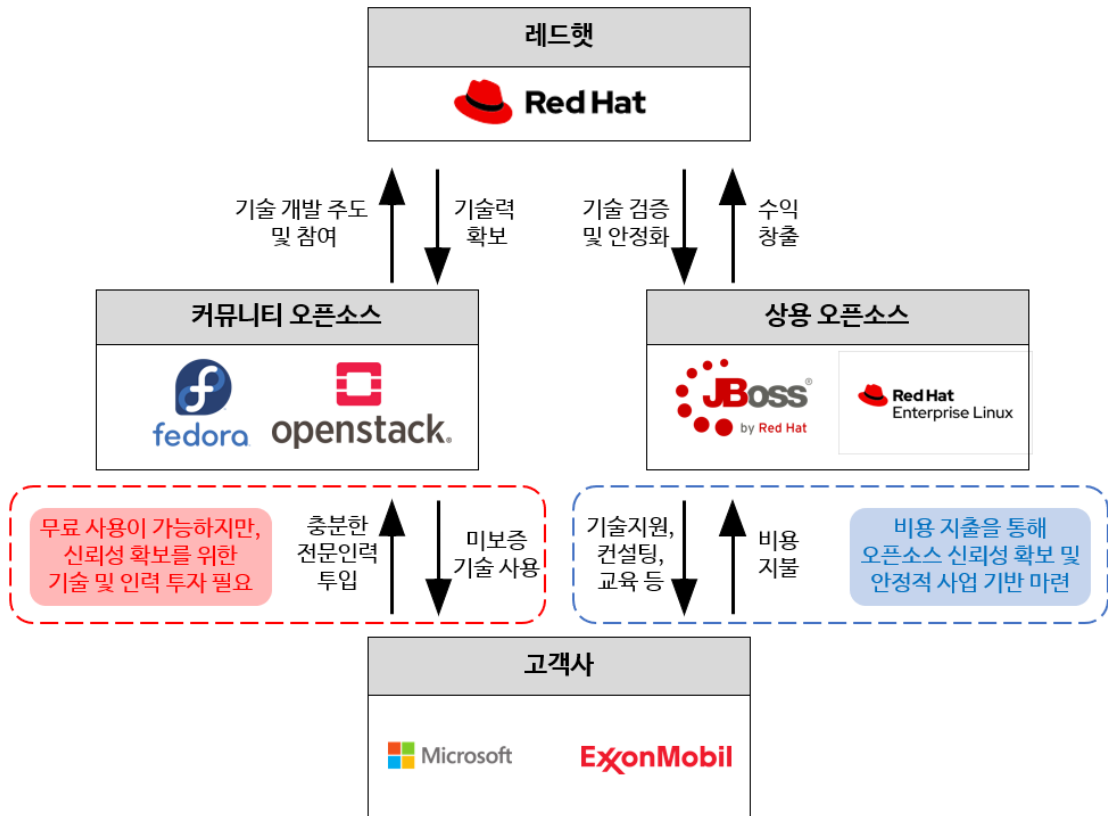
[그림 3] 안드로이드 협력 생태계



(출처: SPRI, 글로벌 오픈소스 (공개SW) 생태계와 주요국 정책)

오픈소스 소프트웨어 개발 참여는 새로운 비즈니스 기회가 되기도 한다. 개발 참여를 통해 자체 기술력을 확보·강화한 후 타 기업에 품질보증·기술지원·컨설팅·교육·인증 등 오픈소스 소프트웨어 활용에 필요한 지원 서비스를 제공할 수 있다. 대표적 사례로 레드햇을 꼽을 수 있으며 기업용 서버 운영체제(RHEL), 어플리케이션 운영·배포 플랫폼(OpenShift) 등의 오픈소스를 지원하고 있다.

[그림 4] 레드햇의 오픈소스 지원 서비스



(출처: SPRI, 글로벌 오픈소스(공개SW) 생태계와 주요국 정책)

3 오픈소스 소프트웨어 공유 플랫폼

오픈소스 형태로 개발한 소프트웨어가 등장하면서 다수의 사용자가 효과적으로 공유·관리·보완할 수 있도록 다양한 플랫폼들이 구축되어 운영 중이다. 이러한 공유 플랫폼을 통해 지식 공유, 질의·토론 등이 가능하며 이를 통해 해당 소프트웨어뿐만 아니라 시장 전체가 더 좋은 방향으로 진화해 나갈 수 있는 선순환 구조를 구성하고 있다.

개인과 기업을 가리지 않고 전 세계적으로 깃헙(GitHub), 깃랩(GitLab), 기티(Gitee) 같은 다양한 오픈소스 공유 플랫폼을 적극적으로 활용하고 있으며, 기업이 자사 소프트웨어를 오픈소스 형태로 개발·운영하는 비중도 지속적으로 증가하고 있다.

[표 2] 오픈소스 공유 플랫폼에서 제공하는 기능

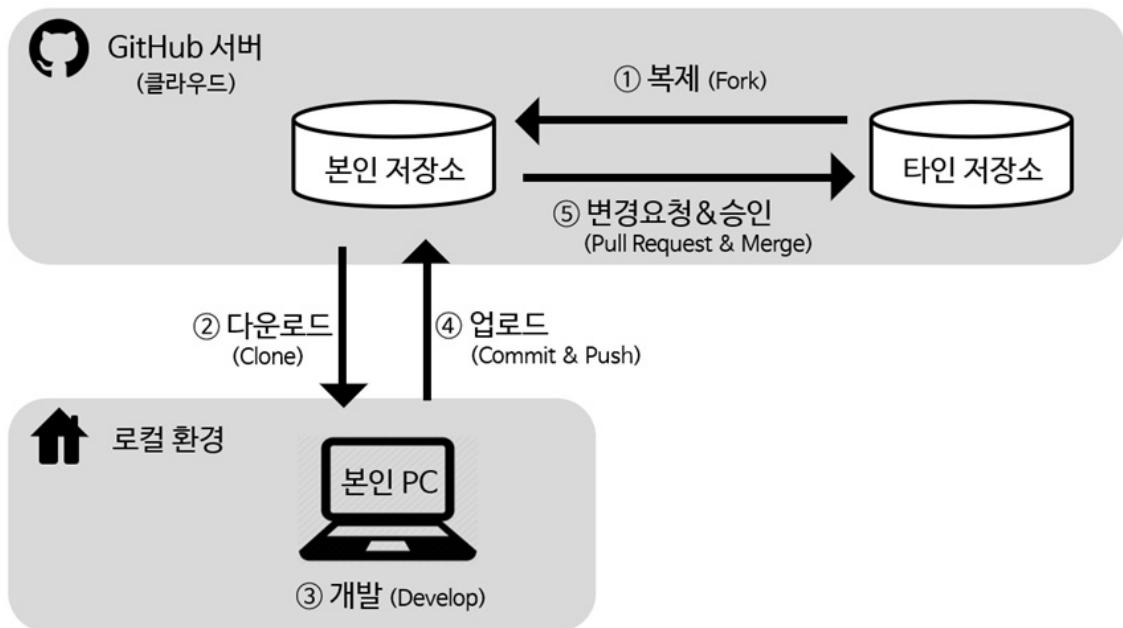
기능	설명
오픈소스 공유	<ul style="list-style-type: none"> - 작성한 코드를 업로드하여 모든 사용자에게 공개 가능 - 필요한 코드를 쉽게 검색하여 활용 가능
오픈소스 관리	<ul style="list-style-type: none"> - 다수가 협업하여 하나의 프로젝트를 개발할 수 있도록 협업 기능 제공 - 코드에 대한 버전 및 종속성 관리 등을 지원
오픈소스 보안	<ul style="list-style-type: none"> - 특정 코드에 대한 보안 이슈를 제보하며, 토론할 수 있는 커뮤니티 적극 장려 - 플랫폼에서 제공하는 보안 점검도구를 통해 취약점 알림

오픈소스 공유 플랫폼은 협업을 위한 다양한 기능을 제공하고 있어 다수의 참여자가 오픈소스 소프트웨어에 대해 공동 개발·관리가 가능하다.

단일 프로젝트에 대해서 개인별 버전 생성이 가능하며 각자가 분업하여 개발한 후 결과물을 병합하는 방식을 많이 사용한다. 오픈소스 공유 플랫폼상의 프로젝트는 클라우드에 저장되며, 활용·변경 등이 필요할 경우 개발자 PC 등으로 내려받아 개발한다. 사용자마다 고유의 클라우드 저장소가 존재하며, 타 사용자의 오픈소스 소프트웨어를 본인 저장소로 이관(Fork)하여 사용하는 것도 가능하다.

다른 사용자 소유의 오픈소스 소프트웨어이거나 공동으로 개발한 경우, 프로젝트 관리자의 승인 과정을 거쳐 코드 변경사항을 반영할 수 있다. 개발자가 오픈소스 소프트웨어 코드를 수정한 뒤 이를 반영해 달라고 요청하면 담당자나 소유권자가 이를 검토 후 승인하게 되며 이러한 절차를 통해 무분별한 코드 변경을 방지할 수 있다.

[그림 5] GitHub 오픈소스 관리체계



오픈소스 공유 플랫폼은 사용자가 안전하고 지속적으로 오픈소스 소프트웨어를 사용할 수 있도록 다양한 정책과 기능을 제공하고 있다.

플랫폼 내 자체 보안도구를 이용한 코드 분석을 통해 오픈소스 소프트웨어에 내재된 취약점이나 오류 등을 탐지하여 사용자에게 알려주는 기능이나 기술 및 보안에 대한 자문서비스 등을 제공하고 있으며 악성코드, 피싱, 저작권·상표권 침해, 개인정보 유출 등 유해한 오픈소스 소프트웨어에 대한 신고 제도를 운영하는 경우도 있다.

[표 3] 플랫폼에서 제공하는 보안 기능

보안 기능	설명
접근제어	<ul style="list-style-type: none"> - 로그인 시, 이중 인증을 통한 계정 보호 - 코드 수정·삭제 시 기밀성, 부인방지 등을 위해 전자서명 활용 - 업무에 따른 코드 및 설정에 대한 접근제어 - 특정 IP만 접근할 수 있도록 제한
보안 모니터링	<ul style="list-style-type: none"> - 오픈소스 코드를 일일이 분석하여 취약점, 버그 등을 자동으로 찾아내어 작성자에게 경고 및 수정 요청 - 오픈소스에 악의적인 코드를 추가하려는 사용자 탐지 및 차단
자문 서비스	<ul style="list-style-type: none"> - 보안을 포함한 다양한 이슈에 대하여 자문 서비스를 운영
신고 제도	<ul style="list-style-type: none"> - 저작권·상표권 침해, 개인정보 유출, 악성코드, 피싱 등의 유해 오픈소스를 신고할 수 있는 제도 운영

4 오픈소스 소프트웨어 선택

오픈소스 소프트웨어 도입 시에는 기능성, 라이선스, 보안성, 참여 활성화 수준 등을 선택 기준으로 고려할 수 있다.

[표 4] 오픈소스 소프트웨어 선택 기준

구분	내용
기능성	<ul style="list-style-type: none"> - 프로젝트에서 필요한 기능을 충족하며 사용하기 적합한지 고려 - 필요 이상으로 지나치게 과도한 기능을 제공하는 것은 지양
오픈소스 사용 조건 및 성능	<ul style="list-style-type: none"> - 프로젝트의 아키텍처나 운영체제와의 호환성 검토 - 벤치마킹을 수행하여 성능 측정 가능
라이선스	<ul style="list-style-type: none"> - 오픈소스에 명시된 라이선스를 반드시 확인 - 특히 GPL²⁾ 등 코드 공개 의무가 있는 라이선스에 유의
보안	<ul style="list-style-type: none"> - 취약점 발견 시 빠르게 조치가 이루어지고 있는지 등 검토 - 특히 마지막 릴리즈가 오래된 경우, 보안취약점을 가지고 있을 가능성이 크므로 유의 - 오픈소스 계정에 게시된 이슈, 오픈소스 설명에 명시된 이슈 트래커 등을 통해 보안 이슈가 있는지 확인

2) General Public License, 자유 소프트웨어 재단에서 만든 오픈소스 소프트웨어 라이선스로서 GNU 일반 공중 사용 허가서라고도 함

구분	내용
참여 활성화 수준	<ul style="list-style-type: none"> - 오픈소스 제공 플랫폼의 Star 개수, Watch 또는 Fork 횟수 등을 통해 해당 오픈소스의 사용자 수를 추측 - 오픈된 이슈 개수, PR수, 마지막 커밋 일시 등을 통해 오픈소스의 활용도, 발전 가능성 등 파악 - StackOverflow 질문 수, 오픈소스 다운로드 수, Google 쿼리 결과 수 등과 같은 수치 활용
소스 코드	<ul style="list-style-type: none"> - 프로젝트에서 사용 중이거나 익숙한 프로그래밍 언어로 작성된 오픈소스를 사용하는 것이 유리 - 낮은 코드 품질의 오픈소스는 버그, 보안취약점, 성능 저하 및 유지관리 등의 이슈를 가질 가능성 - 단위 테스트 코드가 포함되어 있는지, 메서드 또는 함수가 읽기 쉽게 명명되어 사용 중인지, 코딩 컨벤션이 올바르게 지켜지고 있는지 등을 참고하여 소스코드의 품질을 판단
문서화	<ul style="list-style-type: none"> - 설치 방법, 튜토리얼, 참조 설명서 등 문서화가 잘 되어있는 오픈소스일수록 활용도가 높을 가능성

선택 기준을 반영한 고려사항을 통해 적합한 오픈소스 소프트웨어를 보다 쉽게 선택할 수 있다.

참고 오픈소스 소프트웨어 선택 시 고려사항

- ✓ 많이 쓰이고 있는 오픈소스 소프트웨어인가?
- ✓ 개발자 등 담당자가 오픈소스 소프트웨어를 배우는데 어렵지 않은가?
(매뉴얼 등 참고할 수 있는 자료는 충분한가?)
- ✓ 유지보수가 잘 되고 있는가?
- ✓ 유사 오픈소스 소프트웨어와의 차이점은 무엇인가?(얼마나 효율적인가?)
- ✓ 해당 오픈소스 소프트웨어를 쓰지 않는다면 어떤 문제가 생기는가?
- ✓ 라이선스에는 문제가 없는가?
- ✓ 커뮤니티는 활성화되어 있는가?

III

오픈소스 소프트웨어 보안성 관리

1. 오픈소스 소프트웨어의 보안리스크
2. 보안성 관리





오픈소스 소프트웨어 보안성 관리

1 오픈소스 소프트웨어의 보안리스크

오픈소스 소프트웨어의 경우 소스코드가 공개되어 있으므로 취약점을 찾아 악용하기가 용이하며, 누구나 개발에 참여할 수 있으므로 악의적인 코드를 삽입할 수 있다.

대부분의 오픈소스 소프트웨어는 다른 오픈소스 소프트웨어를 활용하여 개발되기 때문에 일단 취약점이나 버그가 발생하면 여러 오픈소스 소프트웨어에 영향을 미치게 된다. 백엔드(Spring), 프론트엔드(React), 모바일(Android), 인공지능(TensorFlow) 등 일부 분야의 경우 소수의 오픈소스 소프트웨어가 높은 점유율을 차지하고 있어 취약점 발생 시 침해사고 등으로 인한 피해가 극대화될 수 있다.

또한, 오픈소스 소프트웨어는 사후 지원·관리의무가 없어서 취약점 및 이슈 등이 발생하더라도 이를 그대로 방치하게 되는 경우가 많다.

참고 오픈소스 소프트웨어 취약점 상위 10개

① 리소스 누수	② 초기화되지 않은 변수	③ Null 포인터 역참조	④ XSS	⑤ 메모리 손상
⑥ 호출 시 추가 인수 오류	⑦ 에러처리 문제	⑧ 안전하지 않은 데이터 처리	⑨ 제어흐름 문제	⑩ 고려되지 않은 예외 사항

(출처 : Synopsys, Open Source Security And Risk Analysis Report, '21.5월)

2 보안성 관리

오픈소스 소프트웨어는 개발 및 관리주체가 명확하지 않은 경우가 많으므로, 사용 시 단계별로 고려사항을 마련하여 리스크를 최소화하는 것이 중요하다.

오픈소스 소프트웨어를 사용하기 전에는 해당 소프트웨어에 대한 기능과 보안에 대한 검증이 필요하다. 해당 오픈소스 소프트웨어에 대한 취약점 및 이슈 등을 확인해야 하며, 필요시 자체 인력이나 오픈소스 지원 서비스 등을 통해 기능과 보안에 대해 검증해야 한다.

오픈소스 소프트웨어를 활용한 개발 중에는 보안부서와의 긴밀한 협업을 통해 취약점을 최소화해야 한다.

[표 5] 오픈소스 소프트웨어 사용단계 별 고려사항

단계	고려사항	설명
개발 전	사전 기능 및 보안성 테스트	<ul style="list-style-type: none"> - 오픈소스에 기능·보안성에 대한 사전검토 - 오픈소스 이슈 현황 확인 - 해당 오픈소스 취약점 정보(CVE, NVD 등) 확인 - 기관(ex. RedHat)에 검토 요청하여 신뢰성 검증
	라이선스 검토	<ul style="list-style-type: none"> - 코드 수정사항 발생 시 사용한 오픈소스에 대한 라이선스 정리를 통한 컴플라이언스 준수 검토 - 상호 충돌하는 라이선스 존재 여부 검토 - SCA를 활용한 라이선스 의무사항 검토 - 법무법인 및 사내 법무팀에게 자문 요청
개발 중	취약점 최소화	- 보안부서와의 협업을 통해 개발이 완료되기 전에 중요기능(인증, 리소스 사용이 큰 기능 등)에 대한 취약점 최소화
	대체수단 확보	- 중요기능의 경우, 기능 실패 혹은 법적 이슈 발생 시에 기능을 장애 없이 수행하기 위한 대비책을 마련
	자체 대응 및 추가 개발 역량확보	- 중요기능의 경우, 단순 사용뿐 아니라 충분한 이해를 바탕으로 자체적인 커스터마이징 및 문제 해결 능력 확보
개발 후	기능 및 보안성 테스트	<ul style="list-style-type: none"> - 테스트 부서, 보안부서 등에서 해당 프로젝트에 대한 기능 및 보안성을 검사 - 배포 전 오픈소스 테스트 자동화 도구를 통해 기능 및 성능 점검 - 오픈소스 취약점 이슈 검토 및 자체적인 정적·동적 분석을 통한 보안성 확보

단계	고려사항	설명
개발 후	종속성 검사	- 프로젝트에 사용된 외부 패키지 종속성 검사 - 종속성 충돌 발생 시, 개발팀과 협의를 통해 빠른 문제해결
	모니터링	- (운영팀) 사용된 오픈소스 현황 관리 - (보안팀) 사용된 오픈소스 취약점 정보 모니터링
	보안패치	- 사용한 오픈소스의 보안패치를 확인 및 적용하여 취약점을 최소화

오픈소스 소프트웨어를 활용한 개발이 완료된 후에는 개발 산출물에 대해 의존성, 바이너리, 스니펫³⁾ 등에 대한 다중 검사를 수행하고, CVE⁴⁾, NVD⁵⁾ 등 취약점 정보를 모니터링하고 최신 패치를 적용해야 한다.

패치 적용 시에는 테스트 환경에 사전 적용하여 소프트웨어의 동작 및 서비스 안정성 등에 부정적 영향이 없는지 확인이 필요하다. 확인이 완료된 패치는 다른 소프트웨어 배포 절차와 마찬가지로 운영환경에 적용한다. 패치 적용 시 인터넷에 위치한 오픈소스 소프트웨어 배포 서버에 직접 접속하여 받는 방식과 파일 형태로 내려받아 대상 시스템에 옮겨 적용하는 방식 등이 있을 수 있는데 금융회사 등의 IT 환경과 망분리 등 관련 규정 등을 참고하여 적합한 방식을 선택한다.

참고 전자금융감독규정 시행세칙

제2조의2 (망분리 적용 예외)

- ② 규정 제15조제1항제5호에서 금융감독원장이 인정하는 경우란 다음 각 호와 같다.
 2. 업무상 외부통신망과 연결이 불가피한 다음의 정보처리시스템(다만, 필요한 서비스 번호(port)에 한하여 연결할 수 있다)
 - 가. 전자금융업무의 처리를 위하여 특정 외부기관과 데이터를 송수신하는 정보처리시스템
 - 나. DMZ구간 내 정보처리시스템과 실시간으로 데이터를 송수신하는 내부통신망의 정보처리시스템
 - 다. 다른 계열사와 공동으로 사용하는 정보처리시스템
- ③ 제1항 및 제2항의 규정은 금융회사 또는 전자금융업자가 자체 위험성 평가를 실시한 후 <별표7>에서 정한 망분리 대체 정보보호통제를 적용하고 정보보호위원회가 승인한 경우에 한하여 적용한다.

3) Snippet, 재사용 가능한 소스코드, 기계어, 텍스트의 작은 부분을 일컫는 프로그래밍 용어

4) Common Vulnerability and Exposures, 공개적으로 알려진 컴퓨터 보안 결함 목록

5) National Vulnerability Database, 美 국가 취약점 데이터베이스

중요업무에 오픈소스 소프트웨어를 활용하여 개발할 경우 인력 및 비용을 투자하여 버그 수정이나 추가개발 등에 대한 자체 대응 역량이나 대체 수단을 확보해야 한다.

오픈소스 소프트웨어 프로젝트에 참여한 개발자 수가 적고 프로젝트 활성화 수준이 낮은 경우, 참여자가 많고 프로젝트 활성화 수준이 높은 경우에 비해 보안이 취약할 가능성이 크기 때문에 가급적 사용을 피하는 것이 바람직하며, 다른 대안이 없어 어쩔 수 없이 사용한다면 보다 높은 수준의 주의가 필요하다.

참고 오픈소스 소프트웨어 저장소 사칭 사례

- ✓ 오픈소스 소프트웨어는 개별 저장소를 통해 관리되며, 일부 공격자는 오픈소스 소프트웨어 저장소를 사칭하는 공격*을 시도

* 타이포스쿼팅(Typosquatting) : 비슷한 모양의 텍스트를 통해 사람들을 속이는 사회공학적 공격기법

- 공격자는 유명 파이선 패키지 저장소인 PyPI의 패키지인 requests를 사칭하는 requesys, requesrs 패키지를 생성하여 랜섬웨어 스크립트 삽입 → 이용자는 requests 패키지로 인식하여 258회 다운로드
- 오픈소스 소프트웨어 프로젝트의 활성화 수준이 낮은 경우, 가급적 사용을 피하는 것이 바람직하며, 다른 대안이 없어 어쩔 수 없이 사용한다면 높은 수준의 주의가 필요

※ 출처 : Sonatype Shines light on typosquatting ransomware threat in PyPI(The Register, '22.8.) 등

오픈소스 소프트웨어 사용 단계별로 보안성을 확보하기 위한 고려사항 및 체크리스트를 만족하기 위해서는 보안전문가와 개발전문가의 유기적인 협업이 필수적이다. 오픈소스 소프트웨어 사용 빈도가 높은 대규모 조직의 경우 오픈소스 소프트웨어와 관련하여 발생하는 보안 이슈를 담당할 조직 구성도 고려해볼 수 있다.

IV

오픈소스 소프트웨어 라이선스 관리

1. 오픈소스 소프트웨어 라이선스 특징
2. 오픈소스 소프트웨어 라이선스 관리



IV

오픈소스 소프트웨어 라이선스 관리

1 오픈소스 소프트웨어 라이선스 특징

오픈소스 소프트웨어 라이선스는 종류에 따라 고지의무, 코드공개, 특허권, 양립성 등의 요구사항이나 제약조건이 상이하며 기재된 조건 및 의무사항을 위반하면 법적제재를 받을 수 있다. 이로 인해 글로벌 기업부터 국내 중소기업에 이르기까지 오픈소스 소프트웨어 라이선스로 인한 법적 소송이 발생하는 경우가 종종 있다.

라이선스 이슈의 대부분은 해당 오픈소스 소프트웨어를 활용하여 개발할 때 개발 결과물의 소스 코드를 의무적으로 공개토록 코드 공개 의무를 부과하는 GPL 등 일부 라이선스에서 발생하고 있다.

다수의 오픈소스 소프트웨어를 함께 활용하여 개발할 경우, 준수해야 할 의무사항 간 관계가 복잡해지면서 상호 충돌 가능성이 커질 수 있다. 특허, 코드 공개 의무, 조합저작물 허용 등 준수해야 할 라이선스 간의 내용이 서로 상이할 때 어느 쪽을 따라야 할지 불분명하게 되면 양립성이 저하된다.

2 오픈소스 소프트웨어 라이선스 관리

하나의 프로젝트에 다수의 오픈소스 소프트웨어를 사용하는 때도 많으므로 체계적인 라이선스 관리가 더욱 중요하다. 오픈소스 사용 절차가 복잡할수록 개발이 지연되므로, 효율성과 리스크 관리를 위한 적합한 체계를 마련하는 것이 중요하다.

이슈가 자주 발생하는 GPL 등의 라이선스나 출처 및 라이선스 정보가 확인되지 않은 오픈소스 소프트웨어는 가능하면 활용을 피하는 것이 좋다. 글로벌 IT 기업인 구글의 경우 자사가 주도하는 오픈소스 소프트웨어 프로젝트에 대해 GPL 계열의 오픈소스 소프트웨어 사용을 금지하고 있는 사례를 참고할 수 있다.

SCA⁶⁾ 등 오픈소스 소프트웨어 분석 도구를 도입하고 오픈소스 소프트웨어 담당 부서 지정 및 라이선스 등에 대한 법적 자문 등을 통해 요구사항이나 제약조건을 확인하고 준수해야만 한다.

참고 오픈소스 라이선스 종류 및 특징 (상위 5개)

주요 내용	Apache 2.0	MIT	GPL 3.0	GPL 2.0	BSD 3
전체 대비 비율	34.1%	29.7%	10.5%	9.9%	5.8%
복제·배포·수정에 대한 권한	○	○	○	○	○
보증의 부인 및 책임의 제한	○	○	○	○	○
배포 시, 라이선스 사본 첨부	○	○	○	○	○
저작권 고지사항	○	○	○	○	○
배포 시, 소스 코드 제공 의무	X	X	○	○	X
수정 시, 수정내용 고지	○	X	○	○	X
명시적 특허 라이선스의 허용	○	X	○	○	X
라이선시가 특허소송 제기 시 라이선스 종료	○	X	○	X	X
조합저작물 작성 및 타 라이선스 배포 허용	○	○	X	X	○

※ 라이선스 내 관련 규정에 대한 언급이 없을 시, 별도의 제약이 없다고 간주

(출처 : WhiteSource, Open Source Licenses in 2022: Trends and Predictions, '22.1월)

6) Software Composition Analysis, 소프트웨어 구성요소 분석

참고 라이선스 위반 사례

- ✓ 국내 A사는 美 아티팩스사를 대상으로 오픈소스 라이선스 위반 관련 소송 합의금으로 205만달러(약 23억원) 지불('17.12.)
 - 美 아티팩스사가 개발·공개한 오픈소스 '고스트스크립트(한글 문서를 PDF 파일로 변환하는 기능을 제공하는 라이브러리)'는 듀얼 라이선스*(GPL 또는 상용)가 적용
 - * 해당 오픈소스를 무료로 사용할 경우 오픈소스가 적용된 소프트웨어의 코드 공개 의무 부여(GPL 라이선스), 코드 공개 의무 없이 이용하고자 할 경우 사용료 지불을 통해 라이선스 구매(상용 라이선스)
 - 美 아티팩스사는 코드 공개 혹은 사용료 지불없이 사용했다는 주장과 함께 美 캘리포니아 법원에 소송 제기
 - 국내 A사는 약식재판 신청 등 소송 무효화 전략으로 맞섰으나, 법원은 고스트스크립트에 GPL과 더불어 '상업용 라이선스'를 동시 적용했다는 점에서 금전적인 손해 산정이 가능하다고 판단하여 국내 A사가 제기한 약식 재판을 기각하였으며, 국내 A사는 美 아티팩스사와 합의 완료

※ 출처 : [단독] 국내A사, 오픈소스 분쟁소송 '205만 달러'로 합의(시사위크, '18.6월)



오픈소스 소프트웨어 관리 절차

1. 관리 절차 개요
2. 세부 관리 절차





오픈소스 소프트웨어 관리 절차

1 관리 절차 개요

금융회사 등은 오픈소스 소프트웨어의 라이선스 준수와 보안 리스크 대응 등을 위해 자사가 이용 또는 제공하고 있는 소프트웨어에 포함된 오픈소스 소프트웨어를 식별하고 목록을 작성해 관리할 필요가 있다.

특히, 새롭게 도입되는 오픈소스 소프트웨어에서뿐 아니라 기존에 활용 중인 오픈소스 소프트웨어에서도 보안 리스크 및 라이선스와 관련해 새로운 이슈들이 발생할 수 있어, 오픈소스 소프트웨어가 활용되는 전체 주기에 걸친 관리가 필요하다.

참고 美 사이버보안 행정명령('21.5월) 중 SBOM⁷⁾ 관련 주요 내용

- ✓ 바이든 행정부는 소프트웨어 공급망 강화를 위해 사이버보안 행정명령을 발동하여 SBOM 제출 의무화
- ✓ 필수적인 항목, 자동화, 절차 등 SBOM 활용에 필요한 최소 요소 공표(NTIA⁸⁾)
- ✓ SBOM 생성·처리를 자동화하기 위한 데이터 형식은 SPDX⁹⁾, CycloneDX, SWID 3가지로 제한

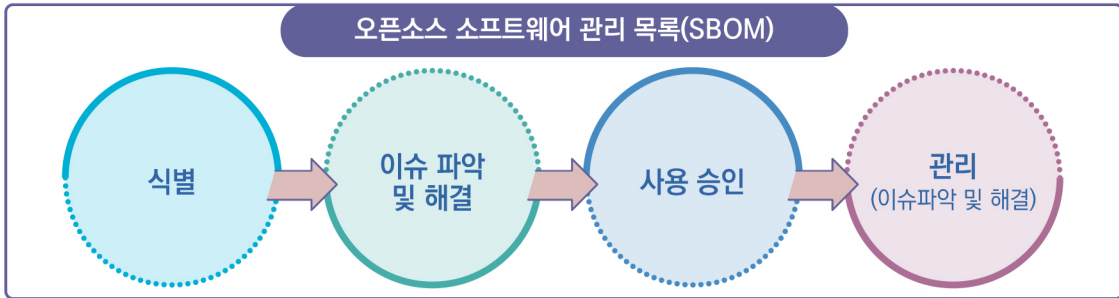
오픈소스 소프트웨어에 대해 작성한 SBOM 목록을 중심으로 ① 식별 → ② 이슈 파악 및 해결 → ③ 사용 승인 → ④ 지속적인 관리로 구성된 절차를 통해 오픈소스 소프트웨어의 안전한 활용이라는 목표에 더욱 가깝게 다가갈 수 있다.

7) Software Bill Of Materials, 소프트웨어 자재명세서

8) National Telecommunications and Information Administration, 美 전기통신 및 정보청

9) Software Package Data eXchange, 소프트웨어 패키지 데이터 교환

[그림 6] 오픈소스 소프트웨어 관리 절차



[표 6] SBOM 데이터 형식별 구성요소 및 파일 형식

데이터 형식명	관리주체	구성요소	파일 형식
SPDX	리눅스 재단	문서 생성정보, 패키지 정보, 파일정보, 스니펫 정보, 기타 라이선스 정보, 관계, 주석	.xls, .spdx, .rdf, .json, .yaml, .xml
CycloneDX	OWASP ¹⁰⁾	BOM 메타정보, 구성요소 정보, 서비스 정보, 의존성 관계, 구성, 취약점, 확장	.json, .xml
SWID	국제표준 (ISO/IEC)	소프트웨어 제품 식별, 제품 버전 특성화, 제품 생산 및 배포 주체(기관 또는 개인), 제품 구성요소 목록, 소프트웨어 제품 간 관계 설정, 기타 설명을 위한 메타정보	.xml

또한, 최근 금융회사가 AI·클라우드 등 다양한 ICT 기술을 전자금융에 접목·위탁하는 사례가 증가함에 따라 서비스의 복잡성 및 연계성 증가로 인해 외부 사업자의 리스크가 금융회사 내부로 전이될 수 있는 위험 또한 점차 확대되고 있다. 이에 제3자 리스크에 대해 선제적·능동적 관리를 위해 전자금융거래와 직접 업무를 보조하거나 대행하는 주요 서비스 제공 사업자(전자금융보조업자 등)가 사용하는 오픈소스 소프트웨어에 대한 관리도 고려해 볼 필요가 있다.

10) the Open Web Application Security Project, 오픈소스 웹 애플리케이션 보안 프로젝트

2

세부 관리 절차

가. 식별

새로운 서비스나 소프트웨어를 개발·도입하거나 기존 서비스를 개편하는 경우 프로젝트에 포함되는 오픈소스 소프트웨어뿐만 아니라 기존 사용 중인 오픈소스 소프트웨어를 식별하고 관리 대상 목록으로 작성하여 오픈소스 소프트웨어 활용·관리 전반에 걸쳐 활용한다.

모듈로 도입되는 오픈소스 또는 상용 소프트웨어의 경우 해당 소프트웨어가 포함하고 있는 구성요소에 대해서도 오픈소스 소프트웨어가 있는지 확인이 필요하며, 분석을 위해 SCA 등 자동화 도구를 활용할 수 있다. 상용 소프트웨어의 경우 보다 정확한 식별을 위해 개발 업체에 포함된 구성요소 중 오픈소스 소프트웨어 목록 제출을 요구해 자동화 도구 분석 결과와 비교해 볼 수도 있다.

국내의 경우 상용 소프트웨어 도입 또는 외주 개발을 통한 시스템 구축 시 지적재산권과 관련한 분쟁 발생 시 모든 책임을 계약업체가 지도록 제안요청서에 명시하는 경우가 많다. 다만, 구성요소에 포함된 오픈소스 소프트웨어에 대한 보다 확실한 식별 및 철저한 관리를 위해서는 외주 개발 또는 상용 소프트웨어 도입 시 포함된 오픈소스 소프트웨어를 식별하여 목록을 제출토록 제안요청서 작성 시 최소 구비요건에 포함하는 방안도 고려해볼 만하다.

참고

전자금융감독규정

제21조(정보처리시스템 구축 및 전자금융거래 관련 계약)

5. 구매 또는 개발한 제품의 소유권, 저작권 및 지적재산권 등의 귀속관계를 명확히 하여 사후 분쟁이 발생하지 않도록 할 것

오픈소스 소프트웨어를 식별하여 목록을 작성할 때 추후 의존성이나 라이선스 등에 대한 관리와 취약점 발생 시 신속한 대응을 위해 SPDX 명세 또는 공개 소프트웨어 정보 교환 명세(TTA¹¹⁾) 등 국내외 관련 표준을 준수하여 작성·관리하는 것이 바람직하다.

참고 공개 소프트웨어 정보 교환 명세 주요 내용

- ✓ 소프트웨어 패키지와 관련된 저작권, 라이선스 컴포넌트들의 정보를 전달하기 위한 표준 서식을 정의, 국외 표준인 SPDX 명세(버전 2.2)* 참조
 - 표준 서식은 생성정보, 패키지정보, 라이선스정보, 파일정보와 검토자 정보를 포함
- * Software Package Data Exchange Specification Version 2.2(출처 : <https://spdx.dev>)

개발이 완료되어 운영 중이거나 이미 제품 형태로 판매되고 있는 소프트웨어의 경우 SCA 등 자동화 도구를 활용하여 제품이나 서비스에 포함된 오픈소스 소프트웨어를 식별해야만 한다. 신규 또는 리뉴얼 프로젝트와 마찬가지로 오픈소스 소프트웨어 활용·관리 전반에 걸쳐 활용하기 위해서 식별된 오픈소스 소프트웨어는 목록으로 작성하여 관리한다.

식별된 오픈소스 소프트웨어는 라이선스와 보안취약점에 대한 리스크를 함께 관리할 수 있도록 한다. 실제로 많은 오픈소스 분석 도구는 오픈소스 라이선스와 보안취약점을 모두 분석할 수 있도록 지원하고 있다.

[표 7] 오픈소스 식별 및 관리 운영 사례

방법	설명
오픈소스 소스코드 저장소를 통한 라이선스 확인	<ul style="list-style-type: none"> - LICENSE 혹은 COPYING 파일에서 라이선스 정보 확인 - 소스코드 파일 상단 주석(저작권 및 라이선스 정보 표기) 확인 - README 파일 또는 웹사이트에서 라이선스 정보 확인
분석 도구를 활용한 라이선스 및 보안취약점 식별	<ul style="list-style-type: none"> - 오픈소스 라이선스 및 보안취약점 식별을 위하여 오픈소스 분석도구를 통한 자동화 식별 환경 구성
오픈소스 게시판 운영	<ul style="list-style-type: none"> - 별도 오픈소스 게시판을 통해 개발자의 오픈소스 이용 문의 응대* <p>* (사례) 특정 오픈소스 사용시 보안취약점 발생 이슈 문의 사항에 대해, 보안취약점 문제가 없는 상위 버전 업데이트 권고</p>

11) Telecommunications Technology Association, 한국정보통신기술협회

참고**과학기술정보통신부, 오픈소스 Log4j 긴급 보안패치 권고**

- ✓ Apache Log4j 2서비스에 보안취약점이 발견됨에 따라 권고하며 관련 취약점을 악용한 악성코드 감염을 우려하여 긴급 보안업데이트 권고('21.12.)
 - 인터넷 서비스 운영·관리 목적의 로그 기록을 남기기 위해 사용하는 프로그램인 Log4j에 업데이트 미수행시, 취약점을 악용한 공격자가 원격에서 공격 코드를 실행할 수 있어 업데이트* 적용 권고
 - * 아파치(Appach) 재단은 해당 취약점을 해결한 보안 업데이트 발표('21.12.)
 - 과학기술정보통신부는 전자금융기반시설, ISMS 인증기업(758개사), CISO(23,835명), C-TAS(328개사), 클라우드 보안인증 기업(36개사), 웹호스팅사(477개사), IDC(16개사) 등을 대상으로 긴급 전파

※ 출처 : 과학기술정보통신부, 「아파치(Apache) Log4j 2 웹서비스 긴급 보안 패치 권고」('21.12.)

나. 이슈 파악 및 해결

수작업 또는 자동화 도구 등을 활용하여 식별된 오픈소스 소프트웨어의 라이선스 및 보안 리스크 관련 이슈를 파악한다.

보안 관련 이슈에 대해서는 관리 목록에 존재하는 오픈소스 소프트웨어에 해당하는 CVE, NVD 등을 공개 취약점 데이터베이스에서 검색하여 확인할 수 있다. 다만, 오픈소스 소프트웨어를 수정하여 사용하였을 경우 해당 취약점이 제거되거나 새로운 취약점이 발생할 수 있으며 이 경우 소스코드 점검 도구 등 별도의 방법을 통해 보안 관련 이슈 파악이 필요하다.

라이선스 관련 이슈의 경우 오픈소스 소프트웨어 소스코드 내 포함된 라이선스 고지 문자열이나 소스코드 자체 또는 바이너리를 비교 분석하여 확인할 수 있다. 다만, 보안 관련 이슈와 마찬가지로 오픈소스 소프트웨어를 수정하여 사용하였을 경우 라이선스 관련 확인이 불충분하게 이루어질 수 있어 주의가 필요하다.

이슈가 발견되었을 때는 해당 오픈소스를 다른 것으로 대체하거나 취약점을 제거하는 작업이 필요하므로 오픈소스 소프트웨어를 직접 활용하는 개발부서 등과 긴밀한 협의가 필요하다.

보안 이슈 중 하나인 취약점의 경우 취약점 분석·평가 등을 통해 발견될 수 있으며, 오픈소스 소프트웨어를 활용하는 과정에서 발생한 경우와 오픈소스 소프트웨어 자체적으로

발생한 경우로 나누어 대처할 수 있다. 활용하는 과정에서 발생한 경우, 해당 원인이 된 설계상의 실수나 소스코드를 수정하면 비교적 간단하게 이슈가 해결된다.

하지만, 오픈소스 소프트웨어 자체적으로 발생한 취약점의 경우 먼저 해당 오픈소스의 내부 설계 및 소스코드를 이해하는 과정이 필요하며, 취약점을 제거한 후에도 라이선스 준수, 자사 평판, 기술 노출 등을 다각적으로 검토하여 오픈소스 소프트웨어 프로젝트의 소스코드 저장소를 통해 취약점을 제거할 것인지, 아니면 금융회사 등의 내부 소스코드 저장소로 해당 오픈소스 소프트웨어를 이관하여 취약점을 제거할 것인지 등을 결정해야 한다.

오픈소스 내부 설계 및 소스코드의 이해, 취약점 수정, 소스코드를 금융회사 등의 내부 저장소로 이관·관리를 기존 개발팀이 대응하는 것도 가능하나, 보다 원활한 대응을 위해서는 오픈소스 소프트웨어 대응을 위한 별도 담당 조직 수립도 고려할 수 있다.

다. 승인

이슈가 해결된 오픈소스 소프트웨어 사용을 승인한다. 식별된 오픈소스 소프트웨어의 경우 원칙적으로 승인을 완료한 경우에만 사용 가능케 하는 것이 바람직하며, 이미 사용 중인 오픈소스 소프트웨어를 새롭게 식별하였으나 아직 승인하지 못한 경우 예외적으로 이슈 파악 및 해결에 필요한 기간에 한정해 사용토록 할 수 있다.

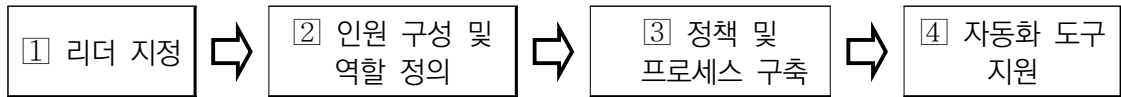
오픈소스 소프트웨어 사용을 위해서는 담당 조직을 통해 사용을 승인하는 절차가 필요하며, 개발 및 보안관리 프로세스에 포함하여 승인 절차를 마련하도록 해야 한다.

오픈소스 소프트웨어 관련 담당 조직이 없는 경우 라이선스 검토 조직 및 보안 리스크 검토 조직의 승인이 모두 필요하며, 담당 조직이 있는 경우 해당 조직에서 승인을 담당할 수 있다.

보안 리스크나 라이선스 관련 검토뿐만 아니라 오픈소스 소프트웨어의 필요성 및 커뮤니티의 활성화 수준 등 오픈소스 소프트웨어 전반을 검토하고 지원하기 위해서는 담당 조직을 구성하는 것이 바람직하다.

참고 오픈소스 거버넌스 체계 구축

- ✓ OSPO(Open Source Program Office) 조직 운영
 - 오픈소스 거버넌스 체계 구축, 오픈소스 전략 수립 및 실행에 필요한 정책, 프로세스 및 도구 제공
 - OSPO가 수행하는 전략적 결정 사항 : 오픈소스 컴플라이언스 활동, 오픈소스 선택, 오픈소스 사용을 위한 승인 및 반납 절차, 이슈 해결 지원 등
 - OSPO 구성 : 일반적으로 오픈소스를 활용하는 부서 소프트웨어 개발 조직에 구성 또는 기업의 비즈니스 및 오픈소스 전략에 따라 구성
 - ※ OSPO 조직이 존재하지 않는 경우, 오픈소스 사용 검토, 정책 수립, 취약점 예외, 라이선스 사용 승인 및 공표 등 이슈 사항에 따라 내부통제 부서(준법, 정보보호 등)와 협업하여 승인 절차 마련 필요
- ✓ OSPO 구성 절차 및 역할
 - (리더 지정) 오픈소스 개발 방법론을 충분히 이해하고, 오픈소스 전략과 정책을 모든 구성원이 이해하도록 전파할 수 있는 커뮤니케이션 역량을 갖춘 리더 임명
 - (인원 구성 및 역할 정의) 기업마다 규모나 업종이 다르고, 오픈소스를 통해 얻고자 하는 목적이 다르므로 OSPO 역할이 달라질 수 있음
 - (정책 및 프로세스 구축) 오픈소스 정책 수립 및 실행을 위한 프로세스 구성
 - (자동화 도구 지원) 자동화하기 위한 도구를 개발하여 제공하는 역할 수행



※ 이 과정에서 부서 간 협업을 유도하고, 발생하는 이슈를 해결

[OSPO 인원 구성과 역할(예시)]

인원 구성	역할
오픈소스 프로그램 매니저	오픈소스 프로그램을 관리&전략을 수립하는 역할로 다음을 담당 <ul style="list-style-type: none"> - 기업의 비즈니스 목표에 부합하는 오픈소스 전략 수립 - 기업의 모든 오픈소스 활동 감독 - 효과적인 오픈소스 활동을 위한 정책 수립 및 프로세스 구축
컴플라이언스 담당	기업의 오픈소스 컴플라이언스*를 보장하기 위한 역할 <ul style="list-style-type: none"> * 외부로 배포 혹은 서비스하는 제품 소프트웨어 및 서비스에 포함된 오픈소스 파악 및 해당 오픈소스 라이선스 의무사항 준수하는 활동
법무 담당	법률 전문가를 포함해야 하며, 포함하지 못할시 법률 조언을 받을 수 있는 창구 마련 필요
IT 담당	도구를 활용하여 오픈소스 컴플라이언스와 오픈소스 보안취약점 점검을 효율화*하기 위한 역할을 담당 <ul style="list-style-type: none"> * 오픈소스 프로그램을 효율적으로 운영하기 위해 가능한 자동화 권장

※ 출처 : 정보통신산업진흥원, 「기업 공개소프트웨어 거버넌스 가이드」('21.11.)

라. 관리

오픈소스 소프트웨어를 활용하는 기간 전체에 걸쳐 지속적인 관리를 수행할 필요가 있다. 특히, 오픈소스 소프트웨어는 활용하는 사람이나 기업이 별도의 수정을 하지 않더라도 오픈소스 소프트웨어 자체 취약점이 새롭게 발견되는 등 보안 이슈가 발생할 가능성이 항상 존재하기 때문에 지속적인 모니터링을 통해 이슈를 파악하고 해결하는 것이 필요하다.

현실적인 한계는 있겠으나, 원칙적으로 조직에서 활용 중인 모든 오픈소스 소프트웨어의 보안 이슈 및 라이선스 변경사항을 모니터링 하는 것이 바람직하며, 이를 위해 식별 단계에서 관련 표준 등을 참고해 작성해 둔 오픈소스 소프트웨어 관리 대상 목록과 자동화 도구 등을 활용할 수 있다.

조직 내부에 오픈소스 소프트웨어 저장소를 구성하고, 프로그램 개발시 해당 저장소를 통해 받아 올 수 있도록 하면, 내부에서 사용하는 오픈소스 소프트웨어 사용 현황을 파악할 수 있고 승인받은 오픈소스 소프트웨어만 허용할 수 있다. 이러한 저장소를 활용함으로써 출처가 불분명한 오픈소스 소프트웨어를 직접 다운로드 받는 것을 방지하고, 오염된 오픈소스 소프트웨어가 유입되는 리스크를 사전에 예방할 수 있으며, 사내에서 사용 중인 오픈소스 라이브러리에서 발생한 이슈에 효과적으로 대응할 수 있다.

오픈소스 소프트웨어 라이선스를 최초 검수한 이후에, 오픈소스 사용 변경, 소스코드 변경 사항이 발생하는 경우 라이선스 검수도 필요하며 효율적으로 관리하기 위해 주기적인 점검 및 검수를 수행할 수 있다.

오픈소스 소프트웨어를 사용하는 부서는 오픈소스 소프트웨어를 더 이상 사용하지 않게 되면, 관리 대상에서 배제하기 위해 이를 통지할 필요가 있다. 관리 대상에서 배제된 오픈소스 소프트웨어에 대해서는 더 이상 라이선스 변동사항 또는 보안 이슈를 추적할 필요가 없으나 이력 관리를 위해 목록은 유지할 필요가 있다.

별첨1

오픈소스 소프트웨어 관리 절차별 점검 체크리스트

※ 법령·행정지도 등 금융규제를 위한 체크리스트에 해당되지 않으며, 금융회사의 자율적인 보안강화를 통한 디지털혁신을 지원하기 위해 마련

관리 절차	항목	현황 (O/X)
식별	<input type="checkbox"/> 소프트웨어 신규 개발·도입 시 포함된 오픈소스 소프트웨어 식별 절차가 마련되어 있는가? (예시: 점검 도구를 사용하여 오픈소스 소프트웨어 식별 등)	
	<input type="checkbox"/> 기 도입되어 사용(또는 운영)하고 있는 소프트웨어에 대해서도 오픈소스 소프트웨어 식별 방안이 존재하는가? (예시: 일정 주기(예: 분기)별로 점검 도구를 사용하여 개발 소스코드 점검 등)	
	<input type="checkbox"/> 식별한 오픈소스 소프트웨어에 대해 소프트웨어 이름, 버전 등을 관리 대상 목록으로 작성하였는가?	
	<input type="checkbox"/> 식별된 오픈소스 소프트웨어에 대해 소프트웨어 자재명세서(SBOM)을 작성(또는 징구)하고 있는가?	
	<input type="checkbox"/> 관리 대상 목록은 소프트웨어 자재명세서(SBOM) 형태로 작성하였는가?	
	<input type="checkbox"/> 오픈소스 라이선스 의무, 제한 및 권리를 검토하는 절차가 있는가?	
	<input type="checkbox"/> 전자금융거래 관련 업무를 직접적으로 보조하거나 대행하는 사업자(전자금융보조업자 등)가 금융회사에 제공하는 서비스(업무)에서 사용하는 오픈소스 소프트웨어를 식별하는 절차를 마련하고 있는가?	
이슈 파악 및 해결	<input type="checkbox"/> 오픈소스 소프트웨어에 대해 보안 이슈를 파악하고 있는가? (예시: 점검 도구 또는 자체적 관리를 통해 오픈소스 소프트웨어의 CVE 취약점 식별 등)	
	<input type="checkbox"/> 오픈소스 소프트웨어에 대해 보안 이슈를 해결하고 있는가? (예시: CVE 취약점의 위험도(Severity)에 따른 정책 수립, 위험도가 치명적(critical)인 경우 출시 전 버전 업데이트 필수 조치 등)	
	<input type="checkbox"/> 오픈소스 소프트웨어에 대해 라이선스 관련 이슈를 파악하는 절차가 있는가?	

관리 절차	항목	현황 (O/X)
이슈 파악 및 해결	<input type="checkbox"/> 오픈소스 소프트웨어에 대해 파악된 라이선스 이슈를 해결하고 있는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어에 대해 조치된 취약점을 관리 대상 목록에 반영하였는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어에 대해 파악된 라이선스 현황을 관리 대상 목록에 반영하였는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어에 대한 보안 이슈 파악에 점검도구 등을 활용하여 체계적으로 대응하고 있는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어에 대한 라이선스 관련 이슈 파악에 점검 도구 등을 활용하여 체계적으로 대응하고 있는가?	
	<input type="checkbox"/> 전자금융거래 관련 업무를 직접적으로 보조하거나 대행하는 사업자(전자금융보조업자 등)가 금융회사에 제공하는 서비스(업무)에서 사용하는 오픈소스 소프트웨어에 대한 보안 이슈를 주기적으로 파악하고 있는가?	
승인	<input type="checkbox"/> 오픈소스 소프트웨어 관련 보안 이슈를 검토한 후 사용을 승인하는가? (예시: 소프트웨어 배포 승인 요청시 오픈소스 소프트웨어의 보안 점검 결과 포함 여부 등)	
	<input type="checkbox"/> 오픈소스 소프트웨어 관련 라이선스 이슈가 충분히 해결되었는지 검토를 수행한 후 사용을 승인하는가? (예시 : 소프트웨어 배포 승인 요청시 외부에 오픈되는 오픈소스 소프트웨어 라이선스 점검 결과 포함 여부 등)	
	<input type="checkbox"/> 오픈소스 소프트웨어별 저장소(커뮤니티) 활성화 수준에 대해 검토하고 활성화 수준이 저조할 경우 관리 대안에 대해서 검토하는가?	
	<input type="checkbox"/> 이슈 파악 및 해결 결과를 검토하고 승인할 오픈소스 소프트웨어 담당 조직*이 구성되어 있는가? * 금융회사가 자체적으로 판단하여 구성	
	<input type="checkbox"/> 전자금융거래 관련 업무를 직접적으로 보조하거나 대행하는 사업자(전자 금융보조업자 등)가 금융회사에 제공하는 서비스(업무)에서 사용하는 오픈소스 소프트웨어에 대한 보안 이슈 해결 방안을 검토하고 있는가?	
관리	<input type="checkbox"/> 승인 완료한 오픈소스 소프트웨어만을 사용하고 있는가?	
	<input type="checkbox"/> 승인하여 사용 중인 오픈소스 소프트웨어 관리 대상 목록을 관리하고 있는가?	

관리 절차	항목	현황 (O/X)
관리	<input type="checkbox"/> 오픈소스 소프트웨어 관리 대상 목록에 소프트웨어 이름, 소프트웨어 버전, 해당 소프트웨어를 사용하고 있는 다른 소프트웨어 목록, 라이선스 현황 등을 기록·관리하고 있는가?	
	<input type="checkbox"/> 관리 대상 목록에 있는 오픈소스 소프트웨어에 대해 정기적인 이슈 파악 및 해결을 하고 있는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어 관리 대상 목록을 소프트웨어 자재명세서(SBOM) 형태로 관리하고 있는가?	
	<input type="checkbox"/> 승인하여 사용 중인 오픈소스 소프트웨어에 대해서는 별도의 저장소를 만들어 자체적으로 관리하고 있는가?	
	<input type="checkbox"/> 외부로 배포하는 어플리케이션에서 GPL 계열의 라이선스를 사용하는 경우, 어플리케이션 소스코드 공개정책을 별도 마련하고 있는가?	
	<input type="checkbox"/> 전자금융거래 관련 업무를 직접적으로 보조하거나 대행하는 사업자(전자금융보조업자 등)가 금융회사에 제공하는 서비스(업무)에서 사용하는 오픈소스 소프트웨어를 전자적 침해위험 관리 대상에 포함하여 관리하고 있는가?	
기타	<input type="checkbox"/> 오픈소스 소프트웨어를 선택하는 기준이 수립되어 있는가?	
	<input type="checkbox"/> 사용이 제한된 오픈소스 소프트웨어 라이선스를 내부 연구개발 및 테스트 등을 위해 예외적으로 승인하는 절차가 있는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어 활용 및 관리를 위해 각 역할(개발자, 오픈소스 매니저, 보안 담당자 등)을 담당하는 조직, 인원 또는 직무를 기술한 문서를 관리하고 있는가?	
	<input type="checkbox"/> 오픈소스 소프트웨어 활용 및 관리를 위해 인력 및 예산을 적절히 지원하고 있는가?	
	<input type="checkbox"/> 소프트웨어 업무 관련 직원에게 오픈소스 정책의 존재를 알리는 문서화된 절차가 있는가?	
	<input type="checkbox"/> 오픈소스 컴플라이언스(보안 이슈 및 라이선스 등)와 관련된 문제 해결을 위해 내부 또는 외부의 법률 전문 자문을 이용할 수 있는 방법이 있는가?	

별첨2

주요 오픈소스 관리도구

Black Duck	<input type="checkbox"/> 오픈소스 소프트웨어를 사용하는 동안 발생하는 라이선스와 취약점, 소스 코드 품질 관리가 가능한 포괄적인 솔루션 <input type="checkbox"/> 소프트웨어 공급망과 애플리케이션 라이프사이클 전반에 걸쳐 오픈소스 소프트웨어의 라이선스와 보안 관리 가능
LABRADOR	<input type="checkbox"/> 오픈소스 소프트웨어의 구성요소를 담은 SBOM(Software Bill Of Materials) 제공으로 소프트웨어 공급망 보안 관리 지원 <input type="checkbox"/> 오픈소스 소프트웨어의 라이선스 및 취약점 리스크를 탐지하고 패치할 수 있는 소프트웨어 안전관리 플랫폼
Clarity	<input type="checkbox"/> 바이너리 코드 분석을 통해 소프트웨어 내에 존재하는 오픈소스 컴포넌트를 식별, 라이선스 및 보안취약점 정보 제공
Clearly Defined	<input type="checkbox"/> 오픈소스 소프트웨어의 출처, 라이선스 정보 등을 확인할 수 있는 데이터베이스 구축·제공하기 위한 커뮤니티 프로젝트 <input type="checkbox"/> 사용자가 데이터베이스 구축·수정에 참여할 수 있도록 하여 오픈소스 소프트웨어 데이터베이스 신뢰성 제고
Code eye	<input type="checkbox"/> 한국저작권위원회가 오픈소스SW 라이선스 종합정보시스템을 통해 제공하는 오픈소스 SW라이선스 검사 서비스 <input type="checkbox"/> 라이선스 검사, 소스코드 비교 등을 수행하여 결과 보고서 제공
Fossa	<input type="checkbox"/> 풍부한 오픈소스 메타데이터 및 정교한 정책 거버넌스를 제공하며, DevOps 환경 지원 등 개발자 친화적 기능들로 구성
FOSSID	<input type="checkbox"/> 오픈소스 라이선스 및 보안취약점 관리 솔루션으로서 소스 코드 내 컴포넌트를 탐지하여 각 컴포넌트의 라이선스 및 보안취약점을 식별 <input type="checkbox"/> 방대한 오픈소스 데이터베이스 및 자동 데이터 수집 기술, AI를 통한 향상된 탐지 성능 등을 제공
Fossology	<input type="checkbox"/> 소스 코드 파일 상단의 라이선스 문자열과 저작권 정보를 검출하여 보고서 제공 <input type="checkbox"/> 오픈소스 소프트웨어 라이선스 및 저작권 정보 분석에 유용
Olive Platform	<input type="checkbox"/> 깃헙 프로젝트를 분석하여 오픈소스 데이터를 관리하고 라이선스 및 의무사항을 확인하여 보고서 제공 <input type="checkbox"/> 의존성과 라이선스를 확인할 수 있는 심플체크 기능을 제공하고 있으며 사용자 편의에 초점을 둔 구성

Snyk	<input type="checkbox"/> 오픈소스 소프트웨어 라이선스와 관련하여 의존성 뷰어, 이슈 우선 순위 선별, 런타임 모니터링 등의 기능을 제공
SPDX	<input type="checkbox"/> 소프트웨어 패키지 데이터 교환(Software Package Data Exchange)의 약자로서 구성요소, 라이선스, 저작권 및 보안 관련 사항을 포함하여 소프트웨어 BOM(Bill of Materials) 정보를 전달하기 위한 공개 표준 <input type="checkbox"/> SPDX 문서를 업로드 하여 구문을 분석하여 유효성 검사 및 라이선스 목록을 검색할 수 있는 온라인 포털 제공
SW360	<input type="checkbox"/> 소프트웨어 구성요소 관리 도구 <input type="checkbox"/> 제품에 사용된 컴포넌트 추적, 보안취약점 평가, 라이선스 의무 관리, 고지문 등 법적 문서 생성 기능 제공
White Source	<input type="checkbox"/> 방대한 데이터베이스에 기반을 둔 라이선스 준수 및 취약점 관리 서비스를 제공하며, 컨테이너 및 서버리스 등 다양한 환경을 지원
Sparrow SCA	<input type="checkbox"/> 오픈소스 소프트웨어 라이선스 식별 및 보안취약점 진단 도구 <input type="checkbox"/> 소스코드, 바이너리 파일 분석 및 오픈소스 소프트웨어 소스코드 일부만 가져오는 스니펫 분석 지원 기능 제공
VERACODE SCA	<input type="checkbox"/> 오픈소스 소프트웨어내 소스 코드에서 발생할 수 있는 보안취약점을 탐지하여 보안 위협 제거 기능 제공

별첨3

식별 및 관리 운영 사례

■ 소스코드 저장소의 오픈소스 목록 식별 자동화

- 오픈소스 라이선스 및 보안취약점 식별을 위한 오픈소스 분석 도구를 통해 자동으로 오픈소스를 식별하는 환경을 구성하여 정기적으로 수행 가능
- 오픈소스 분석도구를 구성하여 사내 임직원 모두 개발단계에서부터 오픈소스 라이선스와 보안취약점 등 이슈 사항을 조기 식별하도록 지원

■ 개발자 대상 교육 및 정보 공유

- 사내 개발자를 대상으로 오픈소스 라이선스와 보안취약점 이슈에 대한 주기적 교육 시행
- 개발자는 오픈소스 사용전 오픈소스 담당자에게 해당 오픈소스 사용 관련 문의 및 정보 공유
- 사내 오픈소스 정책, 담당 조직 및 가이드 문서를 배포하여 누구나 확인할 수 있도록 공유

■ 오픈소스 문의 게시판을 이용한 가이드

- 별도 오픈소스 문의 게시판 운영을 통해 개발자의 오픈소스 문의 응대
 - ※ (예) 특정 오픈소스 사용시 해당 버전에 보안취약점 이슈가 있어, 특정 버전 업데이트를 가이드
- 오픈소스 매니저와 보안담당자가 오픈소스 라이선스와 보안위협에 대한 피드백 제공

별첨4

오픈소스 소프트웨어 관련 국내외 가이드 및 표준

국내 가이드

- 공공 공개소프트웨어 거버넌스 가이드(과학기술정보통신부·정보통신산업진흥원, 2021)
- 공개소프트웨어 라이선스 가이드(과학기술정보통신부·정보통신산업진흥원, 2021)
- 기업 공개소프트웨어 거버넌스 가이드(과학기술정보통신부·정보통신산업진흥원, 2021)
- 개방형OS 도입 가이드(과학기술정보통신부·정보통신산업진흥원, 2021)
- 공개SW를 활용한 소프트웨어 개발보안 점검가이드(행정안전부·한국인터넷진흥원, 2019)

국외 가이드

- The Minimum Elements For a Software Bill of Materials (SBOM) (NTIA, 2021)
- Federal Source Code Policy(Federal CIO Council, 2016)
- Source Code Policy: Open and shared(CFPB, 2012)
- Open Source Software Strategy(EU Commission, 2020)
- Open Source Software Guidelines(EU JRC, 2010)
- The Technology Code of Practice(UK CDDO, 2021)
- Open source software guideline(Queensland Government, 2020)
- Ensure Safe and Successful Usage of Open-Source Software With a Comprehensive Governance Policy(Gartner, 2020)
- Start With These Three Best Practices to Maximize Open-Source Software Value(Gartner, 2019)
- Market Guide for Software Composition Analysis(Gartner, 2021)

국내 표준

- 공개소프트웨어 보안취약점 관리 지침(TTA, 2019)
- 소프트웨어 공급망의 라이선스 거버넌스를 위한 공개소프트웨어 컴플라이언스 지침 (TTA, 2018)
- 공개소프트웨어 라이선스 정책 수립 모델(TTA, 2018)
- 공공 공개소프트웨어 거버넌스 지침(TTA, 2017)
- 공개 소프트웨어 거버넌스 프레임워크(2015)
- 공개 소프트웨어 정보 교환 명세(TTA, 2014)
- 공개소프트웨어 분류 체계 및 프로파일(TTA, 2011)

국외 표준

- ISO/IEC 5230:2020 Information Technology – OpenChain Specification
- ISO/IEC 19770-2:2015 IT asset management – Part2: Software identification tag
- Software Package Data Exchange Specification Version 2.2(SPDX, 2020)

금융분야 오픈소스 소프트웨어 활용·관리 안내서

발행일 : 2022년 12월

발행인 : 김 철 웅

발행처 : 금융보안원

경기도 용인시 수지구 대지로 132

전 화 : (02) 3495-9000

〈비 매 품〉

본 안내서 내용의 무단전재를 금하며, 가공 인용할 때에는 반드시 금융보안원
「금융권 오픈소스 소프트웨어 활용·관리 안내서」라고 밝혀 주시기 바랍니다.

금융분야 오픈소스 소프트웨어 활용·관리 안내서