**82. 소프트웨어 개발 보안**

1. 소프트웨어 개발 보안의 개요 – 보안 취약점 최소화 및 안전한 SW 개발을 위한 보안활동을 의미

**기밀성(Confidentiality), 무결성(Integrity), 가용성(Availability)을 유지하는 것이 목표**

소프트웨어 보안 취약점이 발생하는 경우 : 보안 요구사항 정의x, SW 설계 시 논리적 오류 포함, 기술 취약점을 갖고 있는 코딩 규칙 적용, 소프트웨어의 배치 적절x, 보안 취약점 발견 시 적절하게 대응하지 못한 경우

안전한 SW 개발 보안 관련 기관

**행정안전부**(총괄), 한국 인터넷 진흥원(KISA), 발주기관, 사업자, 감리법인

3. SW 개발 보안 활동 관련 법령

개인정보 보호법 : 개인정보의 처리 및 보호에 관한 사항 규정, 개인의 자유와 권리 보호

정보통신망 이용촉진 및 정보보호 등에 관한 법률 : 정보통신망 이용 촉진, 서비스 이용자의 개인정보 보호

신용 정보의 이용 및 보호에 관한 법률 : 개인 신용정보의 효율적 이용, 체계적인 관리를 통해 정보 오남용 방지

위치정보의 보호 및 이용 등에 관한 법률 : 개인 위치정보의 안전한 이용 환경 조성, 정보의 유출, 오남용 방지

표준 개인정보 보호 지침 : 개인정보의 처리에 관한 기준, 개인정보 침해의 유형 및 예방 조치 세부사항 규정

개인정보 영향평가에 관한 고시 : 개인정보 영향평가를 위한 평가기관의 지정, 영향평가의 절차 세부기준 규정

개인정보의 안전성 확보 조치 기준 : 개인정보를 처리함에 있어 개인정보가 분실, 도난, 유출, 위조, 변조, 훼손되지 않도록 안전성 확보에 필요한 기술적, 관리적, 물리적 안전조치에 관한 최소한의 기준 규정

개인정보 영향평가에 관한 고시 : 개인정보 영향평가를 위한 평가기관의 지정, 영향평가 절차 세부기준 규정

4. 소프트웨어 개발 보안 활동 관련 기타 규정

RFID 프라이버시 보호 가이드라인 RFID 시스템 이용자의 프라이버시 보호 및 안전한 이용환경 조성 가이드라인

위치정보의 보호 및 이용 등에 관한 법률 : 개인 위치정보의 유출 및 오남용 방지 법률

위치정보의 관리적, 기술적 보호조치 권고 해설서 : 위치정보 사업자가 및 위치기반 서비스 사업자가 준수해야 하는 관리적, 기술적 보호조치의 구체적인 기준

바이오정보 보호 가이드라인 : 개인 바이오 정보 보호 및 안정한 활용을 위한 원칙 및 조치사항

뉴미디어 서비스 개인정보 보호 가이드라인 : 뉴미디어 서비스 이용 및 제공 시 개인정보 침해 예방 준수사항

**83. Secure SDLC(소프트웨어 개발 생명 주기)**

1. Secure SDLC의 개요 – 보안상 안전한 SW를 개발하기 위해 SDLC에 보안강화를 위한 프로세스를 포함한 것

2. 요구사항 분석 단계에서의 보안 활동 – 보안 항목에 해당하는 요구사항을 식별하는 작업 수행

전산화되는 정보가 가지고 있는 보안 수준을 보안 요소 별로 등급을 구분하여 분류

조직의 정보보호 관련 보안 정책 참고 후 SW개발에 적용할 보안 정책 항목 출처, 요구 수준, 세부 내용 문서화

**보안 요소**

기밀성 : 시스템 내의 정보와 자원은 인가된 사용자에게만 접근 허용, 전송 중 노출되더라도 데이터 읽기 불가

무결성 : 시스템 내의 정보는 오직 인가된 사용자만 수정 가능

가용성 : 인가받은 사용자는 언제라도 사용 가능

인증 : 사용자가 합법적인 사용자인지를 확인하는 모든 행위, ex)패스워드, 카드, 지문 검사

부인 방지 : 데이터를 송 수신한 자가 송 수신 사실을 부인할 수 없도록 송 수신 증거를 제공

3**. 설계 단계에서의 보안 활동** – 식별된 보안 요구사항들을 SW 설계서에 반영 및 보안 설계서 작성

위협 식별 후 발생 시 영향 범위와 대응책 등 수립, 환경에 대한 보안통제 기준을 수립하여 설계에 반영

네트워크 : 외부의 사이버 공격으로부터 개발 환경을 보호하기 위해 네트워크를 분리하거나 방화벽 설치

서버 : 보안이 뛰어난 운영체제를 사용하고 보안 업데이트, 외부접속에 대한 접근통제 실시

물리적 보안 : 출입통제, 개발 공간 제한, 폐쇄회로 등의 감시설비를 설치

개발 프로그램 : 허가되지 않은 프로그램을 통제하고 지속적인 데이터 무결성 검사를 실시

4. 구현 단계에서의 보안 활동 – 표준 코딩 정의서 및 SW개발 보안 가이드 준수, 설계서의 보안 요구사항 구현

개발 과정 중 지속적인 단위 테스트를 통해 SW에 발생할 수 있는 보안 취약점 최소화, 소스코드 안정성 확보

5. 테스트 단계에서의 보안 활동 – 설계 단계에서 작성한 보안 설계서를 바탕으로 보안사항 반영, 동작 점검

정적 분석 도구, 동적 분석 도구 또는 모의 침투테스트를 통해 설계 단계에서 식별된 위협들의 해결여부 검증

구현 단계에서 추가로 제시된 위협들과 취약점들을 점검할 수 있도록 테스트 계획를 수립하고 시행

테스트 단계에서 수행한 모든 결과는 문서화하여 보존, 개발자에게 피드백

6. 유지보수 단계에서의 보안 활동 – 발생할 수 있는 보안 사고들을 식별, 발생 시 이를 해결 후 보안 패치 실시

**84. 세션 통제**

1. 세션 통제의 개요 – 세션은 서버와 클라이언트의 연결을 의미, 세션 통제는 세션 연결과 그로 인해 발생하는 정보를 관리하는 것, SW 개발 과정 중 요구사항 분석 및 설계 단계에서 진단해야 하는 보안 점검 내용

2. 불충분한 세션 관리 – **일정한 규칙이 있는 세션ID가 발급, 타임 아웃이 너무 길게 설정되어있는 경우**

침입자는 세션 하이재킹과 같은 공격을 통해 획득한 세션ID로 시스템 기능 이용 또는 중요 정보에 접근 가능

세션 하이재킹 : 서버에 접속하고 있는 Client의 세션 정보를 가로채는 공격 기법, 세션 가로채기라고도 함

3. 잘못된 세션에 의한 정보 노출 – 다중 스레드(Multi-Thread)환경에서 멤버 변수에 정보 저장 시 발생

싱글톤 패턴에서 발생하는 **레이스 컨디션으로** 인해 **동기화 오류가 발생하거나** **멤버 변수 정보 누출 가능**

멤버 변수보다 지역 변수를 활용해 변수의 범위를 제한함으로써 방지 가능

레이스 컨디션 : 두 개 이상의 프로세스가 공용 자원을 획득하기 위해 경쟁하고 있는 상태를 의미

4. 세션 설계 시 고려 사항 – 시스템의 모든 페이지에서 로그아웃이 가능하도록 UI(User Interface)를 구성

로그아웃 요청 시 할당 세션 완전 제거, 이전 세션 완전 종료 시 새 세션 생성, 중복 로그인 비허용 시 클라이언트의 중복 접근에 대한 세션 관리 정책을 수립, 패스워드 변경 시 활성화된 세션을 삭제하고 재할당

5. 세션ID의 관리 방법

안전한 서버에서 최소 128비트의 길이로 생성, 안전한 난수 알고리즘 적용, URL Rewrite 기능을 사용하지 않는 방향으로 설계, 로그인 시 이전 세션ID 삭제 후 재할당, 장기간 접속 시 주기적으로 재할당되도록 설계

**85. 입력 데이터 검증 및 표현**

1. 입력 데이터 검증 및 표현의 개요 – 입력 데이터로 발생하는 문제 예방을 위한 구현 단계 검증 보안 항목

SW개발의 구현 단계에서 다음과 같은 유효성 검증 체계 필요

검증되지 않은 데이터 입력 처리 구현, 입력 데이터 처리 객체에 지정된 자료형 확인, 일관된 언어셋 사용 코딩

2. 입력 데이터 검증 및 표현의 보안 약점

sql 삽입 : 입력란에 sql을 삽입해 무단으로 DB조회 및 조작, 예약어 및 특문 필터링 설정으로 방지 가능

경로조작 및 자원 삽입 : 데이터 입출력 경로를 조작하여 서버 자원을 수정/삭제할 수 있는 보안 약점

사용자 입력값을 식별자로 사용하는 경우 경로 순회 공격을 막는 필터를 사용하여 방지 가능

크로스사이트 스크립팅(XSS) : 웹에 악의적인 스크립트를 삽입하여 방문자들의 정보 탈취, 비정상적인 기능 수행을 유발하는 보안 약점, 웹 인터페이스를 통한 시스템 명령어 전달X, 외부 입력값을 내부 명령어로 사용X ↓동일

운영체제 명령어 삽입 : 외부 입력값을 통해 시스템 명령어의 실행 유도, 권한 탈취 및 시스템 장애 유발

위험한 형식 파일 업로드 : 악의적 명령어가 포함된 스크립트 파일 업로드로 시스템 손상 및 제어 유발

업로드 되는 파일의 확장자 제한, 파일명의 암호화, 웹사이트와 파일 서버의 경로 분리, 실행 속성 제거

신뢰되지 않는 URL 주소로 자동접속 연결 : 입력 값으로 사이트 주소를 받는 경우 이를 조작하여 방문자를 피싱 사이트로 유도, 연결되는 외부 사이트의 주소를 화이트 리스트로 관리해 방지할 수 있음

86. 보안 기능

1. 보안 기능의 개요 – 인증, 접근제어, 기밀성, 암호화 등을 올바르게 구현하기 위한 보안 점검 항목

각 보안 기능들은 서비스 환경이나 취급 데이터에 맞게 처리될 수 있도록 구현, 사용자의 중요도 구분, 차별화된 인증 방안 적용, 인증된 사용자가 이용할 기능과 데이터에 대해 개별적으로 접근 권한을 부여하여 인가되지 않은 기능과 데이터로의 접근 차단, 중요 정보의 변조/삭제/오남용 등을 방지하기 위해 안전한 암호화 기술 적용

2. 보안 기능의 보안 약점

절절한 인증 없이 중요기능 허용 : 보안검사를 우회해 인증과정 없이 중요 정보 또는 기능에 접근 및 변경

중요정보나 기능을 수행하는 페이지에서는 재인증 기능을 수행하도록 하여 방지

부적절한 인가 : 접근제어 기능이 없는 실행경로를 통해 정보 또는 권한 탈취 가능

모든 실행경로에 대해 접근제어 검사 수행, 사용자에게는 반드시 필요한 접근 권한만을 부여하여 방지

중요한 자원에 대한 잘못된 권한 설정 : 권한 설정이 잘못된 자원에 접근하여 해당 자원을 임의로 사용

SW관리자만 자원들을 읽고 쓸 수 있도록 설정, 인가되지 않은 사용자의 중요 자원에 대한 접근 여부 검사

취약한 암호화 알고리즘 사용 : 암호화된 환경설정 파일을 해독하여 비밀번호 등의 중요정보 탈취

안전한 암호화 알고리즘 이용, 업무관련 내용 및 개인정보는 IT보안인증사무국의 안정성 확인 암호 모듈을 이용

중요정보 평문 저장 및 전송 : 암호화되지 않은 평문 데이터를 탈취하여 중요한 정보 획득

중요한 정보를 저장하거나 전송할 때는 반드시 암호화 과정을 거침, HTTPS 또는 SSL 등의 보안 채널 이용

하드코드된 비밀번호 : 소스코드 유출 시 내부에 하드코드된 패스워드를 이용해 관리자 권한 탈취

패스워드는 암호화하여 별도의 파일에 저장, 디폴트 패스워드나 디폴트 키의 사용을 피함

하드 코드 : 데이터를 코드 내부에 직접 입력하여 프로그래밍하는 방식

87. 시간 및 상태

1. 시간 및 상태의 개요 – 동시 수행을 지원하는 병렬 처리 시스템이나 다수의 프로세스가 동작하는 환경에서 시간과 실행 상태를 관리하여 시스템이 원활하게 동작되도록 하기 위한 보안 검증 항목

2. TOCTOU 경쟁 조건

검사 시점(Time Of Check)과 사용 시점(Time Of Use)을 고려하지 않고 코딩하는 경우 발생하는 보안 약점

검사 시점에는 사용이 가능했던 자원이 사용 시점에는 사용할 수 없게 된 경우에 발생한다.

프로세스가 가진 자원 정보와 실제 자원 상태가 일치하지 않는 동기화 오류/교착 상태 등 발생, 코드 내 동기화 구문을 사용하여 해당 자원에는 한 번에 하나의 프로세스만 접근 가능하도록 구성, 성능 감소 동반(동기화 구문)

3. 종료되지 않는 반복문 또는 재귀함수 – 종료 조건 정의X, 논리 구조상 종료될 수 없는 경우

시스템 자원이 끊임없이 사용되어 서비스나 시스템이 자원고갈로 정지하거나 종료됨

수행 횟수를 제한하는 설정 추가, 종료 조건을 점검하여 반복 또는 호출의 종료 여부를 확인함으로써 방지 가능

88. 에러 처리

1. 에러처리의 개요 – 소프트웨어 실행 중 발생할 수 있는 오류들을 사전에 정의하여 오류로 인해 발생할 수 있는 문제들을 예방하기 위한 보안 점검 항목들

각 프로그래밍 언어의 예외처리 구문을 통해 오류에 대한 사항을 정의, 예외처리 구문으로 처리하지 못한 오류들은 중요정보를 노출시키거나, 소프트웨어의 실행이 중단되는 등 예기치 못한 문제 유발

2. 오류 메시지를 통한 정보노출 – 오류 발생으로 사용자/디버깅 등의 중요 정보를 SW가 메시지로 외부에 노출

예외처리 구문에 예외의 이름이나 스택 트레이스를 출력하도록 코딩한 경우 해커는 SW의 내부구조 쉽게 파악

오류 발생 시 가능한 한 내부에서만 처리되도록 하거나 메시지를 출력할 경우 최소한의 정보 또는 사전에 준비된 메시지만 출력되도록 함으로써 방지 가능

3. 오류 상황 대응 부재 – SW 개발 중 예외처리를 하지 않았거나 미비로 인해 발생하는 보안 약점

예외처리를 하지 않은 오류들로 인해 SW의 실행이 중단되거나 의도를 벗어난 동작 유도 가능

오류가 발생할 수 있는 부분에 예외처리 구문을 작성, 제어문을 활용하여 오류가 악용되지 않도록 코딩

4. 부적절한 예외처리

함수의 반환 값 또는 오류들을 세분화하여 처리하지 않고 광범위하게 묶어 한 번에 처리하거나, 누락된 예외가 존재할 때 발생하는 보안 약점

모든 오류들을 광범위한 예외처리 구문으로 정의 시 예기치 않은 문제 발생 가능

모든 함수의 반환 값이 의도대로 출력되는지 확인 후 세분화된 예외처리를 수행함으로써 방지

89. 코드 오류

1. 코드 오류의 개요 – SW 구현 단계에서 개발자들이 코딩 중 실수하기 쉬운 형 변환, 자원 반환 등의 오류를 예방하기 위한 보안 점검 항목들

2. 널 포인터(Null Pointer) 역참조 – 널 포인터가 가리키는 메모리에 어떠한 값을 저장할 때 발생

오류가 발생할 경우 라이브러리 함수들이 널 값을 반환할 때 이 값을 포인터로 참조하는 경우 발생

대부분의 OS에서 널 포인터는 메모리의 첫 주소를 가리킴, 해당 주소를 참조할 경우 SW의 비정상 종료 가능

널이 될 수 있는 포인터를 이용하기 전에 널 값을 갖고 있는지 검사함으로써 방지 가능

3. 부적절한 자원 해제 – 자원을 반환하는 코드를 누락, 프로그램 오류로 할당된 자원을 반환X 발생

힙 메모리, 소켓 등의 유한한 시스템 자원이 계속 점유하고 있으면 자원 부족으로 인해 신규 입력 처리 불가

자원 반환 코드 누락 확인 및 오류로 인해 함수가 중간에 종료 시 예외처리에 관계없이 자원을 반환하도록 코딩

5. 초기화되지 않은 변수 사용 – 변수 선언 후 값이 부여되지 않은 변수를 사용할 때 발생하는 보안 약점

변수가 선언되어 메모리가 할당되면 해당 메모리에 이전 사용 내용이 남아 변수가 외부에 노출되는 경우 중요정보 악용, 변수 선언 시 할당된 메모리를 초기화함으로써 방지할 수 있다.

90. 캡슐화

1. 캡슐화의 개요 – 정보 은닉이 필요한 중요 데이터와 기능을 불충분하게 캡슐화 하거나 잘못 사용함으로써 발생할 수 있는 문제를 예방하기 위한 보안 점검 항목

2. 잘못된 세션에 의한 정보 노출