http://dx.doi.org/10.15207/JKCS.2015.6.4.225

창의·인성 교육기반의 디지털 융합 큐레이션 시스템에 관한 취약점 분석

신승수[1*], 김정인[2], 윤정진[3]
[1]동명대학교 정보보호학과, [2]동명대학교 컴퓨터공학과, [3]동명대학교 유아교육과

Vulnerability Analysis of the Creativity and Personality Education based on Digital Convergence Curation System

Seung-Soo Shin[1*], Jung-In Kim[2], Jeong-Jin Youn[3]
[1]Dept. of Information Security, Tongmyong University
[2]Dept. of Computer Engineering, Tongmyong University
[3]Dept. of Early Childhood Education, Tongmyong University

요 약 웹 서비스 사용자가 증가하면서 웹 애플리케이션을 공격하는 방법들이 여러 가지 유형들로 나타나면서 웹 애플리케이션 보안에 관한 중요성의 인식도 증가하고 있다. 정보통신기술 발달과 함께 도래한 지식정보사회는 급변하는 사회에서 창의성과 인성 교육에 필요한 웹사이트의 구축이 절실히 요구되고 있다. 본 논문에서는 창의·인성 교육기반의 디지털 큐레이션 시스템에서 제공하는 교육 콘텐츠에 대한 SQL Injection과 XSS에 대한 공격 방법과 취약점을 분석한다. 그리고 SQL Injection과 XSS에 대한 웹 공격에 대응하는 방법을 제시하고자 한다.

• Key Words : SQL Injection, XSS, 디지털 큐레이션, 웹 보안, 소셜 네트워크 서비스, 창의·인성

Abstract With the growing number of people that use web services, the perception of the importance of securing web applications is also increasing. There are many different types of attacks that target web applications. In the rapidly-changing knowledge and information society, which came into being with the advancements made in information and communication technology, there is currently an urgent need for building web sites for the purposes of developing one's creativity and character. In this paper, attack schemes that use SQL injections and XSS and target educational digital curation systems which provide educational contents with the aim of developing of one's creativity and character are analyze, in terms of how the attacks are carried out and their vulnerabilities. Furthermore, it suggests ways of dealing appropriately with these web-based attacks that use SQL injections and XSS.

• Key Words : SQL Injection, Cross-site Scripting, Digital Curation, Web security, Social Network Service, Creativity and Personality

1. 서론

정보통신기술의 발전 및 인터넷 환경의 변화로 디지털 정보자원의 생성이 기하급수적으로 늘어나고, 모든 학문분야에 걸쳐 디지털 정보자원의 제공, 활용 및 서비스가 다각적으로 이루어지고 있다. 멀티미디어시대 뉴스

*교신저자 : 신승수(shinss@tu.ac.kr)
접수일 2015년 6월 3일 수정일 2015년 8월 17일 게재확정일 2015년 8월 20일

매체는 거대 미디어부터 1인 미디어까지 다양화되어 뉴스의 양(quantity)은 폭증하고, 뉴스의 질(quality)은 떨어졌다[1]. 사용자들은 웹이라는 '정보의 바다' 속에서 허우적거리며 시간을 허비하고 있다. 정작 자신에게 유용한 정보가 무엇인지도 모르고, 정보의 진위도 파악하지 못한 채, 거대한 정보의 파도에 휩쓸려 정보에 종속되어 가고 있는 것이다. 그래서 사용자들은 누군가가 자신의 필요와 목적에 맞게 정보의 진위를 가려주고, 정보를 걸러주고, 콘텐츠를 재구성해주기를 바라게 되었다. 이렇게 '그 누군가'가 수많은 콘텐츠를 필터링하고, 게이트키핑해서 사용자의 필요와 목적에 맞게 공시(publish)하는 것에 대한 새로운 개념이 정립되고 있는데, 이러한 기능을 큐레이션이라 부르며, 그러한 사람을 큐레이터라고 명명하고 있다[2].

해외에서는 이미지 중심의 핀터레스트, 뉴스나 이슈 중심의 서미파이, 칠닷컴, 무비클립닷컴 같은 업체들이 유명세를 타고 있으며, 국내에서는 최근 판도라 TV의 '젤리캠'과 위즈메타[3]의 '비디오쿠키'가 동영상 서비스를 시작하고 있다.

정보통신기술 발달과 함께 도래한 지식정보사회는 급변하는 사회에서 창의성과 인성은 21세기 미래 교육에서 매우 중요한 역량으로 강조되고 있다. 창의성은 글로벌 시대 핵심 역량으로 지속적으로 꾸준히 강조된 개념이었는데, 최근 이러한 창의성과 인성의 두 교육의 유기적 결합을 통해, 인성개발이 곧 창의성 개발로 이어지는 상호 동반 효과에 대한 통합적 논의가 활발하게 진행되고 있다[4]. 이와 더불어 교육계가 현재 지향점으로 천명하고 있는 '창의성과 인성'을 강조하는 패러다임에서 교육 목표와 내용의 변화를 뒷받침 할 수 있는 교육 방법의 선진화가 뒤따라야 하는데 교육 분야에서 소셜 네트워크 서비스(SNS: Social Network Service)를 활용한 ICT 기술을 적용한 연구가 지속적, 장기적으로 요구되는 실정이다. 기존의 큐레이션 서비스 제공 사이트를 분석하여 각각의 특성을 파악하고, 창의・인성 교육에 확대・적용, 분석하여 창의・인성 교육기반의 디지털 큐레이션 시스템 구축이 절실히 요구된다.

창의・인성 교육기반의 디지털 큐레이션 시스템은 기존 큐레이션의 성과물과 그 과정을 기록, 증명, 평가할 수 있는 웹 기반과 다양한 매체로 구성된다. 창의・인성 교육기반의 큐레이션 시스템은 작성자의 성취와 성공에 대하여 보다 다양한 표현이 가능하므로 작성자의 독자적으로 하여금 보다 깊은 통찰을 가능하게 한다. 또한 콘텐츠의 공유 및 수정, 확장 등이 용이하다는 장점을 가지고 있다.

본 연구 과제를 수행하면서 창의・인성 교육기반의 디지털 큐레이션 시스템(http://www.cpcu.kr/)을 구축하고, 디지털 큐레이션 시스템에 대한 웹 공격의 유형과 취약점을 분석했다. OWASP(The Open Web Application Security Project) Foundation은 OWASP Top 10-2013을 발표했는데 이에 포함된 10대 취약점을 보면 Injection, XSS(Cross-site Scripting), 인증과 세션 관리 취약점 취약한 직접 객체 참조 CSRF(Cross -site request forgery), 보안 설정 오류, 민감 데이터 노출 기능 수준의 접근 통제 누락 알려진 취약점이 있는 컴포넌트 사용 검증되지 않은 리다이엑트 포워드로 나누었다. XSS는 가장 심각한 웹 애플리케이션 보안 위협 리스트에서 2위에 있다[5]. 이 Web 해킹 중에서도 XSS공격과 SQL Injection 취약점은 Web 해킹의 대표적 기법이라고 할 수 있다. XSS 공격은 현실적으로 방어하기 상당히 어렵다. 그렇기 때문에 많은 웹사이트에서 XSS의 취약점이 존재하고 악용되고 있다.

본 논문의 구성은 다음과 같다. 2장에서는 SQL Injection와 XSS의 공격 방법 형태를 기술하였고, 3장에서는 디지털 큐레이션 웹 시스템에서 SQL Injection와 XSS 공격에 대한 취약점을 기술하였다. 마지막으로 4장에서는 본 연구를 마무리하는 결론을 맺는다.

2. 관련연구

네트워크와 인터넷의 발전으로 인하여 많은 부분의 오프라인 서비스가 온라인 서비스로 전환되었으며, 현재 온라인 서비스의 대부분을 웹 서비스가 차지하고 있다. 웹이 언제 어디에서나 서비스 제공이 가능하다는 장점으로 인하여 그 비중은 날이 갈수록 증가하고 있으며, 이를 노리는 공격 또한 증가하고 있다[6].

OWASP Top 10에서 높은 위험에 속하는 SQL Injection은 클라이언트의 입력에서 발생하는 신뢰할 수 없는 데이터(명령어, 질의)의 일부분으로써 웹 어플리케이션의 CGI계층으로 보내질 때 발생하며, 공격자의 악의적인 데이터가 데이터베이스의 예기치 않은 실행이나 인가되지 않은 사용자가 민감한 데이터에 접근 할 수 있게 한다. XSS (Cross Site Scripting)은 악의적인 사용자가

특수하게 조작된 script를 사용하여 상대방의 세션정보를 가로채거나 웹 사이트 변조, 콘텐츠 삽입, 피싱, 악성코드 유포에 사용된다. 해당 취약점은 OWASP(The Open Web Application Security Project)에서 Top10으로 3년마다 발표되는 보안문제점 10개에서 위험으로 분류된다. 대부분은 웹 어플리케이션 3계층 중에서 표현계층에서 발생하며 개발단계에서 클라이언트로부터 입력받는 표현계층을 제대로 된 코드 검증을 하지 않아서 발생한다.

2.1 SQL Injection

웹 애플리케이션은 웹 브라우저에서 이용할 수 있는 응용프로그램으로 구조는 다양하지만, 일반적으로 3단계 계층(표현계층, CGI계층, 데이터베이스 계층)으로 이루어진다. 표현 계층에서는 사용자로부터 데이터를 입력받거나 데이터 처리 결과를 사용자에게 보여주는 GUI(Graphic User Interface)역할을 한다. CGI(Common Gateway Interface) 계층에서는 표현 계층과 데이터베이스 계층의 중간에 위치하며, CGI 또는 서버 측 스크립트 프로세스(Server-Side-Script Process)라고도 한다. CGI 계층은 사용자로부터 입력받은 데이터를 데이터베이스에 저장할 수 있도록 데이터를 변환처리하고, 데이터베이스에서 데이터를 처리한 결과를 표현 계층으로 전송한다. 즉, 웹 애플리케이션에서 실질적으로 데이터처리를 하는 부분은 CGI 계층이며, 서버에서 사용하는 스크립트 언어(Server-Side Script Language) (JSP, PHP, ASP)로 구성된다. 데이터베이스 계층에서는 웹 애플리케이션에서 사용자 데이터 입력과 처리 결과에 대한 모든 정보를 관리하여 보관한다. 그리고 데이터베이스 계층이 웹 애플리케이션에서의 중요 정보들을 관리 및 보관하기 때문에 인가된 사용자에게 데이터를 제공하고 비인가 사용자에게는 접근하지 못하도록 데이터를 안전하게 보호하는 역할을 한다[7].

정당한 사용자가 ID와 Password를 입력하면 표현 계층에서는 GET 또는 POST 방식으로 CGI 계층에 데이터를 전송하고, CGI 계층에서 있는 SQL 질의 처리 구문이 데이터베이스와 연결되어 정상적으로 데이터를 처리하게 된다. 이와 달리 악의적인 공격자가 ID입력 폼에 1'' or '1=1'--와 같이 입력하면, CGI 계층에 있는 질의 구문은 SELECT * FROM user WHERE id='1' or '1=1'--' AND passwd='1111';이 된다. 그러면 -- 뒤에 있는 구문은 모두 주석처리 되고, or '1=1' 때문에 항상 참이 되어 정상적인 인증 절차 없이 로그인을 할 수 있다. 이와 같이 SQL Injection 공격은 악성 데이터 값을 이용하여 기존의 웹 애플리케이션에 고정된 SQL 질의 구문을 새로운 악의적인 SQL 질의 구문으로 만들어 비정상적으로 데이터베이스에 데이터를 요청하고 처리하는 공격 방법이다. 이런 SQL Injection 공격을 예방하기 위해서 웹 개발자들은 기본적으로 입력 데이터 값을 필터링하는 방법을 사용하고 있지만, 이를 우회하는 방법들이 많이 존재하고 있기 때문에 단순 필터링 방법으로는 SQL Injection 공격을 방지하기 어렵다. 따라서 단순한 필터링 방법보다 발전된 다른 SQL Injection 공격 탐지 및 예방 방법이 필요하다[8,9,10].

SQL Injection 공격은 다른 공격들에 비해 웹 애플리케이션을 사용하거나 운영하는 시스템에는 위협적이지는 않지만 공격으로 인하여 민감한 정보를 획득하고 변조할 수 있기 때문에 국방, 은행, 전자상거래와 같은 민감한 정보를 다루는 곳에서는 매우 치명적이다. 이런 위협적인 SQL Injection 공격을 탐지하고 예방하기 위해 여러 분야에서 다양한 기법들이 연구[11,12]되고 있다.

2.2 XSS

XSS란 Cross Site Scripting의 약자이다 공격자가 상대방의 브라우저에 Script를 실행할 수 있게 하여 사용자 Session을 가로채거나 웹사이트 변조 악의적 콘텐츠 삽입 피싱 공격을 할 수 있다. 웹이 발전함에 따라 XSS 또한 지능화되고 있으며 현재도 앞으로도 가장 위협적인 Web 취약점이 될 것이다. 이 취약점이 위협적인 이유는 공격 기법 자체가 HTML과 Script를 사용하여 쉽게 공격 코드를 제작할 수 있다는 것과, 이렇게 제작된 간단한 공격 코드를 대부분의 홈페이지에 손쉽게 올릴 수 있다는 것이다.

XSS 공격은 저장(Stored) XSS와 반사(Reflected) XSS 이렇게 크게 두 가지 공격이 있다. 저장(Stored) XSS는 공격자가 XSS 취약점 공격을 위해 가장 많이 살펴보는 곳은 같은 사이트를 방문하는 다른 사용자들에게 보이는 데이터를 입력하는 부분이다. Stored XSS의 기본 방식은 공격자가 게시물에 악성 Script를 삽입한다. 사용자가 게시물을 클릭하면 공격자의 JavaScript가 포함된 응답이 전송된다. 브라우저에서 스크립트가 실행이 되고 공격자는 사용자의 쿠키 세션 등 원하는 정보를 획득하게 된다. 반사(Reflected) XSS는 URL의 CGI 인자에

Script Code를 삽입하는 것이다. 공격자가 이메일을 이용해 어떤 웹페이지 링크를 보내고 사용자가 링크를 클릭하면 그 링크에 대한 웹페이지가 화면에 나오게 된다. 그 때 웹페이지에 대한 링크 URL에 삽입된 스크립트 코드가 실행되면서 웹페이지의 내용이 변경된다. Reflected 방식의 기본적인 방식이다[14,15].

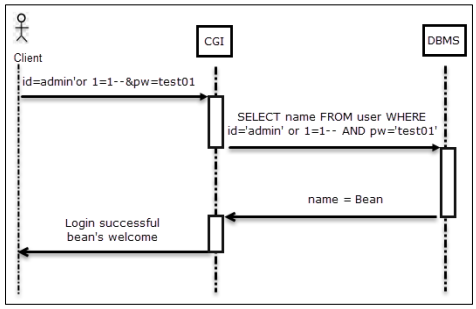
이 공격은 스크립트를 저장하기 위한 웹 사이트는 필요하지 않다. 사용자가 조작된 링크 주소를 클릭하면 링크에 대한 웹 페이지가 로드 되면서 그 스크립트 코드가 실행되기 때문이다. 위에서 보듯이 XSS는 프로그래밍 기술 측면과 복잡성을 고려할 때 공격자에게 최상의 공격방법이다.

3. 디지털 융합 큐레이션 시스템의 취약점 분석

본 장에서는 다양한 웹 공격들 중에서 SQL Injection과 XSS 공격을 이용하여 창의·인성 교육기반의 디지털 큐레이션 시스템(CPCU:http://www.cpcu.kr/)의 취약점을 분석한다. 그리고 정당한 사용자들의 개인정보 유출을 방지하기 위한 대응 방안을 제시한다.

3.1 SQL Injection 공격

제 3자가 다른 사용자의 ID를 이용하기 위해 Client은 post 방식으로 CGI계층에게 정상적인 id값 대신 "'or 1=1--"이라는 DataBase Query문을 임의의 pw와 함께 전송한다. CGI계층에서는 Client로부터 전송 받은 id와 pw값을 필터링하지 않고 DataBase Query으로 가공하여 DBMS계층으로 전송한다. DBMS는 CGI계층부터 전송받은 DataBase Query문 "SELECT name FROM user WHERE id='admin' or 1=1-- AND pw='test01'"의 정보를 이용하여 사용자를 조회한다. 전송 받은 쿼리문을 분석한 결과 id값의 admin이 존재하면 name을 반환시키는 쿼리문을 만들고, 'or 1=1--의 -- 부분은 MSSQL에서 주석을 처리는 문자이고, "AND pw="은 패스워드를 검증하는 쿼리문을 주석 처리하여 패스워드검증을 생략한다. 따라서 Client는 패스워드를 사용하지 않고 아이디만으로 로그인이 가능하다. 위와 같은 과정은 [Fig. 1]과 같다.



[Fig. 1] Flow of Data

3.1.1 정보수집 단계

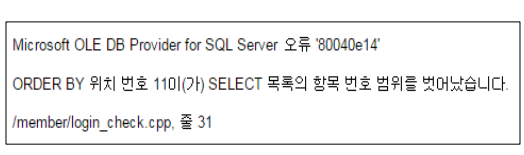
CPCU 사이트의 데이터베이스에서 해당 테이블의 컬럼 개수를 파악하기 위해서 Column Counter, One Column Search와 All Column Search를 실행하면 실행결과로부터 해당 컬럼의 데이터타입의 유형을 유출할 수 있다.

• Column Counter :

SQL Injection의 공격방법은 다양하다. 본 논문에서 제안하는 공격방법은 해당 데이터베이스 테이블의 컬럼 개수를 확보하여 원하는 데이터를 얻기 위한 정보 수집 단계이다. 공격자는 CPCU 사이트의 학번 입력란에 쿼리문 "order by 1--"를 사용하여 CPCU 사이트의 데이터베이스에서 해당 테이블의 컬럼 개수를 알아낸다. 쿼리문을 취득하는 과정은 다음과 같다.

"SELECT * FROM 테이블이름 WHERE id='' order by 1-- AND pw='test01(임의의 패스워드)'"

정상적인 order by는 데이터 정렬에 사용하는 쿼리문이다. 여기서 order by는 쿼리문 "order by 1--"을 사용하여 항목 범위 1로 주고, 나머지 쿼리 부분은 주석 처리를 한다. 'order by 2-- / ' order by 3-- / ' order by 4--,을 반복적으로 1 씩 증가시켜 마지막으로 해당 데이터베이스 컬럼의 개수를 넘게 되면 에러가 발생한다. 에러가 발생된 번호(n-1)을 하면 전체적인 해당 데이터베이스 테이블의 컬럼의 개수를 유추할 수 있다. 쿼리문에서 --는 뒤에 오는 "AND pw(패스워드체크)" 부분을 주석처리 하여 패스워드검증을 생략한다.



[Fig. 2] Column Counter

• One Column Search :

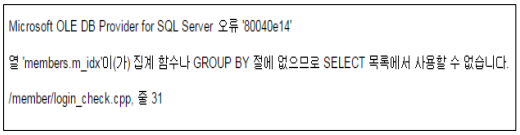
일반적인 로그인 SQL 쿼리문은

"Select * From 테이블이름 Where id='userid' And pw='userpw' "

를 사용한다. 쿼리문의 아이디 부분에 "' having 1 = 1--" 를 입력하면 내부적으로 CGI 단계에서 쿼리문은 다음과 같다.

" Select * From 테이블이름 Where id='' having 1=1-- and pw='userpw' "

정상적인 쿼리문 having은 group by와 같이 사용하는 쿼리문이다. 여기서 having은 group by가 존재하기 않기 때문에 에러가 발생한다. 그 에러의 결과 값으로 원하는 정보를 수집할 수 있다.



[Fig. 3] One Column Search

• All Column Search :

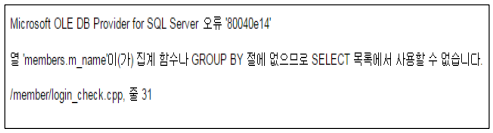
One Column Search에서 얻은 값은 데이터베이스의 테이블명(members)과 한 개의 컬럼(m_idx) 이름이다. All Column Search에서는 One Column Search에서 얻은 값으로 많은 정보를 수집하기 위해 having과 group by을 같이 사용한다. 취득한 정보(members.m_idx)로 공격 쿼리문

"'group by members.m_idx having 1=1--"

을 완성한다. 공격자가 유추 할 수 있는 쿼리문은 CGI 단계에서

"select * from members where id='' group by members.m_idx having 1=1-- and pw='a'"

이다. 여기서, id는 NULL 이고, members.m_idx를 group by로 그룹을 만든다. 그리고 "having 1=1--" 조건을 사용하면 테이블(members)에는 members.m_idx 존재하고 group by로 그룹을 했지만, 다음 두 번째 컬럼(members.m_name) 그룹에 대한 쿼리문의 요청이 존재하지 않으므로 [Fig. 4]와 같은 에러가 발생한다.



[Fig. 4] All Column Search

All Column Search를 반복적으로 하면 최종적으로 [Fig. 5]과 같이 해당 테이블과 해당 테이블의 모든 컬럼을 알아 낼 수 있다.

'group by members.m_idx,members.m_name,members.m_stunum,members.m_pw,members.m_email,members.m_univ,members.m_group,members.m_photo,members.m_joindate,members.m_comment having 1=1--

[Fig. 5] All Column Search

• Data Type :

공격 쿼리문을 사용하여 CPCU 사이트의 데이터베이스 테이블과 컬럼 정보를 모두 취득할 수 있다. 데이터타입을 알아내기 위해

"' or 1 in(select members.m_group from members)--"

라는 쿼리문을 입력하면 CGI 단계에서 다음과 같은 쿼리문이 만들어진다.

"select * from members from m_id='' or 1 in(select members.m_group from members)-- and m_pw=''"

공격자가 요청한 "' or 1 in(select members.m_group from members)--" 에서 m_group 컬럼의 내용을 숫자형으로 강제 변환시켜 에러를 유발한다. [Fig. 5]에서 m_group 컬럼의 데이터 타입은 nvarchar인 것을 알 수 있다. 이런 에러를 반복적으로 유발하여 <Table 1>과 같이 다른 컬럼들의 데이터 타입을 모두 알아 낼 수 있다.

〈Table 1〉 Information of Column

컬럼	타입
m_idx	int
m_name	nvarchar
m_stunum	nvarchar
m_pw	nvarchar
m_email	nvarchar
m_univ	nvarchar
m_group	nvarchar
m_photo	nvarchar
m_joindate	smalldatetime
m_comment	nvarchar

3.1.2 공격 단계

공격자는 Column Counter에서 Data Type까지의 모든 정보들을 획득했다. 그리고 공격단계에서는 MS-SQL의 union 쿼리문을 사용하여 원하는 회원정보를 마음대로 조회한다. CGI 단계에서 공격자가 입력한 쿼리문은 다음과 같다.

"select m_name,m_stunum,m_univ from members where m_stunum=''union all select '1',min(m_name), max(m_stunum),'2','4',min(m_group),'6','7',8,'9' from members where m_idx='167'-- And m_pw ='a'"

정상적인 union 쿼리문은 두 개의 select 구문을 하나로 묶어 보다 효율적인 쿼리문을 사용하는데 목적이 있지만 SQL Injection에서 union은 여러 방면으로 폭넓게 사용한다. union에서는 현재의 테이블의 컬럼 개수를 맞추어주어야 정상적으로 조회가 되는데, 정보수집단계에서 모든 컬럼의 정보들을 취득하였기 때문에 정보 조회가 가능하다. 공격자가 요청한 쿼리문을 분석하면 union 명령어로 select 문과 union 뒤에 오는 select 쿼리문을 하나의 쿼리로 만든다. 공격자는 쿼리문 "where m_idx='번호'"에서 번호를 사용하여 원하는 사용자를 조회할 수 있다.

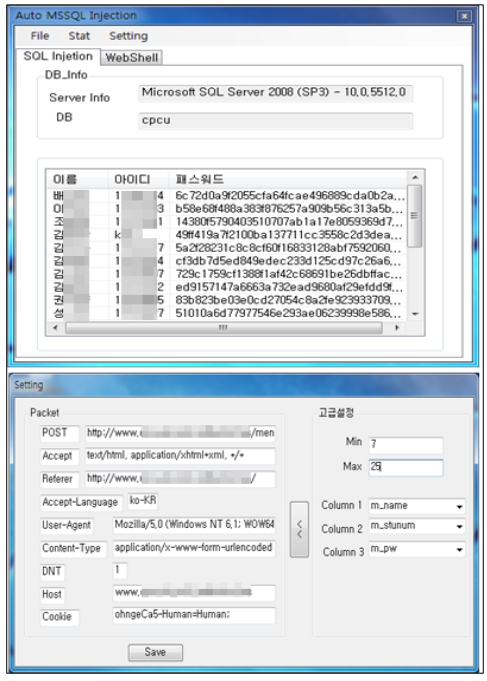
3.1.3 툴을 이용한 공격

CPCU 사이트의 취약점을 찾기 위한 여러 가지 공격들이 존재한다. 그 공격 중에서 SQL-Injection 공격을 악의적인 공격자이 수동적인 공격 방법을 보다 능동적으로 만들기 위해 보편화된 SQL Injection 자동화 툴이 아닌 특정 사이트에 맞게 SQL Injection Tool로 제작 했다. 그로인해 회원들의 개인 정보가 모두 노출되었다.

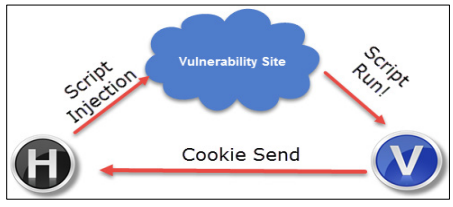
3.2 XSS(Cross Side Script)

XSS의 방법은 두 가지 공격방법이 존재한다. Stored XSS(저장)과 Reflected XSS(반사)이다. 웹 사이트의 취약점을 찾기 위해서 Stored XSS 방식을 사용하였다.

첫 번째, H(공격자)가 웹 사이트의 게시판에서 글을 작성하는 과정에서 특수하게 조작된 script를 삽입 시키면 V(일반사용자)가 웹 사이트에 로그인 후 해당 게시글에 접근을 한다면 해당 스크립트가 실행되어 쿠키값을 H로 보내는 과정은 [Fig. 7]과 같다.

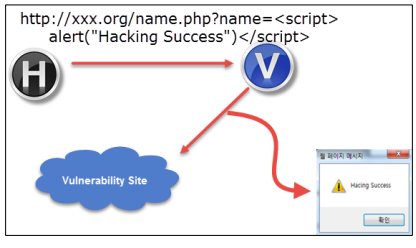


[Fig. 6] Automation Tools



[Fig. 7] Stored XSS

두 번째, H(공격자)가 특수하게 조작된 script를 V(일반사용자)에게 보내면, 일반 사용자가 클릭을 했을 경우 정상적인 URL로 접속되는 과정에서 특수하게 조작된 script가 실행되어 사용자들을 공격하는 과정은 [Fig. 8]와 같다.



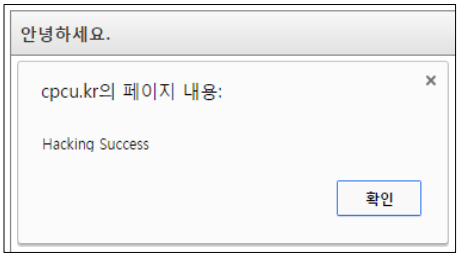
[Fig. 8] Reflected XSS

3.2.1 테스트 단계

CPCU 사이트의 취약점을 찾기 위해서 Stored XSS 방식을 사용했다. 공격자는 CPCU 사이트의 “문의하기” 페이지에서 “내용” 입력칸에

"<sCrIpt>alert("Hacking Success")</ScRiPt>"

을 입력한 뒤 “등록” 버튼을 클릭하면 데이터베이스 서버에 저장된다. 관리자는 회원들의 문의사항을 보기위해 ”문의하기“ 페이지에 접속을 하여 악의적인 사용자가 올린 ”안녕하세요“ 게시글을 클릭하는 순간 악의적인 사용자가 저장해놓은 악성스크립트도 같이 실행이 되어 [Fig. 9]과 같이 메시지 박스가 실행된다. alert을 사용하여 메시지박스가 정상적으로 실행이 되면, 해당 페이지는 Stored XSS 방식의 취약점이 존재하는 것이다. 그리고 스크립트가 실행이 되면 html의 함수 중 location.herf를 사용하여 다른 페이지로 강제 리다이렉트를 시킬 수 있으며, iframe을 이용해서도 다른 공격을 할 수 있다. 현재의 xss의 공격은 다양한 제로데이 취약점들을 이용해 웹 익스플로잇 툴킷이라는 툴 형태로 악성스크립트를 제작하여 사용자 환경에 따라 공격하는 지능적인 공격이 가능하다.



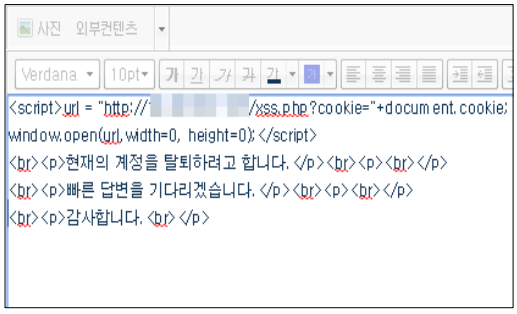
[Fig. 9] Stored XSS Test

3.2.2 공격 단계

공격자는 CPCU 사이트의 정보 수정 페이지 “문의하기” 게시판에 관리자를 유도하는 글을 작성한다. [Fig. 10]에서 공격코드로

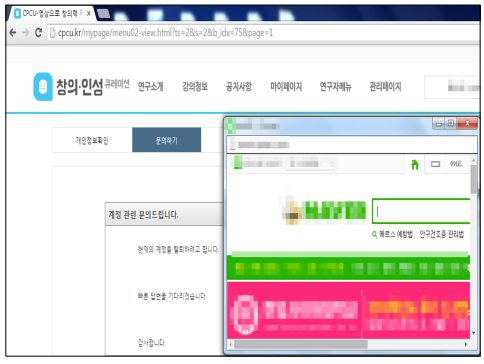
"<script>url = "http://xxx.xxx.xxx.xxx/
xss.php?cookie="+document.cookie;window.open(url,
width=0, height=0);</script>"

라는 악성 스크립트를 입력한다. 해당 악성 스크립트에서 사용하는 window.open의 함수 본래의 의미는 사용자의 편리를 위하여 팝업창을 생성하는 함수이다. 하지만, 여기서 공격자는 window.open 함수기능을 악용하여 window.open의 url을 공격자가 구축해 놓은 특정 사이트로 유도한다. cookie 파라미터 값은 “document.cookie”를 사용하여 사용자가 가지고 있는 쿠키값(또는 세션값)으로 셋팅하고 width와 height를 “0”으로 지정하여 사용자에게 보이지 않게 한다. 이러한 악의적인 행위는 관리자가 게시글을 클릭하는 순간 window.open 스크립트가 실행되어 GET 방식으로 공격자의 웹 서버로 쿠키값을 전송한다.



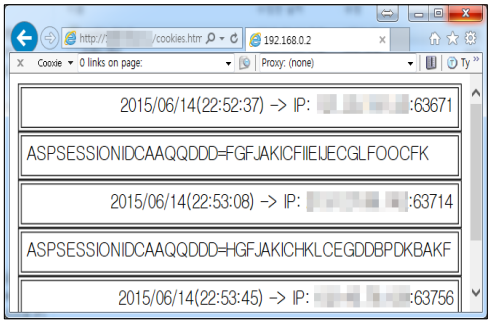
[Fig. 10] Write

공격자는 [Fig. 11]과 같이 관리자가 게시글을 클릭하여 글을 읽는 순간 악성 스크립트가 실행되고, 팝업창이 생성되면서 특정 포털 사이트로 리다이렉션 된다. 팝업창이 생성되는 동시에 쿠키값들은 공격자에게 전송이 된다.



[Fig. 11] Admin Click

공격자가 [Fig. 12]과 같이 미리 구축해 놓은 특정 웹 서버의 페이지에 CPCU 사이트 관리자의 쿠키값(또는 세션값)이 탈취된 것을 알 수 있다.



[Fig. 12] Cookie

3.3 SQL-Injection과 XSS의 대응 방법

SQL-Injection 취약점은 웹 개발자가 CGI 단계 개발 과정에서 사용자에 대한 입력값을 필터링을 하지 않기 때문에 생기는 취약점이다. 개발자 또는 관리자가 SQL-Injection 공격에 대응하는 방법을 다음과 같다.

첫 번째, SQL Injection 공격에 대응하는 방법은 사용자가 입력하는 데이터 값들이 <Table 2>와 같이 특수문자가 포함되어 있는지 검사하고, 인가되지 않은 문자열 또는 문자가 포함된 경우에는 차단한다.

두 번째, SQL 서버의 에러 메시지를 사용자에게 보여주지 않도록 설정한다. 공격자는 리턴 되는 에러 메시지를 분석하여 SQL Injection 스트링을 알아낼 수 있으므로 SQL 서버의 에러 메시지를 외부에 제공하지 않도록 한다.

〈Table 2〉 Information of Character

Special Characters	Meaning
'	Single Quartes
"	Double Quotes
/	Slash
\	Backslash
;	Semicolon
:	Colon
Space	Space Key
-	minus
+	plus

세 번째, 웹 애플리케이션을 사용하는 데이터베이스 관리자의 권한을 제한한다. 일반 권한으로 모든 system stored procedures에 접근하지 못하도록 하여 웹 애플리케이션의 SQL Injection 취약점을 이용해 데이터베이스 서버에 대한 접근이 불가능하도록 한다.

네 번째, 한국인터넷진흥원에서 무료로 제공하는 CASTLE(보안 강화 웹 방화벽)을 사용한다. 이것을 홈페이지에 적용하면, 주요 공격코드들을 차단할 수 있다.

다섯 번째, MSSQL의 확장 프로시저 (xp_ cmdshell, xplog70.dll) 파일들을 삭제한다. 해당 파일들은 관리자들이 시스템명령을 SQL에서 할 수 있도록 하는 편리기능이지만 SQL Injection의 위험이 높아 기본적으로 비활성화 되어있다. 공격자들은 활성화상태로 만들어서 SQL Injection 공격을 극대화 시킬 수 있다. 그러므로 파일을 삭제하여 잠재적인 위협을 제거한다.

XSS(Cross Side Script) 취약점은 웹 개발자가 Client 단계 개발과정에서 사용자에 대한 입력값을 필터링을 하지 않기 때문에 생기는 취약점이다. 개발자 또는 관리자가 XSS에 대응하는 방법은 다음과 같다.

첫 번째, Client 단계 개발 과정에서 개발자는 중요한 정보를 쿠키에 저장하지 않도록 한다. 만약 쿠키를 저장할 경우에는 쿠키를 암호화하여 저장한다.

두 번째, Client 단계 개발 과정에서 Script Code에 대한 특수문자를 이해하고, 완벽히 필터링을 해야 한다. 효과적인 방법은 사용자 입력이 가능한 문자를 화이트리스트 방식으로 <Table 3>와 같이 특수문자가 포함되어 있는지 검사하고, 인가되지 않은 문자열 또는 문자가 포함된 경우에는 차단한다.

〈Table 3〉 Information of ASCII

ASCII Character	Reference Character
<	&lt;
>	&gt;
&	&amp;
"	&quot;
/	&#x2F;
'	&#x27;
(	&#40;
)	&#41;
,	&#44;
%	&#37;

세 번째, 사용자 게시판에는 html 포맷 입력을 할 수 없도록 한다. 일반적인 html 게시판들은 다양하고 편리한 기능을 사용자에게 제공하기 위해 html 포맷을 지원하지만, 꼭 필요한 기능이 아니면 사용을 금지한다.

네 번째, 특정 스크립트(script)가 들어가는 문장을 치

환방식으로 바꾼다. 예로 <script>를 <xx-cript> 로 강제 치환시킨다.

4. 결론

SQL Injection 공격을 예방하기 위해서 웹 개발자들은 기본적으로 입력 데이터 값을 필터링하는 방법을 사용하고 있지만, 이를 우회하는 방법들이 많이 존재하고 있기 때문에 단순 필터링 방법으로는 SQL Injection 공격을 방지하기 어렵다. 따라서 단순한 필터링 방법보다 발전된 다른 SQL Injection 공격 탐지 및 예방 방법이 필요하다. XSS는 공격자가 상대방의 브라우저에 Script를 실행할 수 있게 하여 사용자의 Session을 가로채거나 웹 사이트 변조, 악의적 콘텐츠 삽입, 피싱 공격 등을 할 수 있다. 이런 XSS의 공격방법은 대표적으로 Stored XSS, Reflected XSS가 있다. 웹 개발자들은 철저한 관리, 제품의 업데이트 그리고 수시로 점검을 해야 한다. 웹 서비스를 받고자하는 사용자들은 인터넷의 업데이트를 통한 취약점을 보안함으로써 보안을 높일 수 있다.

ACKNOWLEDGMENTS

본 논문은 2014년 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2014S1A5B6035600)

REFERENCES

[1] Steven Rosenbaum, Curation: A Breakthrough from the Age of Info-Glut, Myungjin Publishing Co., Seoul, 2011.

[2] Jung-Sook Kim, "Video Curating Service System Using Mashup Customization Technique", Journal of Korea Multimedia Society, Vol. 17, No. 4, 2014.

[3] http://www.wizmeta.com/

[4] Fischman W, Solomon B, Greenspan D, Gardner H, Making good: how young people cope with moral dilemmas at work. Cambridge: Harvard University Press. 2004.

[5] Open Web Application Security Project(OWASP), "OWASP Top 10 for 2013", 12 June, 2013.

[6] The Open Web Application Security Project, "OWASP TOP 10 Project", http://www.owasp.org/

[7] In-yong Lee, Jae-ik Cho, Kyu-hyung Cho, Jong0sub Moon, "SA Method for SQL Injection Attack Detection using the Removal of SQL Query Attribute Values", KIISC, 18(5), pp. 135-147, 2008.

[8] Thomas. S, Williams. L, "Using Automated Fix Generation ot Secure SQL Statements", In Proceeding of the 29th international Conference on Software Engineering Workshops (ICSEW. IEEE Computer Society), pp. 54, 2007.

[9] Kosuga. Y, Kernel. K, Hanaoka. M, Hishiyama. M, Takahama. Yu, "Sania : Syntactic and Semantic Analysis for Automated Testing against SQL Injection", In Proceedings of the Computer Security Applications Conference 2007, pp. 107-117, 2007.

[10] Jae-Chul Park, Bong-Nam Noh, "QL Injection Attack Detection : Profiling of Web Application Parameter Using the Sequence Pairwise Alignment", Information Security Applications LNCS, Volume 4298, pp. 74-82, 2007.

[11] Bozic, J.; Wotawa, F., "XSS pattern for attack modeling in testing", Automation of Software Test (AST), 2013 8th International Workshop on, Vol., No., pp.71-74, May, 2013.

[12] Shahriar, H.; Zulkernine, M., "S2XS2: A Server Side Approach to Automatically Detect XSS Attacks," Dependable, Autonomic and Secure Computing(DASC), 2011 IEEE Ninth International Conference, Vol., No., pp.7-14, Dec. 2011.

[13] Hui Zhao; Wen Chen, "A Web Page Malicious Script Detection Method Inspired by the Process of Immunoglobulin Secretion", Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on , Vol., No., pp.241-245, Oct. 2010.

[14] Yi Wang; Zhoujun Li; Tao Guo, "Program Slicing Stored XSS Bugs in Web Application," Theoretical Aspects of Software Engineering (TASE), 2011 Fifth International Symposium on, Vol., No.,

pp.191-194, Aug. 2011.

[15] Sung-hyuck Hong, "XSS Attack and Countermeasure: Survey", The Journal of Digital Policy & Management, Vol. 11, No. 12, pp. 327-332, 2013.

저자소개

신 승 수(Seung-Soo Sin) [정회원]

- 2001년 2월 : 충북대학교 수학과 (이학박사)
- 2004년 2월 : 충북대학교 컴퓨터공학과(공학박사)
- 2005년 ~ 현재 : 동명대학교 정보보호학과 부교수

<관심분야> : 네트워크보안, USN, 스마트카드, 헬스케어보안.

김 정 인(Jung-In Kim) [정회원]

- 1993년 3월 : 게이오대학 계산기과학전공 공학석사
- 1996년 3월 : 게이오대학 계산기과학전공 공학박사
- 1998년 3월 ~ 현재 : 동명대학교 컴퓨터공학과 교수

<관심분야> : 기계번역, 기계학습, 시멘틱웹, 웹2.0

윤 정 진(Jeong-Jin Youn) [정회원]

- 2004년 2월 : 고려대학교 아동학전공(이학박사)
- 2006년 3월 ~ 현재 : 동명대학교 유아교육과 부교수

<관심분야> : 창의·인성, 영재교육, 뇌발달과 인지