TTA Standard

정보통신단체표준(국문표준)

TTAx.xx-xx.xxxx 제정일: 2019년 XX월 XX일

공개소프트웨어 보안취약점 관리 지침

Open Source Software Management Guideline for Security Vulnerability

한국정보통신기술협회
Telecommunications Technology Association

표준초안 검토 위원회 공개소프트웨어 프로젝트그룹(PG602)

표준안 심의 위원회 소프트웨어/컨텐츠 기술위원회(TC6)

	성명	소 속	직위	위원회 및 직위	표준번호
표준(과제) 제안	김병선	㈜비디에스케이	전무	PG602 의장	
표준 초안 작성자	김병선	㈜비디에스케이	전무	PG602 의장	TTAK.KO-xx.xxxx
	김근동	락플레이스	이사	PG602 부의장	
	강신각	ETRI	센터장	PG602 위원	
	류원옥	ETRI	책임	PG602 위원	
	김형채	인베슘	대표	PG602 위원	
	박정현	ETRI	책임	PG602 위원	
	박철웅	SK텔레콤(주)	부장	PG602 위원	
사무국 담당	김재웅	TTA	단장	-	
	민선미	TTA	책임	-	

본 표준 발간 이전에 접수된 지식재산권 확약서 정보는 본 표준의 '부록(지식재산권 확약서 정보)'에 명시하고 있으며, 이후 접수된 지식재산권 확약서는 TTA 웹사이트에서 확인할 수 있습니다.
본 표준과 관련하여 접수된 확약서 외의 지식재산권이 존재할 수 있습니다.

발행인 : 한국정보통신기술협회 회장
발행처 : 한국정보통신기술협회
13591, 경기도 성남시 분당구 분당로 47
Tel : 031-724-0114, Fax : 031-724-0109
발행일 : 20xx.xx

서 문

1 표준의 목적

이 표준의 목적은 국내 기업 및 연구소, 단체 등에서 공개소프트웨어를 개발 혹은 사용함에 있어서 공개소프트웨어의 보안취약점을 조기에 발견하고 위협요소를 제거함으로서 보안취약점으로 인한 피해를 예방하고 효과적으로 관리할 수 있는 절차 및 관리 요소에 대한 표준을 제시함에 있다.

2 주요 내용 요약

최근 공개소프트웨어 개발 및 사용이 산업 전반에 걸쳐 급증하고 있는 반면 공개소프트웨어 보안취약점으로 인한 피해 및 이슈 또한 증가하고 있다. 본 표준은 이러한 어려움을 해소하고 산업에서 공개소프트웨어를 보다 안전하고 유연하게 개발 및 사용할 수 있도록 먼저 공개소프트웨어 보안취약점의 개요 및 특징과 함께 보안취약점으로 인한 피해 사례, 조치방법 및 관리절차에 대한 기준을 제시한다.

3 인용 표준과의 비교

3.1 인용 표준과의 관련성

– 해당 사항 없음.

3.2 인용 표준과 본 표준의 비교표

– 해당 사항 없음.

Preface

1 Purpose

The standard is to provide the guideline and management elements to identify the security vulnerabilities of open software early in developing or using open software in domestic companies, research institutes, organizations, etc., and to remove the threats, thereby preventing and effectively managing the damage caused by security vulnerabilities. And to provide standards for management elements.

2 Summary

Recently, the development and use of open software has increased rapidly throughout the industry, while the damage and issues caused by open software security vulnerabilities are also increasing. In order to solve these difficulties and to develop and use open software more securely and flexibly in the industry, this standard first provides an outline and characteristics of open software vulnerabilities, as well as the criteria for damage cases caused by security vulnerabilities.

3 Relationship to Reference Standards

- None.

목 차

공개소프트웨어 보안취약점 관리 지침
(Open Source Software Management Guideline for Security Vulnerability)

1 적용 범위

본 표준은 전 산업 업종, 제품 및 서비스에 관계없이 내부에서 자체 개발한 소프트웨어 혹은 내부에서 자체 개발하고 일부 공개소프트웨어를 사용한 소프트웨어의 소스 코드를 외부에 공개 시 검토해야 할 보안취약점 관리 지침을 제시한다.

2 인용 표준

– 해당 사항 없음.

3 용어 정의

3.1 공개소프트웨어(公開-, open software)

누구나 자유롭게 사용하고 수정하거나 재배포할 수 있도록 공개하는 소프트웨어. 누구에게나 이용과 복제, 배포가 자유롭고, 특히 소스 코드에 대한 접근을 통하여 개작과 재배포가 자유롭다는 뜻이나 무료와 혼동할 수 있어 'free' 대신에 'open'을 공식적으로 사용한다. 공개소프트웨어라도 공개소프트웨어 본래 의미를 유지하기 위해 다양한 라이선스 정책을 만들어 이를 지키도록 요구하고 있다. 따라서 상업적인 목적으로 공개소프트웨어를 사용하려고 할 때에는 사전에 라이선스의 각 조항들을 검토할 필요가 있다. 공개 소스 소프트웨어와 같은 의미로 사용된다.

3.2 보안취약점(security vulnerability)

시스템 또는 프로그램에 내재되어 있는 버그(잘못된 부분)를 의미하며, 해커는 이를 악용해 시스템에 침입하여 정보 유출, 시스템 파괴 등 유발.

[출처 : TTA 단체표준 TTAK.KO-12.0002/R3 정보 보호 기술 용어]

4 약어

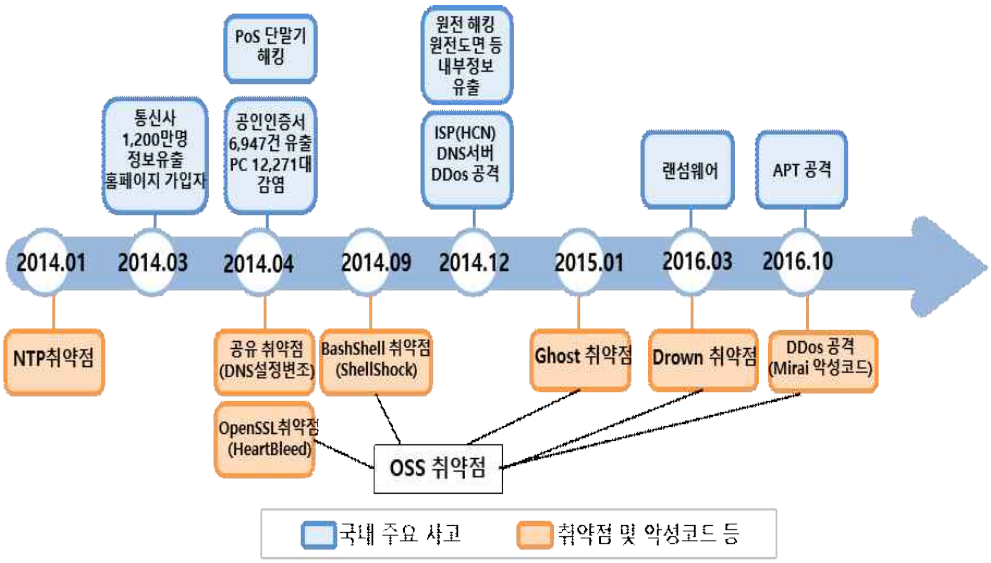
4.1 NVD(National Vulnerability Database): 미국 정부 산하 국립표준기술연구소

(National Institute of Standards and Technology, NIST)에서 운영하는 공개 취약점 DB

5 공개소프트웨어 보안취약점 현황

일반적으로 공개소프트웨어 개발 및 관리에는 수많은 이해관계자들이 참여하므로 코드의 품질이 더 높고, 보안상 안전할 것으로 인식되지만, 다양한 보안 위협에 노출되어 있는 것이 사실이다.
특히, [그림 1]과 같이 2016년 3월에 Drown 취약점은 악성 패킷을 보내 인증서 키 값을 알아내고 해당 키 값을 이용해 암호화된 통신 내용을 복호화하여 주요 정보를 탈취하는 중간자 공격(Man-in-the-Middle attack)에 사용되었다. 2014년 이후 Heartbleed 및 Shellshock 등 공개소프트웨어 보안취약점으로 인한 해킹 사고가 이어지고 있다. 최근 공개소프트웨어로부터 고위험성 취약점이 발견되고 있으며, 실제로 2014년 미국의 NVD에 새롭게 등록된 7,937개의 취약점 중에서 약 4,300개가 공개소프트웨어에서 발견된 취약점이었다.

또한, 국가정보원에서 발표한 「2015 국가정보보호백서」에서는 공개소프트웨어에 존재하는 위험성 높은 취약점으로 인해 보안위협이 크게 고조되었음을 강조하였고, 트렌드마이크로(TrendMicro)의 2015년 보안 예측 보고서에서는 공개소프트웨어에 대한 취약점 악용 시도가 더 많아질 것으로 예측하기도 하였다.



[그림 1] 최근 3년간 주요 침해사고

6 공개소프트웨어에서 발생 가능한 보안취약점

소프트웨어의 공통적인 보안취약점은 보안요구사항이 정의되지 않았거나, 논리적인 오류를 가지는 설계를 수행하였거나, 기술취약점을 가지는 코딩 규칙을 적용하였거나, 소프트

웨어 배치가 적절하지 않았거나, 발견된 취약점에 대해 적절한 관리 또는 패치를 하지 않은 경우 발견된다. 따라서 공개소프트웨어 에서도 발생 가능한 보안취약점을 정리하면 다음과 같다.

6.1 공개된 소스코드로 인한 위협

소스코드가 공개된 만큼 공격자 입장에서도 공격 대상 선정 및 악의적인 역분석이 매우 용이하다. 만약 공개소프트웨어에 대한 가시성을 확보하지 못한다면 사용된 공개소프트웨어 컴포넌트에 알려진 보안취약점을 타겟으로 한 공격에 대응이 불가하다.

6.2 관리적 보안 위협

관리 소홀로 인해 체계적인 버전 관리가 이뤄지지 않거나, 보안 패치가 제공되더라도 패치 미적용으로 인해 계속해서 보안 위협에 노출될 가능성이 있다. 또한, 소프트웨어 개발 시, 공개소프트웨어 사용 여부가 확인되지 않거나, 소스코드를 확보하기 어려운 외주 개발 모듈에 대해서는 공개소프트웨어의 사용 여부조차 파악하기 어렵다.

7 공개소프트웨어 보안 취약점의 특징

7.1 공개된 환경 제공

공개소프트웨어는 보안 공격자들에게 쉬운 공격대상의 환경을 제공한다. 공개소프트웨어는 상용 및 내부 사용 어플리케이션에 전반적으로 사용되고 있음에 따라 보안 취약점이 노출될 경우 보안 공격자들에게 쉬운 공격대상의 환경을 제공하게 된다.

7.2 공개적 노출

공개소프트웨어의 보안취약점은 NVD, 프로젝트 메일링 리스트, 프로젝트 홈페이지 등의 소스에 공개적으로 노출되어 있어 비 전문가도 손쉽게 이를 재현할 수 있다.

7.3 지원 모델

공개소프트웨어는 일반적인 상용 소프트웨어와 달리 사용자가 직접 사용하는 공개소프트웨어에 대한 보안취약점, 패치 및 업데이트를 모니터링해야 하는 Pull 서포트 모델을 가지고 있어 사용자의 모니터링이 매우 중요하다.

7.4 다양한 유입 경로

공개소프트웨어는 개발자가 직접 다운로드, 상용 소프트웨어에 포함, 써드파티 라이브러리에 포함, 외주개발에 사용 등 다양한 유입 경로를 가지고 있음에 따라 공개소프트웨어에 대한 가시성이 확보되지 못할 경우, 사용된 공개소프트웨어 컴포넌트에 알려진 보안 취약점을 타겟으로 한 공격에 대응이 불가능 해진다.

8 공개소프트웨어 개발 생명주기

8.1 공개소프트웨어 개발 생명주기 개요

소프트웨어 개발 생명주기(SDLC, Software Development Life Cycle)는 소프트웨어의 생성에서 소멸까지의 과정을 단계별로 나눈 것으로, 각 단계별 주요활동과 산출물을 통해 프로젝트의 진행방향을 명확하게 파악하고, 관리를 용이하게 한다. 이처럼 공개소프트웨어 사용 또한 소프트웨어 범주 내에 포함되기 때문에 현재 가장 많이 사용하고 있고, 기존의 소프트웨어 프로세스와 일관되도록 기존의 SDLC를 적용하는 것이 바람직하다고 판단한다. 즉, 공개소프트웨어 보안 위협을 완화 및 개선하기 위한 가장 바람직한 방안은 기존의 소프트웨어 개발 생명주기별로 요구되는 보안활동을 적용 및 수행함으로써 안전한 소프트웨어를 개발 및 활용 할 수 있다.

8.2 공개소프트웨어 개발 생명주기의 목적

공개소프트웨어 개발 생명주기의 주요 목적은 공개소프트웨어를 효과적으로 활용하고, 추가적으로 공개소프트웨어 사용에 따른 다양한 위험 요소를 제거 및 예방하는 것이다. 이에 따라 일반적인 SDLC의 소프트웨어개발 설계 단계부터 개발 및 배포 후 유지보수 단계까지 SDLC 전 단계에 걸쳐 공개소프트웨어 사용에 따른 계획 및 관리가 수행되어야 한다.

8.3 공개소프트웨어 개발 생명주기 구성내용

- 설계 단계

무엇(What)을 처리하는 공개소프트웨어를 개발 및 활용 할 것인지 정의하는 단계로, 타당성 검토, 개발 계획, 요구사항 분석으로 분류한다.

- 구현 단계

어떻게(How)에 초점을 두고 실제로 공개소프트웨어를 개발 및 활용하는 단계로, 개발,

패키징, 테스트, 배포로 분류한다.

- 유지보수 단계

소프트웨어를 직접 운영하며, 변경(Change)에 초점을 두고 여러 환경변화에 따라 공개소프트웨어를 적응 및 유지시키는 단계이다.

- 공개소프트웨어 관리 단계

공개소프트웨어 관리 단계에서는 상단의 설계, 구현, 유지보수의 목표와 전략에 따라 반드시 지켜야 할 규정과 지침을 수립해야 한다. 조사, 분석, 평가를 통해 공개소프트웨어 적용과 활용을 위한 정책과 효율적인 조직 구성과 역할 및 책임에 따른 운영 방안을 제시하게 된다. 또한, 라이선스 의무사항 준수 및 법적 문제를 해결하며, 공개소프트웨어 도입, 활용, 배포에 대한 지식과 기술을 향상시키기 위하여 공개소프트웨어의 상황을 파악하는 모니터링과 피드백을 제시한다. 안전한 소프트웨어는 보안관련 기능을 수행하는 소프트웨어가 아니라, 신뢰성이 위협받는 상황에서도 시스템을 신뢰할 수 있는 상태로 유지할 수 있도록 만들어진 소프트웨어이다. 소프트웨어 개발 보안 방법론에서는 안전한 소프트웨어 개발을 위해 소프트웨어 개발 생명주기(SDLC)에 걸쳐 보안활동을 추가하고 있다.

9 공개소프트웨어 개발 생명주기별 보안 취약점 점검 사항

소프트웨어 개발 생명주기별 활동을 기반으로 안전한 공개소프트웨어 사용을 위한 활동들을 단계별로 정리하였다. 기존의 소프트웨어 개발 생명주기에 따른 보안취약점을 진단 및 제거하는 보안활동과 함께 공개소프트웨어 관련 보안 관리 부분을 병행함으로서 공개소프트웨어 개발보안 체계의 확립을 도모할 수 있다.

9.1 요구사항분석 단계

개발대상 소프트웨어가 다루는 정보와 소프트웨어가 제공하는 서비스를 대상으로 보안위협을 도출하여, 도출된 보안위협을 대응(제거, 완화, 회피 등)하기 위한 보안요구사항을 도출하는 단계이다.

- **기본 활동**: 사용자의 문제를 구체적으로 이해하고 소프트웨어가 담당해야 하는 정보영역을 정의한다. 사용자의 기능, 성능, 신뢰도 등에 대한 요구는 요구사항 정의서(Requirements Specifications)로 문서화한다. 요구사항 정의서는 다시 세분화하여 업무, 기술, 성능, 운영 요구사항 정의서로 분석할 수 도 있다. 이 단계

에서는 요구사항 정의서 이외에도 기능차트, 프로세스 정의서, 인터페이스 정의서와 같은 산출물이 생성된다.

- **보안 활동**: 요구사항 중 보안항목 요구사항을 식별하는 보안활동을 추가한다. 어떤 정보들이 시스템화 되어 관리되는지, 이 정보의 보안등급의 기준은 어떻게 세워야 하는지의 점검 작업이 요구된다.

- **공개소프트웨어 관련 추가적인 활동**: 공개소프트웨어를 사용하려는 의도와 적절성 및 모니터링을 위하여, 우선적으로 공개소프트웨어 검토위원회(Open Source Review Board, OSBR)나 컴플라이언스 팀에 알려야 한다. 이에 따라 공개소프트웨어 사용 요청과 일치하지 않는 소스 코드 저장소에 들어있는 공개소프트웨어 컴포넌트를 검사한다. 전체 플랫폼 스캔을 통해 문제점이 있는 공개소프트웨어를 발견하거나 제품 코드 모듈 감사를 한다. 별도의 저장소로 분리하여 관리하는 것을 권장하며, 소스를 검사할 때마다 소스코드 제어 시스템의 알림을 설정하는 것이 관리를 용이하게 만든다.

9.2 설계 단계

보안위협을 기반으로 도출된 보안요구사항이 소프트웨어 기능으로 구현되도록 보안 구조 및 인터페이스, 알고리즘 등을 설계하는 단계이다.

- **기본 활동**: 소프트웨어의 구조와 그 성분을 명확하게 밝혀 구현을 준비하는 단계이다. 외부 시스템 및 사용자와의 인터페이스를 중시하는 외부설계와 시스템 내부를 설계하는 내부설계로 분류하기도 하고 전체적 구조와 데이터 알고리즘을 설계하는 단계를 분리해 기본설계와 상세설계로 분류하기도 한다. 설계 단계는 설계사양서(Design Specification)를 산출물로 만들어 내며, 이 산출물과 요구사양서(Required Specification)를 토대로 사용자 지침서와 시험 계획서를 작성한다. 이 단계에서는 화면 설계서, ERD, 테이블목록, 테이블 정의서, 프로그램 목록, 개발표준 정의서, 단위테스트 시나리오, 통합테스트 시나리오와 같은 산출물이 생성된다.
- **보안 활동**: 시스템을 분석해 위협들을 도출해내는 위협 모델링, 보안통제 기준 설정과 같이 개발보안 가이드가 제시하는 작업을 기존 개발 프로세스에 추가해 진행한다. 특히 설계 단계에서 수행되는 위협 모델링 작업을 통해 최대한 많은 위협들을 도출해 해당 위협들이 충분히 제거될 수 있도록 시스템이 설계되어야

한다.

- **공개소프트웨어 관련 추가적인 활동**: 공개소프트웨어사용명세서를 작성하여 소스코드의 출처 및 라이선스, 보안취약점 히스토리를 검토한다. 공개소프트웨어사용명세서(Bill of Materials, BOM)는 어떤 소프트웨어 및 시스템에 사용된 공개소프트웨어의 목록이며, 세부적으로는 사용된 공개소프트웨어의 명칭, 버전, 다운로드 위치(URL) 및 라이선스, 보안취약점 히스토리 등이 기록된다. 또한 공개소프트웨어사용 명세서를 작성한 후에 자동화 분석도구를 사용하여, 공개소프트웨어 사용 여부를 재검토한다. 즉, 개발자가 사용한 공개소프트웨어를 실수로 명세서에 누락할 경우를 대비한다. 또한, 보안취약점을 식별하는 감사보고서를 만들어, 플래그가 지정된 각 파일에 대한 보안취약점에 대한 해결법을 제시하여 통제할 수 있는 기준을 수립하여야 한다.

9.3 구현 단계

보안요구사항에 대한 설계 내용을 프로그래밍 언어로 보안 약점이 내재되지 않도록 안전하게 구현하는 단계로, 현재 국내에서 의무화가 적용된 단계이다.

- **기본 활동**: 프로그래밍을 하는 단계이다. 각 모듈의 코딩과 디버깅이 이루어지고 그 결과를 검증하는 단위테스트 또는 모듈테스트를 수행한다. 이 단계의 산출물로는 소스코드, 단위테스트 결과서, 결함· 오류보고서, 오류코드 정의서와 같은 산출물이 생성된다.

- **보안 활동**: 표준코딩정의서 또는 소프트웨어 개발보안 가이드를 모든 개발자들이 준수하여 개발하는 것이 중요하다. 또한, 구현 단계에서 단위테스트를 통해 소프트웨어가 가질 수 있는 보안취약점을 충분히 제거할 수 있도록 해야 하며, 코드 리뷰 또는 소스 코드 진단 작업을 통해 소스 코드 수준의 안정성이 보장되도록 하여야 한다.

- **공개소프트웨어 관련 추가적인 활동**: 공개소프트웨어 개발보안 가이드를 준수하여 개발할 때, 코드 리뷰 또는 아키텍처 검토를 통해 소스코드를 수정하여 소스코드의 종속성 문제를 해결하여야 한다. 해결이 되면 공개소프트웨어를 라이브러리에 등록하게 되는데, 이는 소프트웨어 이름, 버전, 내부 소유자 및 제품 이름,

버전, 릴리스 번호, 컴포넌트가 사용되는 위치의 세부정보를 포함한다. 또한, 공개소프트웨어 사용 시 상단의 내용을 문서화하여 최종사용자에게 제공하여야 한다.

9.4 테스트 단계

보안요구사항에 따른 보안기능 등이 적절하게 동작하는지를 단위 모듈 서비스 시험, 통합 모듈 서비스 시험 등을 통해 확인하는 단계이다.

- **기본 활동**: 개발된 모듈들을 통합하여 시험하는 통합테스트, 완성된 시스템으로서 요구사항을 완벽 하게 구현했는지를 확인하기 위한 시스템테스트, 그리고 사용자가 직접 자신의 사용현장에서 검증해 보는 인수테스트 등을 수행한다. 이 단계에서는 통합테스트 결과서, 시스템 이행계획서와 같은 산출 물이 생성된다.

- **보안 활동**: 설계 단계에서 수행된 위협모델링을 통해 도출된 위협들이 구현 단계에서 해당 취약점들이 없는 애플리케이션으로 개발되었는지를 동적 분석 도구를 이용하거나 모의 침투테스트를 통해 검증하는 작업을 수행하여야 한다.

- **공개소프트웨어관련 추가적인 활동**: 공개소프트웨어 주석 검토를 통해 부적절한 내용을 제거하며, 배포 방법과 배포할 패키지 유형 및 배포 메커니즘을 결정한다. 배포 예정인 공개소프트웨어 패키지를 진단 및 개선을 통해 문제가 발견되지 않도록 하며, 별도의 문제가 없다면 공개소프트웨어 패키지를 업로드 할 준비를 한다.

9.5 유지보수 단계

테스트 단계를 통해 보안요구사항이 적정하게 구현된 것을 확인된 경우, 소프트웨어 컴파일 등을 통한 배포본을 제작하는 단계이다. 소프트웨어 배포본을 설치하여 정상적인 IT 서비스를 제공하는 단계로 주기적인 보안취약점을 점검하여 안전한 서비스를 제공해야 한다. 보안취약점이 발생하는 경우 관련 보안패치를 적용하여 안전한 서비스를 제공해야 한다.

- **기본 활동**: 소프트웨어를 직접 이용하고 이용상에 나타나는 문제점을 수정하거나 새로운 기능을 추가 하여 보다 안정적인 소프트웨어로 발전시키기 위한 작업을 수행한다.

- **보안 활동**: 각 개발 단계에서 안전한 소프트웨어를 만들기 위해 노력하였음에도 불구하고 발생될 수 있는 보안사고에 대한 관리 및 사고대응, 패치관리가 병행되어야 한다.

- **공개소프트웨어 관련 추가적인 활동**: 배포용 사이트에 올바르게 업로드가 되었는지 확인을 해야 하며, 배포와 관련한 최종 검증을 수행한다. 최종사용자에게 제품 설명서에 적절한 고지사항이 포함되었는지 여부를 파악하고, 소스코드의 사본을 얻는 방법을 알려준다.

10 공개소프트웨어 보안 취약점 관리를 위한 조직 구성

안전한 공개소프트웨어 활용과 개발을 위해서는 프로젝트에 참여하는 구성원들에게 각각의 직무별 보안활동을 정의하여 프로젝트가 수행되는 동안 책임감을 가지고 보안활동을 수행하도록 하는 것이 필요하다.

10.1 공개소프트웨어 검토위원회(Open Source Review Board, OSRB)

공개소프트웨어 관리에 사용되는 정책, 프로세스, 지침, 양식 관리 생성 및 유지 관리한다. 아키텍처 검토를 통해 공개소프트웨어 코드를 분석하고, 공개소프트웨어 보안 위협을 통제하기 위한 방안을 수립한다.

10.2 공개소프트웨어 집행위원회(Open Source Executive Committee)

공개소프트웨어 전략 수립 및 보안 감사를 수행하며 버전 릴리스 검토 및 최종 승인을 한다.

10.3 구현 개발자(Implementer)

공개소프트웨어에 대한 사용 양식을 제출하여 오픈소스의 용도를 명확하게 설명해야 한

다. 공개소프트웨어를 모니터링하여 수정된 컴포넌트에 대한 변경 이력이나 로그를 관리하여야 하며 고도로 구조화된 개발 환경에서 프로그램을 구현하기 위해 시큐어코딩 표준을 준수하여 개발하여야 한다. 제 3자가 소프트웨어 안전 여부를 쉽게 판단할 수 있도록 문서화해야 한다.

10.4 IT기획자(IT Designer)

보안 관리에 사용되는 도구 및 자동화 인프라를 지원한다. 보안 관리 활동의 효율을 높이기 위한 도구를 도입 및 습득한다.

10.5 프로젝트 관리자(Project Manager)

프로젝트 관리자는 팀 구성원들에게 응용프로그램 보안전략을 알려야 한다. 프로젝트 일정과 보안위험의 상관관계 등과 같은 응용프로그램 보안영향을 이해시키며, 조직의 상태를 모니터링 한다. 기본적인 비즈니스 매트릭스 조합을 정의하고 단계별로 적용한다.

11 공개소프트웨어 보안 취약점 관리 프로세스

11.1 환경분석

요구사항 분석, 자산현황분석, 진단대상 선정, 수행계획 수립을 진행하며 산출물로는 수행계획서 자산 대장이다.

11.2 현황진단

공개소프트웨어 현황분석, 공개소프트웨어 사용목록, 관리현황 분석, Gap분석을 수행하며 산출물로는 공개소프트웨어 구성요소 목록 및 Gap분석 보고서이다.

11.3 위험분석

공개소프트웨어 취약점 분석, 보안 위험식별, 보안위험 평가, 취약결과 분석을 수행하며 산출물로는 취약점 진단서 및 위험평가 지수이다.

11.4 대책수립

보안대책 도출, 이행계획 수립, 공개소프트웨어 정책 수립, 마스터 플랜 수립을 수행하며 산출물로는 마스터 플랜 및 공개소프트웨어 가이드이다.

11.5 이행관리

솔루션 구현, 이행 점검, 공개소프트웨어 보안 교육, 운영관리를 수행하며 산출물로는 점검결과서, 취약점 이력관리 대장이다.

[그림 2] 공개소프트웨어 보안 취약점 관리 프로세스

부 록 Ⅰ-1

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

지식재산권 확약서 정보

Ⅰ-1.1 지식재산권 확약서

– 해당 사항 없음.

※ 상기 기재된 지식재산권 확약서 이외에도 본 표준이 발간된 후 접수된 확약서가 있을 수 있으니, TTA 웹사이트에서 확인하시기 바랍니다.

부 록 Ⅰ-2

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

시험인증 관련 사항

Ⅰ-2.1 시험인증 대상 여부

- 해당 사항 없음.

Ⅰ-2.2 시험표준 제정 현황

- 해당 사항 없음.

부 록 Ⅰ-3

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

본 표준의 연계(family) 표준

- 해당 사항 없음.

부 록 Ⅰ-4

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

참고 문헌

– 해당 사항 없음.

부 록 Ⅰ-5

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

영문표준 해설서

– 해당 사항 없음.

부 록 Ⅰ-6

(본 부록은 표준을 보충하기 위한 내용으로 표준의 일부는 아님)

표준의 이력

판수	채택일	표준번호	내용	담당 위원회
제1판	2019.xxxx	제정 TTAx.xx-xx.xxxx	-	공개소프트웨어 프로젝트그룹 (PG602)