1. 기사 내용

제목	웹 기반 해킹 '급증', 웹 서버 보안 '비상'
기사 날짜	2014년 4월 8일
스크랩 담당자	손우규
요약	• 웹 기반 해킹, 47.2%로 절반 이상 차지 웹 기반 해킹 공격이 갈수록 증가하고 있어 웹 서버 보안에 비상등이 켜졌다. 최근 잇따른 개인정보 유출사고를 살펴보면 공통점이 웹사이트 취약점을 악용한 해킹이다.
출처	• http://www.boannews.com/media/view.asp?idx=40482

2. 용어

용어	설명
해킹	컴퓨터 네트워크의 취약한 보안망에 불법적으로 접근하거나 정보 시스템에 유해한 영향을 끼치는 행위.
보안	안전을 유지함

3. 관련 기술(SQL 인젝션)

3.1 개요

SQL 인젝션 (SQL 삽입, SQL 주입으로도 불린다) 은 코드 인젝션의 한 기법으로 클라이언트의 입력값을 조작하여 서버의 데이터베이스를 공격할 수 있는 공격방식을 말한다. 주로 사용자가 입력한 데이터를 제대로 필터링, 이스케이핑하지 못했을 경우에 발생한다. 공격이 쉬운데 비해 파괴력이 어마어마 하기 때문에 시큐어코딩을 하는 개발자라면 가장 먼저 배우게 되는 내용이다. 이러한 injection 계열의 취약점들은 테스트를 통해 발견하기는 힘들지만 스캐닝툴이나 코드 검증절차를 거치면 보통 쉽게 발견되기 때문에 탐지하기는 쉬운 편이다. 보안회사 Imperva가 2012년에 발표한 보고서에 따르면 월 평균 4회 가량의 SQL 인젝션 공격이 일어난다고 한다. OWASP에서도 수년동안 인젝션 기법이 보안 위협 1순위로 분류되는 만큼 보안에 각별한 주의가 필요하다.

3.2 공격 형태

3.2.1 형태 1

로그인 폼에 아이디와 비밀번호를 입력하면 입력한 값이 서버로 넘어가고, 데이터베이스를 조회하여 정보가 있다면 로그인에 성공하게 된다. 이때 데이터베이스에 값을 조회하기 위해 사용되는 언어를 SQL이라고 하며 다음과 같이 생겼다고 가정하자.

```
SELECT user FROM user_table WHERE id='입력한 아이디' AND password='입력한 비밀번호';
```

여기서는 패스워드를 평문으로 비교하는데 사실 이는 예제를 간단히 하기 위한거고 실제로 이런 식으로 짜면 개인정보보호법 29조 위반[2]으로 과태료 폭탄을 맞을 수 있다. 패스워드는 반드시 PBKDF2 이상의 보안성을 가지는 해시 함수로 해싱해야 한다.

일반적인 유저라면 이렇게 로그인 할 것이다.

아이디:

무냐

비밀번호:

나무

```
SELECT user FROM user_table WHERE id='무냐' AND password='나무';
```

이때 악의적인 유저가 다음과 같이 입력했다.

아이디:

세피로트

비밀번호:

```
' OR '1' = '1
```

```
SELECT user FROM user_table WHERE id='세피로트' AND password=' ' OR '1' = '1';
```

비밀번호 입력값과 마지막 구문을 자세히 살펴보자. 따옴표를 올바르게 닫으며 password=' '를 만들어 버림과 동시에 SQL 구문 뒤에 OR '1' = '1'을 붙였다. WHERE 뒤에 있는 구문을 간단히 축약하면 false AND false OR true로 정리할 수 있는데, 논리학에 따르면 AND 연산은 OR보다 연산 우선순위가 빠르기 때문에 해당 구문 전체는 true가 되어 올바른 값으로 판단하고 실행하게 된다. 즉 아이디와 비번을 제대로 매치하지 못했어도 ' OR '1' = '1 구문 때문에 로그인에 성공하게 되는 것이다.

로그인 뿐만 아니라 JOIN이나 UNION같은 구문을 통해 원하는 코드를 실행하게 할 수도 있다.

sql injection이 되는지는 ' 나 " 를 로그인창에 넣어서 sql 에러를 뿜어내는지 확인하면 된다. 흔한 에러는

Warning: mysql_fetch_array():

Warning: mysql_fetch_assoc():

Warning: mysql_numrows():

Warning: mysql_num_rows():

Warning: mysql_result():

Warning: mysql_preg_match():

3.2.2 형태 2

로그인 폼도 결국엔 서버에 요청을 해서 받는 것이다. http헤더를 보면 응답 헤더에 서버의 종류와 버전이 나온다. Apache 서버는 MySQL 서버, IIS는 MS SQL같은 방식으로 데이터베이스의 종류를 추측할 수 있다. DB엔진을 알아내서 해당 시스템에 맞는 명령어를 이용해 데이터를 뽑아내거나 할수 있다.

3.3 공격 원인

현재 대부분의 기업은 방화벽, IDS, IPS 등 다양한 보안 장비를 이용해 보안을 강화하고 있다. 이러한 환경에서 더 이상 예전의 방법을 이용한 해킹은 성공하기가 매우 어렵다. 또한 웹 서버가 예전의 환경과 달리 대부분 DB서버와 연동이 되게 되어 있고 이러한 DB서버를 해커가 실제 공격의 목표로 하고 있기 때문에 웹 해킹은 더욱 늘어난 것으로 보고 있다. 원인을 좀 더 세부적으로 살펴보면 다음과 같다.

- 웹 프로그래밍 지식의 대중화
- 많은 로그로 인해 실질적인 추적이 쉽지 않음
- 전통적 애플리케이션의 웹 통합 가속화로 인한 풍부한 공격 대상
- 웹 서버 = 내부 서버로 가는 관문
- F/W로 방어 불가
- 개발 당시 보안 고려 제외
- 개발자의 Copy & Paste
- 개발 툴 과잉 의존
- 촉박한 개발 기간으로 인한 형식적인 테스트 단계
- F/W, IDS 등 보안 솔루션에 과잉 의존 이러한 여러 원인 때문에 웹 해킹은 현재 성공한 해킹의 70% 이상을 차지하고 있는 것이 현실

3.4 보완 방법(해결방안)

3.4.1 준비된 선언

준비된 선언(Prepared statement)을 사용하면 사용자의 입력이 쿼리문(SQL 선언 템플릿)으로부터 분리되기 때문에 SQL 삽입을 효과적으로 방어할 수 있다.

준비된 선언은 다음과 같은 단계에 따라 실행된다.

준비: 쿼리문이 데이터베이스 관리 시스템(DBMS)으로 전송된다.
이 때 사용자 입력 값은 전송되지 않는 대신 플레이스홀더로 표시된다.
DBMS가 쿼리문을 분석하고 최적화를 진행한다.
실행: 플레이스홀더에 입력할 사용자 입력 값이 전송되고 DBMS가 쿼리문을 실행한다.
PHP에서는 PDO 또는 MySQLi 중 하나로 준비된 선언을 사용할 수 있다.

```
$username = $_POST["username"];
$password = $_POST["password"];

$pdo = new PDO('mysql:dbname=testdb;host=127.0.0.1', 'user', 'password');
$pdo->setAttribute(PDO::ATTR_EMULATE_PREPARES, false);

$st = $pdo->prepare('SELECT * FROM users WHERE username=:username AND password=:password'); // 준비
$st->bindParam(':username', $username); // 사용자 입력 값 할당
$st->bindParam(':password', $password); // 사용자 입력 값 할당
$st->execute(); // 실행

var_dump($st->fetchObject());
```

3.4.2 이스케이프

SQL에서 특별한 의미를 갖는 문자들을 이스케이프해서 SQL 삽입을 방어할 수도 있다. 예를들어 PHP에서는 이스케이프를 위해 mysqli_real_escape_string(); 함수를 사용할 수 있다.

```
$username = $mysqli->real_escape_string($_POST["username"]);
$password = $mysqli->real_escape_string($_POST["password"]);

$mysqli->query("SELECT * FROM users WHERE username='{$username}' AND password='{$password}'");
```

4. 참고 자료

- https://ko.wikipedia.org/wiki/SQL_%EC%82%BD%EC%9E%85