
웹 기반 소프트웨어 보안 취약성 분석

소프트웨어무결점센터 Workshop

2009. 7. 10

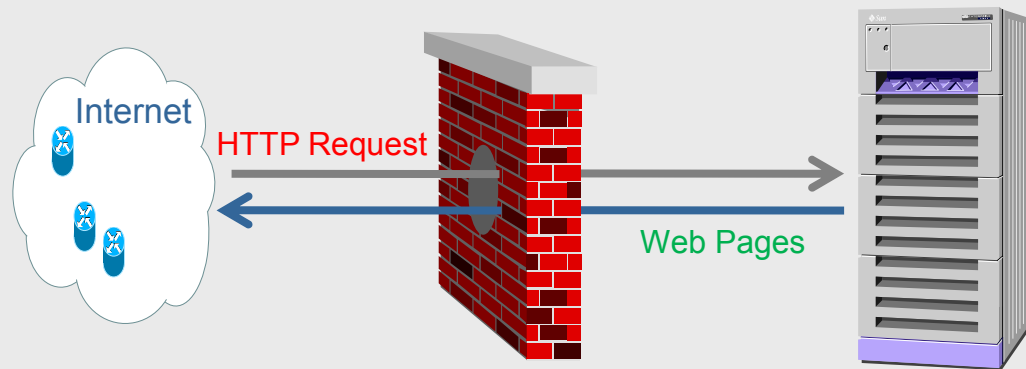
한국항공대학교, 안 준 선

Outline

- 소개
- 관련 연구
- 기존 연구
- 연구 내용
- 연구 계획
- 결론

소개

- 웹 응용 프로그램(Web Application Programs)
 - PHP, JSP, Servlet, ...
 - 웹을 통한 외부의 접근에 의하여 웹페이지를 생성
 - 문자열 기반 작업
 - 외부의 불특정 입력을 받아 수행되므로 위험에 노출됨



소개

- 웹 응용프로그램 취약성 주요 원인
 - 적절히 검증되지 못한 외부의 입력이 응용 프로그램 내에서 민감한 작업에 사용될 경우 취약성 발생

Ex) OWASP Top 10

1. 크로스 사이트 스크립팅	2. 삽입 취약점
3. 악성파일 실행	4. 불안정한 직접 객체 참조
5. 크로스 사이트 요청 변조	6. 정보유출 및 부적절한 에러처리
7. 훼손된 인증 및 세션 관리	8. 불안정한 암호화 저장
9. 불안정한 통신	10. URL 접근 제한 실패

관련연구 : SQL Injection

- SQL Injection
 - CWE-89 : Failure to Preserve SQL Query Structure
 - A vulnerable web application code segment

```
$id = $_Request('name');  
mysql_query("SELECT MessageID, Subject FROM".  
            "messages WHERE MessageID='$id'");
```

- Input value for attacking

```
` OR 1=1;
```

- Resulting parameter for `mysql_query`

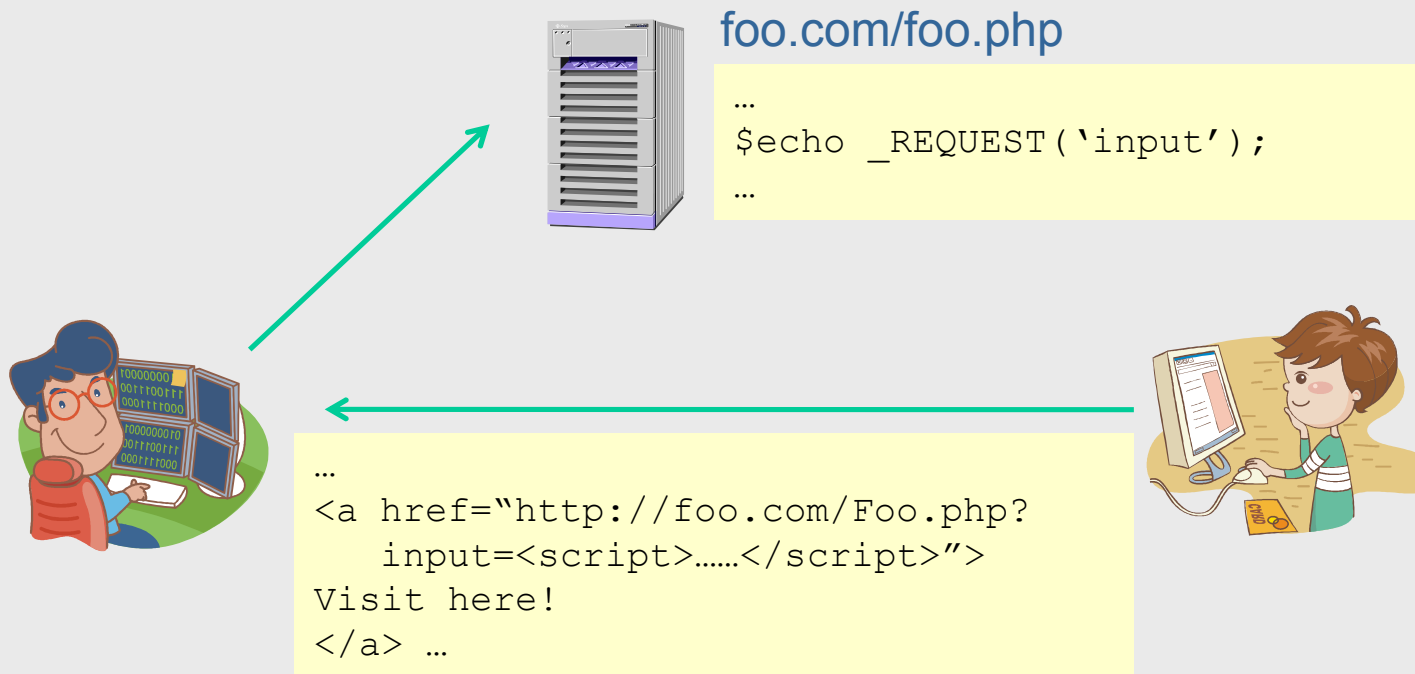
```
SELECT MessageID, Subject FROM messages WHERE  
MessageID = ''; DROP TABLE('messages');
```

관련연구 : XSS Scripting

- 크로스 사이트 스크립트(XSS)
 - CWE-79 : Failure to Preserve Web Page Structure
 - 공격자가 희생자의 브라우저에서 특정 스크립트를 수행함으로써 공격
 - 종류
 - Type 1 : Reflected XSS
 - Type 2 : Stored XSS
 - Type 3 : Dom XSS

관련연구 : XSS Scripting

- Reflected XSS (Non-Persistent)
 - 외부 입력을 client를 위한 웹페이지 생성에 직접 사용



관련연구 : XSS Scripting

■ Stored XSS (Persistent)

The image shows two screenshots of a web application interface. The top screenshot is a form titled "방명록" (Guestbook) with a "새글" (New Post) button. The form fields include: "비밀번호" (Password) with 10 dots, "이름" (Name) with "good", "이메일" (Email), "홈페이지" (Homepage), "옵션" (Options) with checkboxes for "HTML사용" (checked), "답변메일받기" (unchecked), and "비밀글" (unchecked), and "제목" (Title) with "ㅎㅎㅎ". The "제목" field contains the malicious payload: `www.daum.net`. The bottom screenshot shows the "방명록" page after submission. It features a "로그인" (Login) and "회원가입" (Sign Up) link. A table displays the submitted entry: "이름" (Name) is "good" and "제목" (Title) is "ㅎㅎㅎ". To the right of the entry, it shows "(2007-04-23 03:46:45 , Hit : 0)". Below the table, the text "www.daum.net" is visible, which is the result of the stored XSS attack.

관련연구 : XSS Scripting

- Dom based XSS
 - Local cross site scripting : 클라이언트 스크립트를 사용한 외부 입력값 전달
 - 서버 프로그램의 입력값 필터링이 우회됨

```
...
$argsPhp = sanitize_string($_Request("input"));
$echo "
<script>
  var idx = document.URL.indexOf('=');
  var arg1 = document.URL.substring(idx+1,document.URL.length);
  document.write("value1 : "+arg1);
  document.write("value2 : "
  ".$argsPhp.
  "); <script>"
...
```

관련연구 : XSS Scripting

■ Dom based XSS (계속)

```
<script>
var idx = document.URL.indexOf('=');
var arg1 = document.URL.substring(idx+1,
    document.URL.length);
document.write("value1 :"+arg1);
document.write("value2 : satnized ");
</script>
```



```
...
<a href="http://foo.com/foo.php?
  input=<script>.....</script>">
Visit here!
</a> ...
```



foo.com/foo.php



소개

- 연구 목표
 - 웹 응용 프로그램 내에 존재하는 크로스 사이트 스크립팅(XSS) 및 SQL 삽입(SQL Injection) 취약점 검출
 - 기본 분석 방법
 - 안전한 자료 흐름 (Secure Information Flow) 분석
: 외부의 입력이 어디에서 사용되는가 ?
 - 문자열 분석 (String Analysis)
: 어떤 형태의 문자열이 민감한 작업에 사용되는가?

기존연구

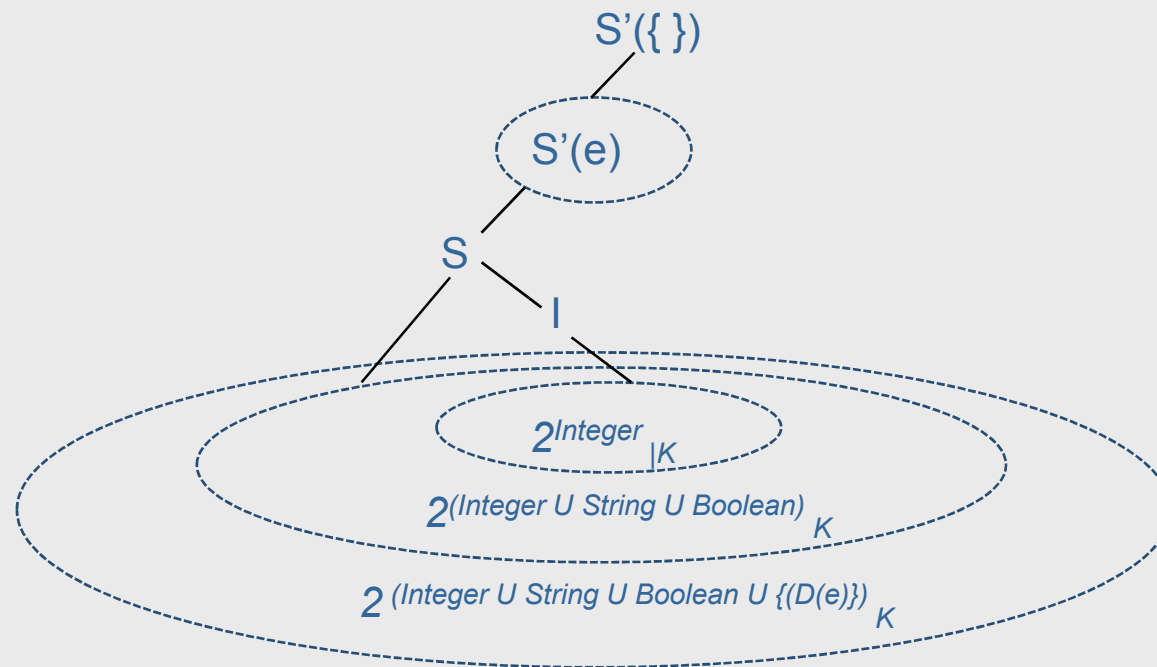
- 문자열 분석(String Analysis)
 - Christensen et al, SAS '03
 - Java 문자열 데이터 분석 (정규 표현식 기반)
 - 생성되는 SQL문의 문법 검사
 - Minamide, WWW '05
 - PHP 문자열 데이터 분석 (CFG 분석)
 - 스트링에 대한 함수를 Finite State Transducer로 표현
 - XSS 분석
 - Doh et al, SAS '09
 - Abstract Parsing
 - 주어진 문법에 기반하여 가능한 문자열 값을 Parsing Stack 상태의 집합으로 표현 ← Pros and Cons

기존연구

- 안전한 자료흐름(Secure Info. Flow) 분석
 - Huang et al, WWW '04
 - 외부의 입력이 sanitize 되지 않고 exec()과 echo() 함수에 사용되는지 검사
 - Type 기반 분석
 - satitize함수는 직접 명시하여야 함
 - Su and Wasserman, PLDI '07
 - 문자열 분석과 안전한 자료흐름 분석을 함께 수행
 - SQL Injection 취약성 분석
 - Su and Wasserman, ICSE '08
 - 문자열 분석과 안전한 자료흐름 분석을 함께 수행
 - Type 1 and Type 2 XSS 취약성 분석
- : $L(\langle \text{rexp of attack pattern} \rangle) \cap L(V, \Sigma, X, R') = \emptyset$

연구내용

- 가벼운 XSS, SQL Injection 취약성 분석기
 - 외부로부터 위험한 토큰(' , <script>)이 민감한 함수(echo, mysql_query 등)에 전달되는지 분석.
 - 대치 연산을 제외 문자 집합을 사용하여 표현



연구내용

- 실행화면 (Eve Activity Tracker, 905 lines)

```
PHP Analyzer : test2.php
File Analyze Edit Style
Verify Security
$m Memory Status Table;
$user_id = trim($user_id);
$connect=dbconn();
$check=mysql_fetch_array(mysql_query("select count(*) from $member_table where user_id='$user_id'"));
mysql_close($connect);
head();
?>
<table border=0 width=100% height=100%>
<tr>
<td align=center>
<?
if($check[0]) echo "$user_id 는 이미 등록된<br> 아이디입니다.";
else echo "$user_id 는 사용하지할수 없습니다.";
?>
</td>
</tr>
</table>
<form>
<tr>
<td align=center><input type=button value='Close window' onclick=window.close(); class=submit></td>
</tr>
</form>
</table>
<?
    @mysql_close($connect);
    foot();
?>
```

Line #5 : 변수 \$user_id 이(가) 위험한 값을 가질 수 있습니다. SQL Injection 공격의 위험이 있습니다.
Line #13 : 변수 \$user_id 이(가) 위험한 스크립트를 가질 수 있습니다. XSS 공격의 위험이 있습니다.
Line #14 : 변수 \$user_id 이(가) 위험한 스크립트를 가질 수 있습니다. XSS 공격의 위험이 있습니다.
4 : [\$user_id.InitOf\$user_id, \$member_table:UserTable, \$connect.TopString,]
12 : [\$user_id.InitOf\$user_id, \$check[0].InitOf\$check[0], \$member_table:UserTable, \$check.MYSQL_FETCH_RESULTI, \$connect.TopString,]

caret: text position : 293(12), view location : [19, 203]

연구내용

- 문제점, 개선 방향
 - 안전성
 - `<scr<script>ipt>document.write(document.cookie)</sc</script>ipt>`
 - ...
 - » 관대한 브라우저 : `<scr%00ipt> <script >`
 - » URL 인코딩, 더블 인코딩 :
`%3cscript%3e, %253cscript%253e`
 - Regular Expression 인자의 처리
 - `preg_replace('<[Ss][Cc][Rr][Ii][Pp][Tt][^>]*>', 'Cleaned', $data)`

연구 계획

- 삽입 취약성 분석 (SQL Injection, XSS 1, 2)
 - 대형 프로그램 분석
 - E107 (132,850 lines)
 - String 분석보다 빠른 속도 입증
 - 주요 과제 : 라이브러리 함수 구현, 파서 구현
 - 안전성/정밀성 확보
 - 요약 도메인 재설계 : Regular expression 표현
 - 안전성 검증

연구 계획

- Type 3 XSS 삽입 취약성 분석기 구현
 - 웹 프로그램으로부터 생성되는 스크립트 언어에 대한 안전한 자료 흐름 분석 필요
 - 2단계 분석
 - 1차 분석
 - Type1, Type2 XSS 분석
 - 외부의 입력이 생성되는 html 파일 내에서 리터럴로만 사용되는지 분석
 - 2차 분석
 - Abstract Parsing : 외부 입력을 불확실한 literal 값으로 간주하여 구문분석 수행
 - 불완전한 구문트리 기반으로 클라이언트 스크립트 내에서의 안전한 자료 흐름 분석 수행

결론

- 웹 기반 소프트웨어 보안 취약성 분석
 - 문자열 분석 + 안전한 자료 흐름 분석 필요
 - 주요 연구 목표
 - 삽입 취약성 분석 : 가벼운 분석 구현
 - Su07 : e107 SQL 삽입 취약성 분석 - 39시간 소요
 - SQL Injection, XSS (Type 1, 2)
 - XSS 취약성 분석 (Type 3)
 - Secure information flow analysis of client-side scripts