http://dx.doi.org/10.14400/JDC.2015.13.2.177

크로스 사이트 스크립팅(XSS) 취약점에 대한 공격과 방어

최은정*, 정휘찬**, 김승엽**
서울여자대학교 정보보호학과*, 폴 수학학교**

Attacks and Defenses for Vulnerability of Cross Site Scripting

Eun-Jung Choi*, Whi-Chan Jung, Seung-Yeop Kim****
Dept. of Information Security, Seoul Women's University*
School of Paul Math International School**

요 약 크로스사이트 스크립팅은 웹 애플리케이션의 취약점으로 사용자의 정보(쿠키, 세션 등)를 탈취하거나, 비정상적인 기능을 자동으로 수행하게 하거나 할 수 있다. 크로스사이트 스크립팅(XSS) 공격유형은 한 번의 HTTP 요청과 응답에서 행해지는 XSS인 반사 XSS(reflect XSS)와 페이로드를 전송한 뒤 다수의 피해자가 해당 페이지에 접속하면 XSS 공격에 노출되는 저장 XSS(Stored XSS)로 나눌 수 있다. 그리고 크로스사이트 스크립팅 공격에 대한 방어로 사용자 입력하는 데이터에 대한 입력 값 검증, HTML 인코딩 과정에서의 출력 값에 대한 검증, 사용자가 악의적 코드를 웹 어플리케이션에 삽입하기 위한 시도를 막기 위해 위험 가능성 삽입지점 제거를 제시하였다. 본 논문에서는 이 두 가지 방법에 대한 공격방법과 절차를 제시하고 모의해킹을 수행하였다. 이를 통해 크로스사이트 스크립팅 공격에 대한 이해가 가능하고 대응책을 통해 공격에 대비할 수 있도록 하였다.

주제어 : XSS, 크로스 사이트 스크립팅, 해킹, Injection, 네트워크 보안

Abstract Cross Site Scripting enables hackers to steal other user's information (such as cookie, session etc.) or to do abnormal functions automatically using vulnerability of web application. This attack patterns of Cross Site Scripting(XSS) can be divided into two types. One is Reflect XSS which can be executed in one request for HTTP and its reply, and the other is Stored XSS which attacks those many victim users whoever access to the page which accepted the payload transmitted. To correspond to these XSS attacks, some measures have been suggested. They are data validation for user input, output validation during HTML encoding procedures, and removal of possible risk injection point to prevent from trying to insert malicious code into web application. In this paper, the methods and procedures for these two types are explained and a penetration testing is done. With these suggestions, the attack by XSS could be understood and prepared by its countermeasures.

Key Words : XSS, Cross Site Scripting, Hacking, Injection, Security

* 이 논문은 2014 학년도 서울여자대학교 컴퓨터과학 연구소 교내학술연구비에 의하여 지원되었음
Received 10 December 2014, Revised 20 January 2015
Accepted 20 February 2015
Corresponding Author: Eun-Jung Choi
(Seoul Women's University)
Email: chej@swu.ac.kr

ISSN: 1738-1916

1. 서론

정보통신 인프라의 급속한 확장으로 면대면 으로 처리되던 서비스들의 대부분이 인터넷을 기반으로 온라인을 통해 제공되고 있다. 정보검색을 시작으로 이메일, 인터넷 뱅킹, 전자결제 등의 서비스와 함께 사회 관계망 서비스인 SNS(Social Network Service), 클라우드 서비스에 이르기까지 다양하고 광범위한 서비스의 이용이 가능해졌다. 이러한 사회적 현상으로 개인정보 유출로 인한 사생활 침해, 신용카드나 금융정보 유출을 통한 금전적 손해 등의 다양한 피해 사례들도 발생하고 있다[1][2]. 정보통신 기술의 지속적인 발전을 위해서는 신뢰할 수 있고 안전성을 보장하는 보안의 문제가 기본적으로 제공되어야 한다[3][4].

가트너가 발표한 2015년 10대 전략기술을 살펴보면 클라우드 컴퓨팅, 사물인터넷 등 첨단 IT기술과 함께 위험 기반 보안과 자기 방어(Risk-Based Security and Self-Protection)의 정보보호 기술이 제시되고 있다[5]. 또한 2013년도에 발표된 OWASP(The Open Web Application Security Project)[6]의 10대 이슈를 통해서도 다양한 웹서비스 관련 문제를 알 수 있다. 다음의 <Table 1>은 OWASP의 2013년에 발표된 10개의 문제이다[7]. 특히 3위에 랭크된 Cross-Site Scripting(XSS) 취약점은 2010년에 이어 상위에 랭크되면서 주요한 해킹의 방법으로 사용되고 있다.

<Table 1> The OWASP Top 10 – 2013

A1	Injection
A2	Broken Authentication and Session Management
A3	Cross-Site Scripting (XSS)
A4	Insecure Direct Object References
A5	Security Misconfiguration
A6	Sensitive Data Exposure
A7	Missing Function Level Access Control
A8	Cross-Site Request Forgery (CSRF)
A9	Using Components with Known Vulnerabilities
A10	Unvalidated Redirects and Forwards

XSS는 웹 애플리케이션에서 발생하는 취약점의 하나로 웹사이트 관리자가 아닌 이가 웹 페이지에 악성 스크립트를 삽입하는 것이다[8]. 전자 게시판처럼 여러 사용자가 보는 공간에 악성 스크립트가 담긴 글을 게재하는 방식으로 이루어진다. 본 논문에서는 크로스사이트 스크립팅 공격과 방어에 대해 알아보고 모의해킹을 통해 XSS 공격 사례를 제시하였다.

2. 크로스사이트 스크립팅(XSS) 공격

크로스사이트 스크립팅은 웹 애플리케이션의 취약점[9][10]의 일종으로 웹 애플리케이션이 사용자로부터 입력 받은 값을 제대로 검사하지 않고 사용할 경우 나타난다. 이 취약점을 통해 해커는 사용자의 정보(쿠키, 세션 등)를 탈취하거나, 비정상적인 기능을 자동으로 수행하게 하거나 할 수 있다. 주로 다른 웹사이트와 정보를 교환하는 식으로 작동하므로 사이트 간 스크립팅이라고도 한다[11]. XSS는 HTML 인젝션의 일종으로 웹페이지의 내용을 변경하거나 피해자의 웹브라우저에 임의의 자바스크립트를 실행하는 공격이다[12]. 웹사이트에서 메일주소, 사용자 ID, 블로그 댓글, 우편번호 등과 같은 사용자의 정보를 입력 받은 후 이를 다시 보여주는 과정에서 발생한다. 따라서 XSS는 피해자의 브라우저에 스크립트를 실행할 수 있게 함으로써 공격자가 사용자의 세션을 가로채거나, 웹 사이트 변조, 악의적인 콘텐츠 삽입, 피싱 공격 행위를 할 수 있고, 악의적인 스크립팅을 이용해 사용자 브라우저를 통해 정보를 가로챌 수 있다.

2.1 공격유형

2.1.1 반사 XSS

반사 XSS(reflect XSS)는 한 번의 HTTP 요청과 응답에서 행해지는 XSS로 어떠한 정보도 저장되지 않는다. 그리고 어떤 공격 페이로드나 검색어를 이용하든 각 경우마다 새로운 결과 페이지를 생성하게 된다. 따라서 페이로드를 전송한 브라우저에 피해사례가 발생하게 된다. 결과적으로 공격자가 생성해 놓은 링크를 피해자가 클릭하게 유도하는 방식이 반사 XSS의 공격 시나리오이다.

기본적인 공격방식은 URL의 CGI 인자에 Script Code를 삽입하는 것이다. 공격자가 이메일을 이용해 어떤 웹페이지 링크를 보내고 사용자가 링크를 클릭하면 그 링크의 웹페이지가 나온다. 이 때 웹페이지 링크 URL에 삽

입된 스크립트 코드가 실행되면서 웹페이지 내용이 변조된다.

2.1.2 저장 XSS

저장(Stored) XSS라고 불리는 영구적 XSS(persistent XSS)에서는 일대다 공격이 가능하다는 점이 장점이다. 공격자가 페이로드를 전송한 뒤 다수의 피해자가 해당 페이지에 접속하면 XSS 공격에 노출된다. 반사 XSS와 영구 XSS 모두 위험하고 두 공격이 동시에 일어날 수 있다. 특히 특정 페이지에 삽입된 페이로드를 통한 XSS도 가능하다.

기본 공격 방식은 공격자가 게시물에 악성 스크립트를 삽입한다. 피해자가 게시물을 클릭하면 공격자의 자바스크립트가 포함된 응답이 전송된다. 그리고 브라우저에서 스크립트가 실행되면 공격자는 피해자의 쿠키, 세션 등의 정보를 얻을 수 있다.

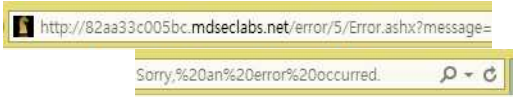
3. XSS 공격 사례

XSS 모의해킹을 mdsec서버[13]에서 진행하였다.

3.1 반사 XSS에서의 공격

앞에서 말했듯이, 반사 XSS는 웹사이트에 저장되지 않는 방법이다. 반사 XSS는 취약점에 노출된 사용자가 직접 해당 링크를 클릭하도록 유도함으로써 성공하게 된다. 일반적으로는 공격 코드가 클릭하는 링크 자체에 포함되어 있다. 링크 자체에 포함되어 공격을 할 수 있는 경우는 링크에 매개변수가 정의된 경우이다.

예를 들어, 에러 메시지를 출력하는 경우 다음의 그림 [Fig. 1]과 같이 나타난다.



[Fig. 1] URL with hacking code

[Fig. 1] 그림의 인터넷 주소에서 마지막 부분의 “message=” 부분이 매개변수로 정의된 부분이다. 이 부분 뒤에 적힌 구문 “Sorry,%20an%20error%20occurred.”은 다음과 같이 나타나게 된다. “%20”은 띄어쓰기를 뜻한다. 이 과정을 수행하면 웹브라우저는 다음의 그림 [Fig. 2]와 같은 결과가 나타난다.

[Fig. 2] Example of XSS hacking

다음으로 “message=” 이후의 구문에 스크립트 공격 코드 “<script>alert(1)</script>”를 입력하고 다시 로드하면 다음의 그림 [Fig. 3]이 나타나게 된다.

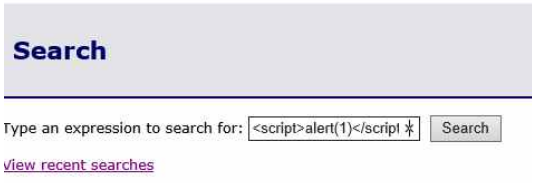
[Fig. 3] Example of XSS hacking

이렇게 되면 “Sorry, an error occurred.”라는 문장은 보이지 않고 팝업창으로 ‘1’이 출력되는 결과가 나타나게 된다. 왜냐하면 “message=” 구문에 기존에 있던 문장을 삭제하고, 스크립트 구문을 삽입했기 때문이다. 스크립트 정의 내의 ‘alert(1)’이 팝업창을 띄워 1을 출력하라는 것으로, [Fig. 3]과 같은 현상이 발생한 것이다. 이를 이용하면 사용자의 쿠키값을 얻어내거나, 개인정보를 탈취할 수 있다.

3.2 저장 XSS에서의 공격

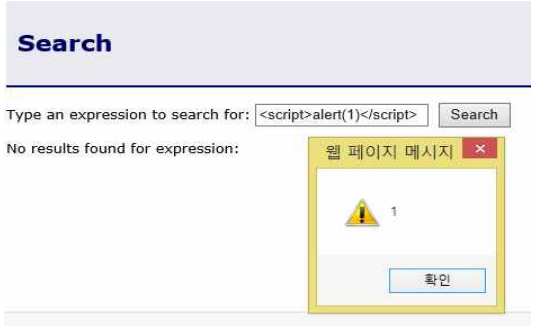
저장 XSS 공격 방법도 반사 XSS 공격 방법과 유사하다. 저장 XSS 공격 방법은 여러 가지가 있는데, 예를 들어 검색 히스토리를 통해 공격하는 방법과, 게시판 내용을 이용해 공격하는 방법이 있다.

먼저, 검색 히스토리를 통해 공격할 때는 검색창에 스크립트 공격코드를 삽입하게 된다. 스크립트 공격코드는 다음의 그림 [Fig. 4]와 같이 이전과 동일하게 'alert(1)'로 설정했다.



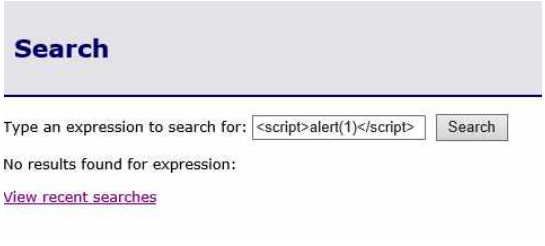
[Fig. 4] Script with hacking code

'Search' 버튼을 눌렀을 때는 다음의 그림 [Fig. 5]과 같다.



[Fig. 5] Example of XSS hacking

이전과 똑같이 팝업창에 [Fig. 5]와 같이 출력된다. 이는 스크립트 코드에 의한 것이다. 그리고 원래 'No results found for expression:' 문장 뒤에 자신이 검색한 단어가 출력되어야 하는데, 검색한 것이 스크립트 코드였으므로 아무런 내용도 출력되지 않고 팝업창만 띄워진 것이다.



[Fig. 6] Example of XSS hacking

그림 [Fig. 6]과 같이 'View recent searches'라는 링크를 클릭하게 되면 다음의 그림 [Fig. 7]과 같이 이전에 자신이 검색했던 내용들을 보여준다.

[Fig. 7] Example of XSS hacking

링크를 클릭한 즉시 그림[Fig. 6]과 같은 팝업창이 나타난다. 이는 자신이 검색했던 히스토리를 보여주는 페이지이기 때문에 검색했던 단어들이 모두 출력되기 때문이다. 'who are NGS'라는 구절과 'Search again'이라는 링크 사이에 비어있는 줄이 있는 것도 그 줄에 스크립트 구문이 출력되어야 하는데, 출력되기 이전에 스크립트가 실행되었기 때문에 줄이 비어있게 된 것이다.

커뮤니티 게시판의 내용을 이용한 XSS 공격은 필자가 서버를 구축하지 못한 관계로 실습을 진행하지 못했다. 그림 없이 공격 과정을 설명하자면, 먼저 공격자가 자유게시판에 평범하게 글을 올린다. 다만 여기서 글 내용 사이에 스크립트 공격 코드를 삽입한다. 공격자가 올린 글을 볼 수 있는 다른 사용자는 공격자의 글을 클릭해 내용을 확인하게 되고 내용 사이에 삽입되어 있는 스크립트 코드는 눈치 채지 못한 채 계속 확인하게 된다. 이 때 공격자가 삽입한 스크립트가 실행되고 게시글을 확인한 사용자의 개인정보를 탈취할 수 있게 되는 것이다.

4. 크로스사이트 스크립팅(XSS) 방어

XSS 취약점은 웹 애플리케이션과 브라우저 모두 영향을 끼친다는 점에서 다른 웹 공격과는 다르다. 다양한 크로스 사이트 스크립트 공격 방식과 가능성이 존재하지만 이 취약점을 막는 개념은 간단하다. 공격은 대개 XSS

페이로드를 게시할 웹사이트를 장악하는 것으로 시작된다. 그 뒤 페이지에 방문하는 웹 브라우저가 공격 코드에 노출된다. 그러므로 XSS 방어법은 서버와 브라우저 모두 구현되어야 한다. XSS을 막을 때 문제가 되는 부분은 사용자 제어가 가능한 데이터가 악의적 방식으로 처리될 때 확인하기 어렵다는 것이다. 웹 애플리케이션의 모든 페이지는 사용자 데이터를 처리하고 그 결과만 보여준다. 따라서 이 과정에서 XSS가 발생하게 되는 것이고 이외에도 여러 취약점이 발생할 수 있는 지점들이 있을 수 있어 크로스사이트 스크립팅 결함은 매우 광범위하게 나타나고 있다. XSS 취약점을 막을 수 있는 최고의 방법은 웹 애플리케이션 수준에서 방어막을 만드는 것이다. 그러나 HTML, JavaScript등 다양성과 복잡성 때문에 XSS 방어법을 구현하는 것은 힘들다.

반사 XSS와 저장 XSS가 발생되는 근본적 원인은 사용자가 제어 가능한 데이터를 검증 없이 웹 애플리케이션 응답으로 출력하기 때문이다. 데이터가 HTML 소스코드로 삽입되므로 악의적인 데이터는 콘텐츠뿐만 아니라 따옴표구문을 이용해서 페이지 태그를 조작하고 스크립트를 삽입하는 등 문법을 이용하여 해당 페이지의 흐름을 방해할 수 있다. 반사 XSS와 영구 XSS 취약점을 제거하기 위해 먼저 애플리케이션에서 사용자가 제어 가능한 데이터 응답으로 출력되는 모든 경우를 확인한다. 그러나 모든 경우를 확인하기 위해서 웹 애플리케이션 소스코드를 전부 면밀히 살피는 작업은 실질적 대응 방안이 아니다. 잠재적으로 XSS 위험이 있고 방어가 필요한 동작을 확인하고 실제로 취약점 발생을 막아야 한다. 이를 위해 다음의 대응 방법이 요구된다.

4.1 입력 값 검증

웹 애플리케이션은 사용자가 제공한 데이터를 받는 부분에 문맥에 맞는 검증을 수행해야 한다. 데이터 검증 항목은 다음과 같다.

첫째, 입력 데이터의 길이는 너무 길지 않아야 한다. 둘째, 특정하게 제한된 문자로 구성돼야 한다. 셋째, 특정한 규격 표현과 서로 일치해야 한다. 마지막으로 각 필드에 입력 데이터의 형태에 따라 각자 다른 검증 규칙이 적용되어야 한다.

4.2 출력 값 검증

임의의 사용자나 제 3자로부터 발생된 데이터를 애플리케이션이 응답으로 출력할 때 데이터를 안전한 상태로 만들기 위해 HTML로 인코딩해야 한다. HTML 인코딩은 알파벳 문자를 각각 맞는 HTML 엔티티(Entity)로 바꾸는 것이다. 그러면 브라우저는 인코딩된 악의적인 문자를 HTML 구조의 한 부분이 아니라 문서 내용의 한 부분으로 다루며 안전한 상태로 제어한다. 다음은 주요 문제가 있는 문자열을 HTML 인코딩한 것이다[14].

<Table 2> Character encoding

ASCII character	Reference character	ASCII character	Reference character
&	&amp;	“	&quot;
<	&lt;	‘	&#x27;
>	&gt;	/	&#x2F
(	&#40;	)	&#41;

태그 속성으로 사용자 입력을 삽입할 경우, 브라우저는 값을 처리하기 전에 HTML 디코딩을 한다. 그렇기 때문에 이런 상황에서는 모든 문제 있는 문자를 간단히 HTML 인코딩하는 것을 방어하는 것은 효과적이지 않다. 공격자는 페이로드에서 HTML 인코딩 문자를 다음과 같이 우회할 수 있다[13].

```
<img src="javascript&#58;alert(document.cookie)">
<img src="image.gif" onload="alert(&apos;xss&apos;)">
```

그러나 이런 위치에 사용자가 통제 가능한 데이터를 입력하는 것을 피해야 한다. 다양한 이유로 입력받는 것을 피할 수 없으면 필터우회를 막아야 한다.

입력 값 검증과 출력 값 검증 작업을 결합시키는 이유는 방어의 측면에서 두 단계로 방어가 가능하기 때문이다. 즉 두 개의 필터 중 한 방어 필터가 공격에 취약해도 남은 방어필터가 여전히 방어하는 것이다. 입력 값과 출력 값을 검증하는 많은 필터들이 우회될 수 있지만 이중으로 방어함으로써 두 필터 중 한 방어필터에 결함이 있어도 애플리케이션은 공격자가 공격을 성공시킬 수 없다. 두 방어 중 출력 값 검증을 먼저 생각한 뒤에 입력 값 검증을 다뤄야 한다.

4.3 위험 가능성 삽입 지점 제거

사용자의 입력 값이 애플리케이션 페이지에 삽입될 때 위험한 부분이 많다. 이런 위험한 부분들이 안전하게 기능을 발휘하게하기 위해 대응 방안을 찾아야한다.

먼저 사용자의 웹 메일 옵션에서 'JavaScript 허용여부' 등을 선택할 수 있다면 허용하지 않는 것으로 설정하는 것이다[15].

다음으로 자바스크립트를 이용하여 코딩시에 사용자가 제어 가능한 데이터를 직접 자바스크립트 안으로 삽입하지 않도록 해야 한다. 예를 들어 <script> 태그 내의 코드, 이벤트 핸들러 내의 코드 등에 적용된다. 애플리케이션이 이를 안전하게 수행하려 할 때 공격자는 방어필터를 우회할 수 있다. 그리고 공격자는 데이터 문맥의 제어권을 얻게 되면 악의적인 스크립트 명령을 삽입하거나 악의적인 동작을 실행시킬 수 있다.

공격자는 스크립트 상에 명령을 끼워 넣거나 애플리케이션의 인코딩 유형을 지정하기 위해 요청 매개 변수를 사용한다. 이 때문에 공격자가 애플리케이션 응답 인코딩 유형을 조작할 수 있는 상황은 위험하다. 때로는 이러한 상황에서 입력 값과 출력 값 필터는 잘 작동하지 못할 수 있다. 이는 필터가 악성이라고 탐지 못하는 특수한 입력 형태로 인코딩되기 때문이다. 그래서 애플리케이션은 응답 헤더 전체에서 인코딩 유형을 지정해야 한다. 따라서 다음과 같이 크로스사이트 스크립팅 필터를 적용할 수 있다[13].

```
Content-Type : text/html; charset=ISO-8859-1
```

4. 결론

본 논문에서는 크로스사이트 스크립팅(XSS) 취약점에 대해 알아보고 이에 대한 공격시나리오 및 대응법에 대해 알아보았다. 또한 시연을 통해 XSS 공격의 과정과 결과에 대해 보여 주었다. XSS는 복잡성과 프로그래밍 기술 측면에서 공격자에게 최상의 공격 수단이다. 대표적인 공격 방법으로 반사 XSS와 저장 XSS가 있다. 크로스사이트 스크립팅은 JavaScript를 이용해 쉽게 공격코드를 작성할 수 있고 윈도우, OSX, 리눅스, 인터넷 익스플로러, 사파리, 오페라 등은 모두 공격 가능한 페이로드를 작성하기 쉽다. 또한 XSS는 일반 사용자뿐만 아니라 방화벽, 안티 바이러스 소프트웨어를 사용하고 보안 패치를 설치한 사용자에게도 영향을 미친다. 웹브라우저 벤더들이 XSS를 포함한 다양한 웹 공격들의 대표적인 형태를 막기 위해 조치를 취하므로 사용자들은 최신 버전의 브라우저를 사용해야하며 최신 취약점 및 해킹에 대한 정보를 습득해야 한다. 추후에는 크로스사이트 요청위조(CSRF)와 같이 XSS 공격의 다양한 확장 형태에 대한 대응방법 가이드 및 안전한 코딩 규칙을 제시할 것이다.

ACKNOWLEDGMENTS

This work was supported by a research grant from Seoul Women's University(2014).

REFERENCES

[1] Ryan Barnett, "Full List of Incidents", Web Application Security Consortium. January 8, 2013.
[2] Brodkin, "The top 10 reasons Web sites get hacked", Network World (IDG), October 4, 2007
[3] Keun-Ho Lee, "Analysis of Threats Factor in IT Convergence Security", Journal of the Korea Convergence Society, , Vol. 1, No. 1, pp49~55, 2010
[4] Bo-Kyung Lee, "A Study on Security of Virtualization in Cloud Computing Environment for Convergence Services", Journal of the Korea Convergence Society, Vol. 5, No. 4, pp93~99, 2014
[5] http://www.gartner.com/newsroom/id/2867917
[6] https://www.owasp.org/index.php/About_OWASP
[7] https://www.owasp.org/index.php/Top10#OWASP_Top_10_for_2013
[8] Prateek Saxena, Dawn Song, and Yacin Nadji. "Document structure integrity: A robust basis for cross-site scripting defense.", In 16th Annual Network & Distributed System Security Symposium, San Diego, CA, USA, February 2009.

[9] N. Jovanovic, C. Kruegel, and E. Kirda., “Pixy: A static analysistool for detecting web application vulnerabilities (short paper).” In IEEE Symposium on Security and Privacy, 2006.
[10] Robert Auger, "Cross Site Scripting", Web Application Security Consortium. February 1, 2011.
[11] Symantec Corp, "Symantec Internet Security Threat Report: Trends for July - December 2007 (Executive Summary)" XIII. pp. 1 - 3. April 2008
[12] P. Saxena, D. Akhawe, S. Hanna, S. McCamant, F. Mao, and D. Song. A symbolic execution framework for JavaScript. In IEEE Symposium on Security and Privacy, 2010.
[13] Dafydd Stuttard, Marcus Pinto. The Web Applicaton Hacker’s Handbook: Finding and Exploiting Security Flaws. Wade AIcorn. pp. 39-70, 431-497. 2011
[14] Yunkee Seong, “Cross Site Scripting(XSS) attacks and the defenses”, Internet & SecurityFocsus, KISA, 2013. 11.
[15] SiChoon Noh, “tudy of Web Hacking Response Procedures Model based on Diagnosis Studies for Cross-Site Scripting (XSS)Process”, Journal of Information and Security, vol 13, no 6, 2013. 12.

은 정(Choi, Eun Jung)

- 1997년 2월 : 서울여자대학교 컴퓨터학(이학사)
- 2000년 2월 : 서울여자대학교 대학원 컴퓨터학(이학석사)
- 2005년 8월 : 서울여자대학교 대학원 컴퓨터학(이학박사)
- 2006년 3월 ~ 현재 : 서울여자대학교 정보보호학과 교수
- 관심분야 : 시스템보안, 암호, 빅데이터
- E-Mail : chej@swu.ac.kr

정 휘 찬(Jung, Whi Chan)

- 2014년 3월 ~ 현재 : 폴 수학학교
- 관심분야 : 보안, 해킹, 암호
- E-Mail : sakoon1231@gmail.com

김 승 엽(Kim, Seung Yeop)

- 2014년 3월 ~ 현재 : 폴 수학학교
- 관심분야 : 안드로이드 프로그래밍, 웹 보안
- E-Mail : pmath.sauvignon@gmail.com