㈜아이티아이즈 안전한 코딩방법은 클라우드 지원서비스를 제공하기 위한 운영문서로서 검토되고 승인됨	구분	직위	성명	일자	서명
	승인	클라우드서비스 최고책임자	조왕래		
	검토	클라우드서비스 관리자	박수용		
	작성	클라우드서비스 담당자	김선환		

안전한 코딩방법

2022.09.26.

IT eyes Inc.

재•개정 이력사항				
번호	**날짜**	**쪽**	**내용**	**담당자**
1	2022.09.26	24	안전한 코딩지침 Ver1.0	김선환

본 문서는 ㈜아이티아이스 클라우드 서비스 제공을 위해서 컨설팅,
마이그레이션, 매니지드, XaaS 서비스 등을 대상으로 작성함.

제 1 장 총칙

제 1 조(목적)

본 지침은 행정안전부의 행정기관 및 공공기관 정보시스템 구축·운영 지침(2022.04.21 개정판) 제 6 장 제 50 조에 의거하여 제품의 코딩 관리에 필요한 사항을 규정함을 목적으로 한다.

제 2 조(적용범위)

이 지침은 개발자에 의해 만들어지는 모든 소프트웨어에 적용된다.

① 웹 시큐어코딩 점검기준에서는 행정안전부에서 개정한 전자정부 SW 개발·운영자를 위한 Java 시큐어코딩 가이드(2012.09 개정판), 전자정부 SW 개발·운영자를 위한 C 시큐어코딩 가이드(2012.09 개정판)을 참조, 웹 시큐어코딩 점검 도구는 행정안전부에서 개정한 전자정부 SW 개발자·진단원을 위한 공개 SW 를 활용한 소프트웨어 개발보안 점검 가이드(2019.06 개정판)을 기준으로 한다.

② 모바일 시큐어코딩에서는 행정안전부에서 개정한 모바일 대민서비스 보안취약점 점검 가이드(2021.12 개정판)을 기준으로 한다.

③ 빅데이터, 인공지능 플랫폼 시큐어코딩에서는 과학기술정보통신부에서 개정한 Python 시큐어코딩 가이드(2022.02 개정판)을 기준으로 한다.

제 3 조(용어정의)

이 지침에서 사용되는 용어 정의는 다음과 같다.

① “소프트웨어(SW) 개발보안”은 SW 개발과정에서 개발자 실수, 논리적 오류 등으로 인해 SW 에 내포될 수 있는 보안취약점(vulnerability)의 원인, 즉 보안약점(weakness)을 최소화하는 한편, 사이버 보안위협에 대응할 수 있는 안전한 SW 를 개발하기 위한 일련의 보안활동을 의미한다.

② “시큐어코딩(secure coding)”라 함은 소프트웨어(SW)를 개발함에 있어 개발자의 실수, 논리적 오류 등으로 인해 SW 에 내포될 수 있는 보안취약점(vulnerability)을 배제하기 위한 코딩 기법을 뜻 한다.

③ “전자정부서비스”라 함은 행정 업무의 전산화· 정보화로 신속 정확하고 효율적인 행정 서비스를 제공하는 것을 뜻한다.

④ “웹 애플리케이션”은 소프트웨어 공학적 관점에서 인터넷이나 인트라넷을 통해 웹 브라우저에서 이용할 수 있는 응용 소프트웨어를 말한다.

⑤ “C”란 미국의 벨(Bell) 연구소에서 개발된 범용 프로그래밍 언어를 말한다.

⑥ “Java”란 객체 지향 프로그래밍(실세계 현상을 컴퓨터상의 객체로 실현) 언어를 말한다.

⑦ “Python”란 네덜란드 수학자가 개발한 프로그래밍 언어 중 하나를 말한다.

⑧ “데이터”란 인간 또는 컴퓨터를 비롯한 자동 기기에 의해 행해지는 통신과 해석, 처리로 형식화된 사실과 개념, 명령을 표현한 것을 말한다.

⑨ “프로세스”란 컴퓨터 내에서 실행중인 프로그램을 말한다.

⑩ “캡슐화”는 객체의 자료와 행위를 하나로 묶고, 실제 구현 내용을 외부에 감추는 것을 말한다.

⑪ “API(application programming interface)”는 운영체제와 응용프로그램 사이의 통신에 사용되는 언어나 메시지 형식을 말한다.

⑫ ”단위테스트”는 프로그램의 기본 단위인 모듈을 테스트하여 정해진 기능을 정확히 수행하는지 확인하는 것을 말한다.

⑬ “통합테스트”는 단위테스트가 끝난 모듈을 통합하는 과정에서 발생할 수 잇는 오류를 찾는 테스트를 말한다.

⑭ “소스코드”는 디지털기기의 소프트웨어 내용을 프로그래밍 언어로 나타낸 설계도를 말한다.

⑮ “빅데이터”란 디지털 환경에서 생성되는 데이터를 말하며 형태도 수치 데이터 뿐 아니라 문자와 영상 데이터를 포함하는 대규모 데이터를 말한다.

⑯ “인공지능”은 인간의 학습능력과 추론능력, 지각능력, 자연언어의 이해능력 등을 컴퓨터 프로그램으로 실현한 기술을 말한다.

⑰ “플랫폼”은 컴퓨터 시스템의 기본이 되는 특정 프로세서 모델과 하나의 컴퓨터 시스템을 바탕으로 하는 운영체제를 말한다.

⑱ “정적 분석 도구”란 소스 코드의 실행 없이 코드의 의미를 분석해 결함을 찾아내는 프로그램을 말한다.

제 2 장 웹 시큐어코딩 점검

제 4 조 웹 시큐어코딩의 개요

웹 소프트웨어 개발시 소프트웨어의 보안약점을 최소화하기 위해 웹 시큐어코딩 점검 지침을 제시한다.

제 5 조 대상 점검 범위 및 점검 시기

① 대상 점검 범위는 웹애플리케이션 서비스를 제공하기 위한 소프트웨어 소스를 말하며, 해당소스는 Java 또는 C 언어로 이루어진 점검기준을 제공한다.

② 해당 소프트웨어의 단위 테스트, 통합 테스트 시기에 점검 도구를 이용하여 점검을 진행한다.

제 6 조 Java 기반 시큐어코딩 점검 기준

① Java 언어로 개발된 웹은 아래의 보안 약점 항목들을 기준으로 검사를 실시한다.

1. 입력데이터 검증 및 표현은 프로그램 입력값에 대한 검증 누락 또는 부적절한 검증, 데이터의 잘못된 형식지정으로 인해 발생할 수 있는 보안약점(총 26 개)을 말한다.
2. 보안기능은 보안기능(인증, 접근제어, 기밀성, 암호화, 권한관리 등)을 적절하지 않게 구현시 발생할 수 있는 보안약점(총 24 개)을 말한다.
3. 시간 및 상태는 동시 또는 거의 동시 수행을 지원하는 병렬 시스템, 하나 이상의 프로세스가 동작하는 환경에서 시간 및 상태를 부적절하게 관리하여 발생할 수 있는 보안약점(총 7 개)을 말한다.
4. 에러처리는 에러를 처리하지 않거나, 불충분하게 처리하여 에러정보에 중요정보(시스템 등)가 포함될 때 발생할 수 있는 보안약점(총 4 개)을 말한다.
5. 코드오류는 타입변환 오류, 자원(메모리 등)의 부적절한 반환 등과 같이 개발자가 범할 수 있는 코딩오류로 인해 유발되는 보안약점(총 7 개)을 말한다.
6. 캡슐화는 중요한 데이터 또는 기능성을 불충분하게 캡슐화하였을 때, 인가되지 않는 사용자에게 데이터 누출이 가능해지는 보안약점(총 8 개)을 말한다.
7. API 오용은 의도된 사용에 반하는 방법으로 API 를 사용하거나, 보안에 취약한 API 를 사용하여 발생할 수 있는 보안약점(총 7 개)을 말한다.

제 7 조 C 기반 시큐어코딩 점검 기준

① C 언어로 개발된 웹은 아래의 보안 약점 항목들을 기준으로 검사를 실시한다.

1. 입력데이터 검증 및 표현은 프로그램 입력값에 대한 검증 누락 또는 부적절한 검증, 데이터의 잘못된 형식지정으로 인해 발생할 수 있는 보안약점(총 19 개)을 말한다.

2. 보안기능은 보안기능(인증, 접근제어, 기밀성, 암호화, 권한관리 등)을 적절하지 않게 구현시 발생할 수 있는 보안약점(총 17 개)을 말한다.

3. 시간 및 상태는 동시 또는 거의 동시 수행을 지원하는 병렬 시스템, 하나 이상의 프로세스가 동작하는 환경에서 시간 및 상태를 부적절하게 관리하여 발생할 수 잇는 보안약점(총 3 개)을 말한다.

4. 에러처리는 에러를 처리하지 않거나, 불충분하게 처리하여 에러정보에 중요정보(시스템 등)가 포함될 때 발생할 수 있는 보안약점(총 3 개)을 말한다.

5. 코드오류는 타입변환 오류, 자원(메모리 등)의 부적절한 반환 등과 같이 개발자가 범할 수 있는 코딩오류로 인해 유발되는 보안약점(총 2 개)을 말한다.

6. 캡슐화는 중요한 데이터 또는 기능성을 불충분하게 캡슐화하였을 때, 인가되지 않는 사용자에게 데이터 누출이 가능해지는 보안약점(총 2 개)을 말한다.

7. API 오용은 의도된 사용에 반하는 방법으로 API 를 사용하거나, 보안에 취약한 API 를 사용하여 발생할 수 있는 보안약점(총 5 개)을 말한다.

제 8 조 웹 시큐어 코딩 점검 도구

웹 시큐어 코딩 점검에 사용할 점검 도구를 제시한다. 각 도구의 점검 유형은 [첨부 1]을 참조하여 점검도구 선정한다.

① Spotbugs(SpotBugs Team, LGPL 라이센스, 정적 분석을 통해 코드에서 버그를 찾는 툴)

② FindSecurityBugs(SpotBugs Team, LGPL 라이센스, Spotbugs 의 기능 확장용 소프트웨어)

③ PMD(PMD,BSD 라이선스, 정적 분석을 통해 코드에서 버그를 찾는 툴)

④ Jenkins(키와쿠지 코스케 제작, 진단결과를 보고하는 오픈 소스 Continuous Integration 툴)

제 9 조 웹 시큐어 코딩 점검절차

① 요구사항(보안 요구사항)는 오용사례와 위험분석으로 설계보안항목에 대한 정의와 명세를 작성하고, 오용사례에 대한 정의 및 케이스 예시를 작성한다.

② 구조 설계(위험 분석)절차는 공격저항 분석(attack resistance analysis), 모호성 분석, 허점 분석 등으로 위험요소를

분석한다.

③ 테스트 계획는 공격 패턴, 위험 분석 결과, 악용 사례를 기반으로 위험기반 보안테스트를 수행한다.

④ 코드 검토 절차는 구현 오류(implementation bug)에 중점을 두며 특히 소스코드에 존재하는 취약성을 발견할 목적으로 수행되는 코드 정적분석에 중심을 둔다.

⑤ 테스트/테스트 결과(위험분석)는 위험 분석 및 침투 테스트를 수행한다. 침투테스트로 실제 작동 환경에서의 필드 소프트웨어에 대한 좋은 이해를 제공한다.

⑥ 현장과의 피드백절차에서는 보안 운영으로 얻은 공격자와 공격 도구에 대한 경험과 지식은 개발자에게 다시 피드백한다.

제 10 조 웹 시큐어 코딩 조치 후 사후점검

① 현장의 담당자는 점검 체크리스트[첨부 2]를 근거하여 조치 후 관리자에게 다시 보고한다.

제 3 장 모바일 시큐어 코딩 점검

제 11 조 모바일 시큐어코딩의 개요

모바일 소프트웨어 개발시 소프트웨어의 보안약점을 최소화하기 위해 모바일 시큐어코딩 점검 지침을 제시한다.

제 12 조 대상 점검 범위 및 점검 시기

① 대상 점검 범위는 모바일 애플리케이션 서비스를 제공하기 위한 소프트웨어 소스를 말하고 모바일 소스코드 보안약점 점검기준을 제공한다.

② 해당 소프트웨어의 단위 테스트, 통합 테스트 시기에 점검 도구를 이용하여 점검을 진행한다.

제 13 조 모바일 소스 점검 기준

① 소스코드에서 발생 할 수 있는 주요 보안 취약점을 아래의 점검 기준으로 점검한다.

1. 입력데이터 검증 및 표현은 프로그램 입력값에 대한 검증 누락 또는 부적절한 검증, 데이터의 잘못된

형식지정으로 인해 발생할 수 있는 보안약점(총 5개)을 말한다.

2. 보안기능은 보안기능(인증, 접근제어, 기밀성, 암호화, 권한 관리 등)을 부적절하게 구현 시 발생할 수 있는 보안약점(총 8개)을 말한다.

3. 시간 및 상태는 동시 또는 거의 동시 수행을 지원하는 병렬 시스템이나 하나 이상의 프로세스가 동작되는 환경에서 시간 및 상태를 부적절하게 관리하여 발생할 수 있는 보안약점(총 1개)을 말한다.

4. 에러처리는 에러를 처리하지 않거나, 불충분하게 처리하여 에러 정보에 중요정보(시스템 등)가 포함될 때 발생할 수 있는 보안약점(총 3개)을 말한다.

5. 코드오류는 타입변환 오류, 자원(메모리 등)의 부적절한 반환 등과 같이 개발자가 범할 수 있는 코딩오류로 인해 유발되는 보안약점(총 2개)을 말한다.

6. API 오용은 의도된 사용에 반하는 방법으로 API를 사용하거나, 보안에 취약한 API를 사용하여 발생할 수 있는 보안약점(총 1개)을 말한다.

7. 모바일 환경 특화는 공개영역(CWE, CWE/SANS Top25, OWASP Mobile Top 10 등) 관련 모바일 환경에 특화된 보안약점(총 6개)을 말한다.

제 14조 모바일 시큐어코딩 점검 도구

모바일 시큐어 코딩 점검에 사용할 점검 도구를 제시한다. 각 도구의 점검 유형은 첨부 1을 참고한다.

① Spotbugs(SpotBugs Team, LGPL 라이센스, 정적 분석을 통해 코드에서 버그를 찾는 툴)

② FindSecurityBugs(SpotBugs Team, LGPL 라이센스, Spotbugs의 기능 확장용 소프트웨어)

③ PMD(PMD,BSD 라이선스, 정적 분석을 통해 코드에서 버그를 찾는 툴)

④ Jenkins(키와쿠지 코스케 제작, 진단결과를 보고하는 오픈 소스 Continuous Integration 툴)

제 15조 모바일 시큐어 코딩 점검절차

① 요구사항(보안 요구사항)는 오용사례와 위험분석으로 설계보안항목에 대한 정의와 명세를 작성하고, 오용사례에 대한 정의 및 케이스 예시를 작성한다.

② 구조 설계(위험 분석)절차는 공격저항 분석(attack resistance analysis), 모호성 분석, 허점 분석 등으로 위험요소를 분석한다.

③ 테스트 계획는 공격 패턴, 위험 분석 결과, 악용 사례를 기반으로 위험기반 보안테스트를 수행한다.

④ 코드 검토 절차는 구현 오류(implementation bug)에 중점을 두며 특히 소스코드에 존재하는 취약성을 발견할 목적으로 수행되는 코드 정적분석에 중심을 둔다.

⑤ 테스트/테스트 결과(위험분석)는 위험 분석 및 침투 테스트를 수행한다. 침투테스트로 실제 작동 환경에서의 필드 소프트웨어에 대한 좋은 이해를 제공한다.

⑥ 현장과의 피드백절차에서는 보안 운영으로 얻은 공격자와 공격 도구에 대한 경험과 지식은 개발자에게 다시 피드백한다.

제 16 조 모바일 시큐어 코딩 조치 후 사후점검

① 현장의 담당자는 모바일 점검 체크리스트(첨부 3)를 참고하여 조치 후 관리자에게 다시 보고한다.

제 4 장 빅데이터,인공지능 플랫폼 시큐어코딩 점검

제 17 조 빅데이터, 인공지능 플랫폼 시큐어코딩 개요

빅데이터, 인공지능 플랫폼 시큐어코딩은 해당 환경에서 가장 많이 사용하는 언어인 Python 언어 기준으로 하고 개발시 보안약점을 최소화하기 위해 시큐어코딩 점검 지침을 제시한다.

제 18 조 대상 점검 범위 및 점검 시기

① 대상 점검 범위는 빅데이터, 인공지능 서비스를 제공하기 위한 소프트웨어 소스를 말하고 빅데이터, 인공지능 소프트웨어 소스코드 보안약점 점검기준을 제공한다.

② 해당 소프트웨어의 단위 테스트, 통합 테스트 시기에 점검 도구를 이용하여 점검을 진행한다.

제 19 조 Python 기반 시큐어코딩 점검 기준

① 주로 python 언어로 개발되는 빅데이터, 인공지능 플랫폼 환경은 아래의 보안 약점 항목들을 기준으로 검사를

실시한다.

1. 입력데이터 검증 및 표현은 프로그램 입력값에 대한 검증 누락 또는 부적절한 검증, 데이터의 잘못된 형식지정으로 인해 발생할 수 있는 보안약점(총 15개)을 말한다.
2. 보안기능은 보안기능(인증, 접근제어, 기밀성, 암호화, 권한관리 등)을 적절하지 않게 구현시 발생할 수 있는 보안약점(총 14개)을 말한다.
3. 시간 및 상태는 동시 또는 거의 동시 수행을 지원하는 병렬 시스템, 하나 이상의 프로세스가 동작하는 환경에서 시간 및 상태를 부적절하게 관리하여 발생할 수 있는 보안약점(총 2개)을 말한다.
4. 에러처리는 에러를 처리하지 않거나, 불충분하게 처리하여 에러정보에 중요정보(시스템 등)가 포함될 때 발생할 수 있는 보안약점(총 3개)을 말한다.
5. 코드오류는 타입변환 오류, 자원(메모리 등)의 부적절한 반환 등과 같이 개발자가 범할 수 있는 코딩오류로 인해 유발되는 보안약점(총 3개)을 말한다.
6. 캡슐화는 중요한 데이터 또는 기능성을 불충분하게 캡슐화하였을 때, 인가되지 않는 사용자에게 데이터 누출이 가능해지는 보안약점(총 4개)을 말한다.
7. API 오용은 의도된 사용에 반하는 방법으로 API를 사용하거나, 보안에 취약한 API를 사용하여 발생할 수 있는 보안약점(총 4개)을 말한다.

제 20조 빅데이터, 인공지능 플랫폼 시큐어 코딩 점검 도구

빅데이터, 인공지능 플랫폼에서 주로 사용하는 python 언어는 점검시 다른 소프트웨어를 사용하는 것이 아닌 해당 환경에 점검 프로젝트를 받아서 점검을 진행한다. 이에 python 의 프로젝트를 모아놓은 PyPI(Python Package Index)에서 주로 사용하는 점검 도구들를 제시한다. 각 도구의 점검 유형은 [첨부 4]을 근거로 점검 한다.

① Pyt(파이썬 테인, 오픈소스 정적 분석 도구)

② Bandit(일반적인 보안 위험 찾는 도구)

③ Pyntch(Python 2.x 만 지원, 정적 코드 분석 도구)

④ RATS(정적 코드분석 보안 관련 오류)

⑤ Requires.io(파이썬의 의존성 보안 모니터링)

⑥ Safety(파이썬 의존성 검사)

제 21 조 빅데이터, 인공지능 플랫폼 시큐어 코딩 점검절차

① 요구사항(보안 요구사항)는 오용사례와 위험분석으로 설계보안항목에 대한 정의와 명세를 작성하고, 오용사례에 대한 정의 및 케이스 예시를 작성한다.

② 구조 설계(위험 분석)절차는 공격저항 분석(attack resistance analysis), 모호성 분석, 허점 분석 등으로 위험요소를 분석한다.

③ 테스트 계획는 공격 패턴, 위험 분석 결과, 악용 사례를 기반으로 위험기반 보안테스트를 수행한다.

④ 코드 검토 절차는 구현 오류(implementation bug)에 중점을 두며 특히 소스코드에 존재하는 취약성을 발견할 목적으로 수행되는 코드 정적분석에 중심을 둔다.

⑤ 테스트/테스트 결과(위험분석)는 위험 분석 및 침투 테스트를 수행한다. 침투테스트로 실제 작동 환경에서의 필드 소프트웨어에 대한 좋은 이해를 제공한다.

⑥ 현장과의 피드백절차에서는 보안 운영으로 얻은 공격자와 공격 도구에 대한 경험과 지식은 개발자에게 다시 피드백한다.

제 22 조 빅데이터, 인공지능 플랫폼 시큐어 코딩 조치 후 사후점검

① 현장의 담당자는 모바일 점검 체크리스트[첨부 5]을 근거로 조치 후 관리자에게 다시 보고한다

[첨부 1] 점검 도구 점검 유형

① Spotbugs

번호	탐지 유형	사례 및 설명
1	Bad practice	클래스 명명규칙, null 처리 실수 등 개발자의 나쁜 습관을 탐지
2	Correctness	잘못된 상수, 무의미한 메소드 호출 등 문제의 소지가 있는 코드를 탐지
3	Dodgy code	숫자형의 곱셈결과를 문자형으로 변환하는 등 부정확하거나 오류를 발생시킬 수 있는 코드를 탐지
4	Experimental	메소드에서 생성된 stream 이나 리소스가 해제하지 못한 코드를 탐지
5	Internationalization	Default 인코딩을 지정하지 않은 경우 등 지역특성을 고려하지 않은 코드 탐지
6	Malicious code vulnerability	보안 코드에 취약한 가변적인 배열이나 콜렉션, Hashtable 탐지
7	Multithreaded correctness	멀티쓰레드에 안전하지 않은 객체 사용 등을 탐지
8	Performance	미사용 필드, 비효율적 객체생성 등 성능에 영향을 주는 코드를 탐지
9	Security	CSS, DB 패스워드 누락 등 보안에 취약한 코드를 탐지

② FindSecurityBugs

번호	탐지 유형	사례 및 설명
1	부적절한 입력 탐지	Untrusted Servlet parameter, session Cookie, Query String 등
2	SQL Injection 탐지	Hibernate, JDO, JPA, Spring JDBC, LDAP Injection 등
3	XSS	JSP, Servlet, JavaScript(Android) XSS 를 탐지
4	취약한 암호화 알고리즘	SHA-1, MD2/MD4/MD5 또는 NullCipher 등 취약한 암호화 함수를 탐지
5	취약한 URL Redirection	유효하지 않은 URL 로 Redirection 취약점 탐지
6	기타	보안 플래그가 없는 쿠키, Regex DOS 취약점, 신뢰할 수 없는 Context-type 또는 헤더

③ PMD

번호	탐지 유형	사례 및 설명
1	Basic (rulesets/basic.xml)	대부분의 개발자들이 동의하는 규칙
2	Naming (rulesets/naming.xml)	표준 자바 naming 규약을 위한 테스트
3	Unused code (rulesets/unusedcode.xml)	불필요한 코드 제거
4	Design (rulesets/design.xml)	디자인 원리 체크
5	Import statements (rulesets/imports.xml)	Import 문장에 대한 문제점들 점검
6	Junit tests (rulesets/junit.xml)	Test Case 와 Test Method 관련 특정 문제 검색
7	Strings (rulesets/string.xml)	스트링 관련 작업을 할 때 발생하는 일반적인 문제들 규명
8	Braces (rulesets/braces.xml)	for, if, while, else 문장이 괄호를 사용하는지 여부 검사
9	Code size (rulesets/codesize.xml)	과도하게 긴 메소드, 너무 많은 메소드를 가진 클래스 검사
10	Javabeans (rulesets/javabeans.xml)	직렬화 될 수 없는 bean 클래스 같이 JavaBeans 코딩 규약을 위배하는 JavaBeans 컴포넌트 검사
11	Finalizers	finalize() 메소드 관련한 다양한 문제들을 검사
12	Clone (rulesets/clone.xml)	clone() 메소드에 대한 규칙
13	Coupling (rulesets/coupleing.xml)	클래스들간 과도한 커플링 표시 검색
14	Strict exceptions (rulesets/strictexception.xml)	예외 테스트
15	Controversial (rulesets/controversial.xml)	좀 더 의심스러운 검사
16	Logging (rulesets/logging-java.xml)	java.util.logging.Logger 를 위험하게 사용하는 경우 검색

'[첨부 2] 웹 시큐어코딩 체크리스트

① Java

유형	체크사항	체크 여부
입력 데이터 검증 및 표현	데이터베이스와 연동된 웹 어플리케이션에서 입력된 데이터에 대한 유효성 검증 절차가 있는가	
	시스템 자원에 대한 식별자로 사용하는 경우 외부 입력값을 검증 절차가 있는가	
	외부 입력 내용이 동적 웹 페이지 생성에 사용될 경우 입력 내용에 검증 절차가 있는가	
	사용자 입력값이 운영체제 명령어의 일부 또는 전부로 구성되어 실행될 때 검증절차가 있는가	
	위험한 형식의 파일이 업로드가 실행되지 않도록 업로드에 제한이 있는가	
	신뢰되지 않는 url 주소 접속을 제한하는 절차가 있는가	
	XQuery 를 사용하여 xml 데이터에 대한 동적 쿼리문을 생성할 때 사용자의 외부 입력값에 대한 검증절차가 있는가	
	XPath 쿼리문 생성할 때 사용자의 외부 입력값에 대한 검증절차가 있는가	
	LDAP 명령어를 수행할 때 외부 입력값에 대한 검증절차가 있는가	
	사용자로부터 받은 요청에 대해서 사용자가 의도한 대로 작성되고 전송된 것인지 확인하는 절차가 있는가	
	외부 입력값에 대하여 디렉터리 경로 조작 문자열 검증절차가 있는가	
	HTTP 요청에 들어 있는 인자값을 검증하는 절차가 있는가	
	사용자의 입력값(정수)이 변수의 값 범위에 포함되는지 검증하는 절차가 있는가	
	보호 메커니즘을 우회하는 입력값 변조를 막는 절차가 있는가	
	Java Data Objects API 를 이용한 검사과정을 거치지 않는 문자열을 검증하는 절차가 있는가	
	J2EE Persistence API 를 이용한 검사과정을 거치지 않는 질의문을 검증하는 절차가 있는가	
	외부에서 입력된 값이 질의 명령어에 연결되는 문자열로 사용되지 않게 검증하는 절차가 있는가	
	외부에서 입력된 값이 LDAP 질의문의 내용을 변경되지 않게 검증하는 절차가 있는가	
	외부에서 시스템 설정이나 구성요소를 제어 할 수 없게 막아져 있는가	
	외부에서 입력되는 스크립트 문자열이 웹 페이지 생성에 사용되지 않게 검증하는 절차가 있는가	
	외부에서 입력되는 내용이 스크립트 또는 프로그램 명령어 문자열 생성에 사용되지 않게 검증하는 절차가 있는가	
	프로그램 내에서 라이브러리를 적재할 때 절대 경로를 사용하는가	
	외부의 입력되는 내용에서 의도하지 않은 클래스가 적재되지 않게 검증하는 절차가 있는가	
	원격으로 소스 코드 또는 실행파일을 다운로드 했을 때 무결성 검사를 하는 절차가	

	있는가	
	Hibernate API 를 이용한 검사과정을 거치지 않는 문자열을 검증하는 절차가 있는가	
	보안 결정이 신뢰할 수 없는 입력값(쿠키, 환경변수)으로 이루어지는 경우가 있는가	
보안기능	적절한 인증과정 없이 중요정보를 열람하는 경우가 있는가	
	SW 가 모든 가능한 실행경로에 대해 접근제어를 검사하고 있는가	
	SW 가 중요 보안관련 자원에 대하여 읽기, 수정 권한을 제한하고 있는가	
	취약한 암호화 알고리즘을 사용된 SW 가 있는가	
	프로그램이 보안과 관련된 민감 데이터를 송수신 할 경우 암호화를 하고 있는가	
	프로그램 코드 내부에 하드코드된 비밀번호가 있는가	
	길이가 짧은 키가 존재하는가(128 비트 미만)	
	예측 가능한 난수를 사용하는 경우가 있는가	
	비밀번호를 암호화되지 않는 텍스트 형태로 저장되는 경우가 있는가	
	코드에 하드코드된 암호화키가 있는가	
	취약한 패스워드를 허용하는가	
	사용자의 영속적인 쿠키에 민감 데이터가 저장되게 설정 되어 있는가	
	보안에 민감한 데이터를 브라우저 쿠키에 저장할 때 보안 속성이 세팅되게 설정 되어 있는가	
	주석문에 비밀번호가 기입되어 있는가	
	비밀번호를 솔트 없이 일방향으로 해쉬하여 저장하는가	
	소스 코드 또는 실행파일 다운로드시 무결성 검사를 하는가	
	사용자의 요청이 위조되지 않게 보안처리가 되어 있는가	
	세션이 일정 시간 흐르면 종료 되도록 설정되어 있는가	
	애플리케이션이 비정상적으로 중단 됐을 때, 메모리의 보안 관련 데이터가 외부에 노출 되지 않게 설정되어 있는가	
	SW 가 코드 내부에 고정된 사용자 계정 이름을 포함하고 있는가	
	RSA 알고리즘 사용시 OAEP 패딩을 사용하였는가	
	코드에 고정된 salt 값을 사용하고 있는가	
	다른 페이지로 민감한 정보를 보낼 때 POST 방식으로 파라미터를 전달하고 있는가	
	하나의 포트에 다수의 소켓이 연결되어 있는가	
시간 및 상태	공유자원을 여러 스레드가 접근하여 사용할 경우 한 번에 하나의 스레드만 접근 가능하도록 되어 있는가	
	제어문을 사용하지 않는 재귀함수가 있는가	
	정적 필드에 저장된 DB 연결이 있는가	
	서블릿의 필드에 사용자의 입력 데이터를 저장하고 있는가	
	J2EE 에서 스레드를 사용하는 곳이 있는가	
	심볼릭명을 사용하여 특정 대상을 지정하는 경우가 있는가	
	중복 검사된 잠금을 사용하는 곳이 있는가	
에러 처리	오류메시지에 민감한 정보를 포함하고 있는가	
	오류 발생시 예외처리가 잘 되어 있는가	

	프로그램 수행중 결과 값에 대한 적절한 처리 또는 예외상황에 대한 조건에 대한 검사가 잘 되어 있는가	
	사용자에게 비밀번호에 대한 강한 조건을 걸고 있는가	
코드 오류	null 이 될 수 있는 래퍼런스는 참조하기 전에 null 값인지 검사하는 절차가 있는가	
	사용이 완료된 자원을 제대로 반환하고 있는가	
	스레드의 notify()를 직접 호출하고 있는가	
	serialPersistenFields 를 사용할 때 private, static, final 로 선언하였는가	
	스레드의 run()대신 start()을 사용하였는가	
	하위 클래스에서 동기화된 메소드를 재정의해야 하는 경우, 상위 클래스와 동일하게 동기화된 메소드로 재정의하였는가	
	사용자가 사용할 수 있는 자원의 양을 제한하였는가	
캡슐화	다중 스레드 환경에서 정보를 저장하는 멤버변수가 포함되지 않도록 하였는가	
	디버그 코드가 남아있는가	
	시스템 데이터 정보가 노출되는 부분이 있는가	
	private 로 선언된 배열이 public 으로 선언된 메스드 통해 반환되는 경우가 있는가	
	public 으로 선언된 데이터 또는 메소드 인자가 private 선언된 배열에 저장되는 경우가 있는가	
	민감한 데이터를 가진 내부 클래스를 사용하는 경우가 있는가	
	변경되면 안 되는 public 멤버 변수들이 final 로 변경되어 있는가	
	동적으로 클래스를 로드하는 경우가 있는가	
API 오용	보안결정에서 도메인명을 이용한 DNS lookup 을 사용하는 경우가 있는가	
	J2EE 애플리케이션이 컨테이너에서 제공하는 연결 관리 기능을 사용하고 있는가	
	J2EE 애플리케이션이 소켓을 직접 사용하고 있는가	
	J2EE 프로그램에서 System.exit 를 사용하고 있는가	
	Object.equals(), Comparable.compareTo()과 Comparator.compare() 구현에서 매개변수를 Null 과 비교하여 지정된 값을 반환하고 있는가	
	EJB 프로그램에서 서버 소켓을 사용하고 있는가	
	한 클래스 내에서 equals()와 hashCode() 중 한쪽만 구현되어 있는가	

② C

유형	체크사항	체크 여부
입력 데이터 검증 및 표현	데이터베이스와 연동된 웹 어플리케이션에서 입력된 데이터에 대한 유효성 검증 절차가 있는가	
	시스템 자원에 대한 식별자로 사용하는 경우 외부 입력값을 검증 절차가 있는가	
	외부 입력 내용이 동적 웹 페이지 생성에 사용될 경우 입력 내용에 검증 절차가 있는가	
	사용자 입력값이 운영체제 명령어의 일부 또는 전부로 구성되어 실행될 때 검증절차가 있는가	

	LDAP 명령어를 수행할 때 외부 입력값에 대한 검증절차가 있는가	
	외부 입력값에 대하여 디렉터리 경로 조작 문자열 검증절차가 있는가	
	사용자의 입력값(정수)이 변수의 값 범위에 포함되는지 검증하는 절차가 있는가	
	보호 메커니즘을 우회하는 입력값 변조를 막는 절차가 있는가	
	스텍에 할당되는 버퍼들이 정의된 버퍼의 한계치를 넘지 않았는가	
	힙에 할당되는 버퍼들이 최초 정의된 힙의 메모리 사이즈를 초과하여 문자열에 저장되었는가	
	외부에서 입력된 값이 LDAP 질의문의 내용을 변경되지 않게 검증하는 절차가 있는가	
	외부에서 시스템 설정이나 구성요소를 제어 할 수 없게 막아져 있는가	
	프로그램 내에서 라이브러리를 적재할 때 절대 경로를 사용하는가	
	버퍼에 인덱스나 포인터를 사용해서 데이터를 기록할 때 인덱스가 음수 값이거나 포인터 연산 결과가 버퍼 이전의 값이 나오는가	
	버퍼의 범위를 벗어나서 읽는 경우가 존재하는가	
	외부 인자에 대한 적절한 검사 없이 그 인자가 배열 참조의 인덱스로 사용하는 경우가 있는가	
	null 문자가 붙지 않은 문자열이 있는가	
	잘못된 부호 확장으로 지정된 변수보다 값이 큰 경우가 있는가	
	부호정수가 무부호 정수로 변환되는 경우가 있는가	
보안기능	SW가 모든 가능한 실행경로에 대해 접근제어를 검사하고 있는가	
	SW가 중요 보안관련 자원에 대하여 읽기, 수정 권한을 제한하고 있는가	
	취약한 암호화 알고리즘을 사용된 SW가 있는가	
	프로그램이 보안과 관련된 민감 데이터를 송수신 할 경우 암호화를 하고 있는가	
	프로그램 코드 내부에 하드코드된 비밀번호가 있는가	
	길이가 짧은 키가 존재하는가(128비트 미만)	
	예측 가능한 난수를 사용하는 경우가 있는가	
	비밀번호를 암호화되지 않는 텍스트 형태로 저장되는 경우가 있는가	
	코드에 하드코드된 암호화키가 있는가	
	주석문에 비밀번호가 기입되어 있는가	
	비밀번호를 솔트 없이 일방향으로 해쉬하여 저장하는가	
	SW가 코드 내부에 고정된 사용자 계정 이름을 포함하고 있는가	
	응용프로그램을 실행할 때 사용자의 권한만을 통해 수행되도록 설계되어 있는가	
	권한을 높여 수행이 완료된 직후에 원래 권한으로 바로 복귀되도록 설계되어 있는가	
	RSA 알고리즘 사용시 OAEP 패딩을 사용하였는가	
	코드에 고정된 salt 값을 사용하고 있는가	
	하나의 포트에 다수의 소켓이 연결되어 있는가	
시간 및 상태	공유자원을 여러 스레드가 접근하여 사용할 경우 한 번에 하나의 스레드만 접근 가능하도록 되어 있는가	
	제어문을 사용하지 않는 재귀함수가 있는가	
	심볼릭명을 사용하여 특정 대상을 지정하는 경우가 있는가	

에러 처리	오류메시지에 민감한 정보를 포함하고 있는가	
	오류 발생시 예외처리가 잘 되어 있는가	
	프로그램 수행중 결과 값에 대한 적절한 처리 또는 예외상황에 대한 조건에 대한 검사가 잘 되어 있는가	
코드 오류	null 이 될 수 있는 래퍼런스는 참조하기 전에 null 값인지 검사하는 절차가 있는가	
	사용이 완료된 자원을 제대로 반환하고 있는가	
	부호 정수를 무부호 정수로 변환시 타입체크가 잘 이루어 지는가	
	정수를 문자로 변환시 타입체크가 잘 이루어 지는가	
	스택 변수 주소의 리턴 스택상의 지역변수의 주소 반환이 안되는 곳이 있는가	
	매크로 사용 시 공유 자원에서 동작하는 특정한 함수 패밀리가 함께 사용되고 있는가	
	스택 버퍼가 해제되어 있는가	
	부모 스레드가 자식 스레드 보다 먼저 종료되는 경우가 있는가	
	사용자가 사용할 수 있는 자원의 양을 제한하였는가	
캡슐화	디버그 코드가 남아있는가	
	시스템 데이터 정보가 노출되는 부분이 있는가	
API 오용	보안결정에서 도메인명을 이용한 DNS lookup 을 사용하는 경우가 있는가	
	위험하다고 알려진 라이브러리 함수가 사용된 곳이 있는가	
	작업 디렉터리 변경 없이 chroot Jail 생성한 곳이 있는가	
	버퍼 오버플로우가 발생할 수 있는 문자열 연산 함수를 호출한 곳이 있는가	
	다중 스레드 프로그램에서 getlogin()을 사용한 곳이 있는가	

[첨부 3] 모바일 시큐어코딩 체크리스트

유형	체크사항	체크 유무
입력 데이터 검증 및 표현	외부 입력값에 대하여 디렉터리 경로 조작 문자열 검증절차가 있는가	
	외부 입력값에 대하여 절대경로 문자열 포함여부 검증절차가 있는가	
API 악용	Object.equals(), Comparable.compareTo()과 Comparator.compare() 구현에서 매개변수를 Null 과 비교하여 지정된 값을 반환하고 있는가	
	한 클래스 내에서 equals()와 hashCode() 중 한쪽만 구현되어 있는가	
보안특성	민간함 정보를 통신 채널 통해 내보낼 때 암호화 과정을 거치는가	
	취약한 암호화 알고리즘을 사용된 SW 가 있는가	
	예측 가능한 난수를 사용하는 경우가 있는가	
	파일에 대한 접근권한이 최소한으로 유지되고 있는가	
	컴포넌트에 대한 접근권한을 외부에 제공하고 있는가	
	설정되어 있는 공유 아이디가 있는가	
시간 및 상태	공유자원을 여러 스레드가 접근하여 사용할 경우 한 번에 하나의 스레드만 접근 가능하도록 되어 있는가	
	제어문을 사용하지 않는 재귀함수가 있는가	
에러 처리	오류메시지에 민감한 정보를 포함하고 있는가	
	오류 발생시 예외처리가 잘 되어 있는가	
코드 품질	null 이 될 수 있는 래퍼런스는 참조하기 전에 null 값인지 검사하는 절차가 있는가	
캡슐화	private 로 선언된 배열을 public 으로 선언된 메소드를 통해 반환한 경우가 있는가	
	public 으로 선언된 데이터가 private 로 선언된 배열에 저장되는 경우가 있는가	
	디버깅을 위해 작성한 시스템 정보 출력 코드를 전부 삭제 하였는가	

[첨부 4] 파이썬 점검도구 점검 유형

① Pyt(파이썬 테인)

번호	탐지 유형	사례 및 설명
1	명령어 삽입	입력 데이터를 통한 공격
2	SQL Injection 탐지	Hibernate, JDO, JPA, Spring JDBC, LDAP Injection 등
3	XSS	JSP, Servlet, JavaScript(Android) XSS 를 탐지
4	디렉토리 조작	외부 입력값에 디렉터리 경로 조작 문자열을 넣어 공격

② Bandit

번호	탐지 유형	사례 및 설명
1	일반적 보안 위험	각 파일 스캔 후 보안 점검

③ Pyntch

번호	탐지 유형	사례 및 설명
1	런타임오류 감지	중요한 정보가 노출될 수 있는 런타임 예외 상황을 검사

④ RATS

번호	탐지 유형	사례 및 설명
1	Time of Check Time of Use	검사시점과 사용시점 사이에 자원 상태가 변할 수 있는 취약점 검사
2	Buffer oberflows	buffer 들이 한계치를 넘어서 저장이 되었는지 검사

[첨부 5] 빅데이터, 인공지능 플랫폼 시큐어코딩 체크리스트

유형	체크사항	체크 유무
입력데이터 검증 및 표현	데이터베이스와 연동된 웹 어플리케이션에서 입력된 데이터에 대한 유효성 검증 절차가 있는가	
	동적코드를 실행할 수 있는 함수를 사용하고 있는가	
	외부 입력 내용이 자원의 식별자로 사용되는 경우 입력 내용에 검증 절차가 있는가	
	외부 입력 내용이 동적 웹 페이지 생성에 사용될 경우 입력 내용에 검증 절차가 있는가	
	사용자 입력값이 운영체제 명령어의 일부 또는 전부로 구성되어 실행될 때 검증절차가 있는가	
	위험한 형식의 파일이 업로드가 실행되지 않도록 업로드에 제한이 있는가	
	신뢰되지 않는 url 주소 접속을 제한하는 절차가 있는가	
	외부에서 전송된 xml 문서에 포함된 DTD(DocumentTypeDefinition)을 비활성화 하고 있는가	
	XQuery 또는 XPath 쿼리에 사용되는 외부 입력데이터에 대하여 검증절차가 있는가	
	LDAP 명령어를 수행할 때 외부 입력값에 대한 검증절차가 있는가	
	사용자로부터 받은 요청에 대해서 사용자가 의도한 대로 작성되고 전송된 것인지 확인하는 절차가 있는가	
	사용자의 입력 값을 다른 시스템의 서비스 호출에 사용하는 경우 입력 값을 검증하는 절차가 있는가	
	http 요청에 들어있는 변수가 http 응답헤더에 포함되어 사용자에게 다시 전달될 때 검증하는 절차가 있는가	
	중요한 정보가 서버에 저장되고 보안확인 절차도 서버에서 진행되는가	
	검증없이 외부로부터 입력된 값을 입출력 함수의 포맷 문자열로 사용하는 곳이 있는가	
보안기능	적절한 인증과정 없이 중요정보를 열람하는 경우가 있는가	
	SW가 모든 가능한 실행경로에 대해 접근제어를 검사하고 있는가	
	SW가 중요 보안관련 자원에 대하여 읽기, 수정 권한을 제한하고 있는가	
	취약한 암호화 알고리즘을 사용된 SW가 있는가	
	프로그램이 보안과 관련된 민감 데이터를 송수신 할 경우 암호화를 하고 있는가	
	프로그램 코드 내부에 하드코드된 비밀번호가 있는가	
	길이가 짧은 키가 존재하는가(128비트 미만)	
	예측 가능한 난수를 사용하는 경우가 있는가	
	취약한 패스워드를 허용하는가	
	사용자의 영속적인 쿠키에 민감 데이터가 저장되게 설정 되어 있는가	
	주석문에 시스템 중요 정보가 기입되어 있는가	
	비밀번호를 솔트 없이 일방향으로 해쉬하여 저장하는가	
	소스 코드 또는 실행파일 다운로드시 무결성 검사를 하는가	
	인증시도 횟수가 적정하게 제한되어 있는가	

시간 및 상태	공유자원을 여러 스레드가 접근하여 사용할 경우 한 번에 하나의 스레드만 접근 가능하도록 되어 있는가	
	제어문을 사용하지 않는 반복문, 재귀함수가 있는가	
에러처리	오류메시지에 민감한 정보를 포함하고 있는가	
	오류 발생시 예외처리가 잘 되어 있는가	
	프로그램 수행중 결과 값에 대한 적절한 처리 또는 예외상황에 대한 조건에 대한 검사가 잘 되어 있는가	
코드오류	null 이 될 수 있는 래퍼런스는 참조하기 전에 null 값인지 검사하는 절차가 있는가	
	사용이 완료된 자원을 제대로 반환하고 있는가	
	모든 변수가 사용 전에 초기값을 할당 받고 있는가	
캡슐화	다중 스레드 환경에서 정보를 저장하는 멤버변수가 포함되지 않도록 하였는가	
	디버그 코드가 남아있는가	
	private 로 선언된 배열이 public 으로 선언된 메스드 통해 반환되는 경우가 있는가	
	public 으로 선언된 데이터 또는 메소드 인자가 private 선언된 배열에 저장되는 경우가 있는가	
API 오용	보안결정에서 도메인명을 이용한 DNS lookup 을 사용하는 경우가 있는가	