**SQL 인젝션 공격**

**1. SQL 인젝션 공격 개요**

- SQL 인젝션 취약점은 웹 서버 영역의 데이터베이스로 전송되는 SQL 쿼리문을 사용자가 임의로 조작할 수 있는 경우에 발생

- 공격자는 취약점을 이용하여 데이터베이스에 저장되어 있는 다른 사용자의 개인정보 등 허가되지 않은 정보에 접근하거나 데이터를 변조

- 두 가지 사례 (회원 ID를 입력하여 회원정보를 조회하는 웹 페이지)

- SQL 쿼리문을 구성하는 웹 애플리케이션의 소스코드를 다음과 같다고 가정

$id = $\_REQUEST[ ‘id’ ];

$query = “SELECT name, email FROM users WHERE id = ‘$id’;”;

- 위 경우에서 SQL 인젝션 취약점 발생

**1) WHERE 구문 우회**

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

- 사용자는 ID가 1인 사용자 정보 요청

- 요청을 받은 웹 애플리케이션은 내부의 데이터베이스로 다음과 같은 SQL 쿼리문 전송

SELECT name, email FROM users WHERE ID=’1’

- 쿼리문에는 WHERE 조건문에서 사용자가 입력한 ID 값 1이 조건으로 입력

- 이 쿼리문은 users 테이블에서 ID가 1인 사용자의 이름과 이메일을 가져오는 쿼리문

- 쿼리문을 받은 데이터베이스는 쿼리문을 실행하여 ID가 1인 사용자 정보 반환

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

- 공격자는 원래의 ID 값 1 대신 SQL 쿼리문 조작을 위해 다음과 같이 입력

1’ or ‘1’=’1

- 이 값을 SQL 쿼리문에 그대로 삽입하면 다음과 같이 구성

SELECT name, email FROM users WHERE id =’1’ or ‘1’=’1’;

- WHERE 이하를 보면 “or” 키워드와 함께 삽입되어 항상 참이 되는 조건으로 변경

- ID 값이 1인 사용자의 데이터뿐만 아니라 모든 사용자의 이름과 이메일 전달

- 공격자가 입력할 수 있는 입력 값에 제한이 없고, 그 입력값이 SQL 쿼리문에서 그대로 실행된다면 SQL 인젝션 성공

**2) UNION 공격**

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

- 공격자는 “or” 키워드를 이용한 구문 대신에 UNION 키워드를 삽입 후 사용자 이름과 패스워드를 요청하는 SELECT 구문을 삽입

- 다음과 같은 SQL 쿼리문이 실행

SELECT name, email FROM users WHERE ID=’1’ UNION SELECT name, pw FROM users#’

- UNION을 사이에 두고 SELECT 구문 두 개가 위치

- UNION으로 합집합으로 두 개의 SELECT 구문의 결과를 모두 포함

- ID가 1인 사용자 정보 외, users 테이블의 모든 사용자의 이름과 비밀번호 같이 반환

- 쿼리문 끝에 “#”은 MySQL DB에서 “#” 뒤에 오는 내용 주석 처리 위함

- “#”이 없다면 짝이 없는 가장 끝의 문자 “ ‘ “로 인해 에러 발생 가능성