http://dx.doi.org/10.15207/JKCS.2015.6.4.187

# 웹셸에 안전한 디지털 융합 큐레이션 시스템에 관한 연구

신승수[1*], 김정인[2], 이준연[3]
[1]동명대학교 정보보호학과, [2]동명대학교 컴퓨터공학과, [3]동명대학교 미디어공학과

# A Study on Secure Digital Convergence Curation System to WebShell

Seung-Soo Shin[1*], Jung-In Kim, Jun-Yeon Lee[3]
[1]Dept. of Information Security, Tongmyong University
[2]Dept. of Computer Engineering, Tongmyong University
[3]Dept. of Media Engineering, Tongmyong University

**요 약** 정보통신기술 발달과 함께 도래한 지식정보사회는 급변하는 사회에 대처할 수 있는 지식의 중요성이 부각되고 있다. 이와 더불어 창의성과 인성을 강조하는 패러다임에서 교육 목표와 내용의 변화를 뒷받침 할 수 있는 교육 방법의 선진화를 위해 소셜 네트워크 서비스(SNS: Social Network Service)를 활용한 ICT 기술을 적용한 연구가 지속적, 장기적으로 요구되는 실정이다. 이에 부합되는 창의・인성 교육을 확대・적용, 분석을 통해 창의・인성 교육에 기반한 디지털 큐레이션 시스템을 구축하였다. 이 디지털 큐레이션 시스템은 최근 들어 웹 해킹들이 급증하는 시점에서 웹 해킹 중 하나인 WebShell에 안전하다. 본 논문에서는 웹 해킹 중 하나인 WebShell에 안전한 디지털 큐레이션 시스템을 분석하고 WebShell에 대한 대응 방안에 대하여 분석한다.

• Key Words : WebShell, 디지털 큐레이션, 웹 보안, 소셜 네트워크 서비스, 창의·인성, ICT

**Abstract** In the knowledge and information society which came into being with the advancements made in information and communication technology, there is an increasing perception of the importance of having knowledge and therefore being able to appropriately respond to the rapidly-changing society. Along with this, for the paradigm that stresses creativity and character, there must accompany advanced ways of conducting education which are capable of supporting changes in the educational objectives and contents. With respect to this, there is a need for sustained and long-term research into ways of utilizing SNS and ICT in the field of education. Accordingly, in this paper, a digital curation system was developed for educational contents that aim to develop one's creativity and character. Recently, web hacking is taking place actively. In this paper, a digital curation system that is secure against WebShell - one of the web hacking methods - is analyzed, as well as how to appropriately deal with this type of an attack.

• Key Words : WebShell, Digital Curation, Web Security, Social Network Service, Creativity and Personality, ICT

*교신저자 : 신승수(shinss@tu.ac.kr)
접수일 2015년 5월 14일 수정일 2015년 6월 26일 게재확정일 2015년 8월 20일

## 1. 서론

최근 인터넷상에 존재하는 수많은 정보에 자신의 취향과 기준으로 좋아하는 정보를 가공하여 다른 사람에게 공유하는 소셜 큐레이션 서비스가 주목받고 있다. 소셜 큐레이션이 트위터, 페이스북과 다른 차이점은 개인의 취향을 반영한 이미지 위주의 정보 공유방식이다. 이미지 기반의 공유방식은 사용자가 쉽고 빠르게 관심 가는 정보를 찾을 수 있다는 장점을 가지고 있으나 이미지만으로 사용자의 생각과 의견으로 소통하기에는 이미지가 가진 모호성에 따라 혼란의 여지가 있다. 이는 보기 쉽게 시각화되어 있는 인터페이스임에도 불구하고 사용자에게 불편한 요소로 작용할 수 있다[1]. 정보통신기술 발달과 함께 도래한 지식정보사회는 급변하는 사회에서 창의성과 인성은 21세기 미래 교육에서 매우 중요한 역량으로 강조되고 있다[2]. 창의성은 글로벌 시대 핵심 역량으로 지속적으로 꾸준히 강조된 개념이었는데, 최근 이러한 창의성과 인성의 두 교육의 유기적 결합을 통해, 인성 개발이 곧 창의성 개발로 이어지는 상호동반 효과에 대한 통합적 논의가 활발하게 진행되고 있다. 이와 더불어 교육계 또한 현재 지향점으로 천명하고 있는 '창의성과 인성'을 강조하는 패러다임에서 교육목표와 내용의 변화를 뒷받침 할 수 있는 교육 방법의 선진화를 위해 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 교육 기반의 디지털 큐레이션 시스템 구축이 절실히 요구된다[3]. 디지털 큐레이션은 인터넷에 널린 정보들을 주제별로 혹은 관련된 연계성, 연관성을 지닌 무엇인가를 모아서 정돈하고 정리해서 스스로에게나 다른 사람에게 알기 쉽게 또 접근하기 쉽게 내보내는 작업을 말한다. 거기에 그 정보에 대한 평가 혹은 첨언 등이 들어간다면 그 디지털 큐레이션에 대한 가치는 더 높아질 것이다[4]. 지식기반과 정보화를 통해 다양한 학문과 기술들이 통합되어 새로운 지식과 가치를 창출할 것으로 전망하고 있으며 21세기 국가 경쟁력을 확보하기 위해 '통합적 사고'를 가질 수 있는 인재양성에 적극적인 학교현장의 노력이 필요하다고 보고 있다[5,6].

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[7,8]. 웹의 확대와 더불어 각종 사용자 정보 침해사건도 증가하고 있고, 웹을 통한 악성코드의 탐지 및 방지를 위한 여러 연구가 진행되고 있다[9,10,11]. 웹의 다양한 서비스에 발맞춰 이를 노리는 각종 공격들이 새로 등장하고 있기 때문이다[12]. 과거 정적인 페이지를 통하여 단순한 정보를 제공하던 시절에는 웹 서버가 해킹에 노출되어도 큰 피해가 없었지만 지금은 사정이 다르다. 웹상에서 수많은 정보들이 가공되어 처리되므로 일련의 중요하고도 의미 있는 정보들 혹은 대단히 사적인 정보들이 유출될 염려가 있다. 최근 들어 개인정보유출이나 금융권 해킹 문제가 사회적 이슈로 자주 등장하는 것도 이와 무관하지 않다[13,14,15].

본 연구 과제를 수행하면서 JSP, PHP 기반의 CPCU 웹 사이트를 구축하고, 다양한 웹 공격유형 중에서 WebShell 공격에 대해 분석한다. WebShell의 위험성과 다양한 변종 패턴들을 파악하고 대응방안을 제시한다. 본 논문의 구성은 다음과 같다. 2장에서는 WebShell에 대하여 기술하고, 3장에서는 JSP 기반의 CPCU 웹 사이트에 대한 공격방법, 4장에서는 WebShell 대응방안과 그에 대한 분석을 하였다. 마지막으로 본 연구를 마무리하는 결론을 맺는다.

## 2. 관련연구

### 2.1 WebShell

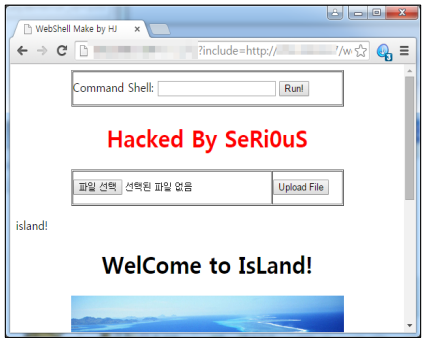
WebShell이란 공격자가 원격에서 대상 웹 서버에 명령을 수행할 수 있도록 작성한 웹 스크립트(asp, jsp, php, cgi) 파일이다. zip, jpg, doc와 같은 데이터 파일종류 이외에 악의적으로 제작된 스크립트 파일인 WebShell을 업로드하여 웹 서버를 해킹하는 사고가 빈번히 발생하고 있다. 최근에는 파일 업로드뿐만 아니라 SQL Injection과 같은 웹 취약점을 공격한 후 지속적으로 피해시스템을 관리할 목적으로 WebShell을 생성한다. 공격자는 WebShell을 대상 서버에 업로드한 후 웹을 이용하여 시스템 명령어를 수행하므로 네트워크 방화벽 영향을 받지 않고 서버를 제어할 수 있다. WebShell은 웹페이지 소스코드 열람, 악성 스크립트(iframe 등.) 삽입, 파일 업로드,

서버 및 데이터베이스 자료 유출 등의 다양한 공격이 가능하다. 최근 WebShell은 탐지를 어렵게 하기 위해 WebShell의 일부분만을 피해시스템에 업로드 하는 등 그 유형이 나날이 발전하고 있다.

웹 어플리케이션은 3단계 즉, 표현계층, CGI계층, DBMS 계층으로 구성된다[16,17]. 표현계층(Client)은 사용자로부터 데이터를 입력받아 처리 결과를 사용자에게 전송하는 GUI(Graphic User Interface)역할을 한다. CGI 또는 서버 측 스크립트 프로세스라고 불리는 CGI 계층은 표현계층과 DBMS 계층의 중간에 위치하며, 표현계층에서 전송받은 데이터를 DBMS에 저장할 수 있도록 변환 처리하고, DBMS에 데이터를 처리한 결과를 표현계층으로 전송하는 역할을 한다. 즉, 웹 애플리케이션의 3단계에서 실질적인 데이터처리 부분은 CGI 계층이며, 서버에서 사용하는 서버 스크립트 언어는 JSP, PHP, ASP로 구성된다. WebShell은 CGI계층을 공격하는 웹 애플리케이션 악성코드로 서버 스크립트 언어로 작성된다. 실제 CGI(Common Gateway Interface)계층에는 많은 취약점이 존재하고, 해커들은 취약점을 이용하여 WebShell을 서버에 업로드 시켜 서버의 권한을 획득한다. WebShell은 해커가 서버 사이드 스크립트 언어(Server-Side Script Language)로 제작하는 방법이 많이 사용되고 있다. WebShell은 간단한 서버 사이드 스크립트 언어로도 WebShell을 제작이 가능하기 때문에 많은 변종의 위험성이 존재한다. 해커는 WebShell을 실행하여 서버의 보안 시스템을 무력화 하고, 아무런 인증절차 없이 서버시스템에 침투가 가능하다. 서버에 침투한 해커는 내부네트워크까지 접근할 수 있으며, 또 다른 공격으로 연계하여 권한상승이 가능하다. 만약 권한상승에 성공한 해커는 서버의 민감한 정보까지 접근을 할 수 있다.

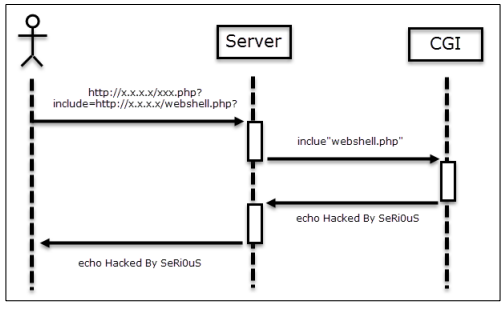
## 2.2 PHP 기반의 WebShell

PHP 기반의 웹 사이트가 사용자의 입력을 하나의 서버에서 데이터 처리를 하지 않고 다른 웹 서버의 파일을 참조하여 데이터를 처리 방식에서 취약점이 발생하는데 이것이 RFI(Remote File Include)이다. 즉, [Fig. 1]과 같다. 여기서 RFI( Remote File Include) 취약점은 외부참조 파라미터에 대한 필터링을 하지 않기 때문에 발생한다. 필터링을 하지 않기 때문에 공격자는 기존의 URL을 임의의 URL로 변조하여 WebShell에 취약한 서버에서 실행할 수 있다.




[Fig. 1] Remote File Include

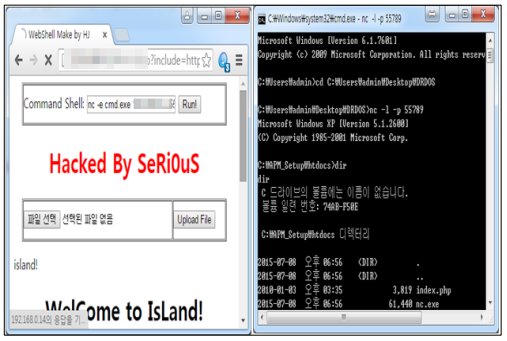
웹 서버의 파일을 참조하여 데이터를 처리하는 방식인 RFI(Remote File Include)의 취약점을 활용하여 공격하는 데이터 흐름은 [Fig. 2]와 같다. 공격자는 변조된 파라미터값을 서버에게 전송한다. 공격 대상 웹서버에서 공격자로부터 전송받은 변조된 파라미터 값을 CGI 어플리케이션은 외부파일참조 파라미터로 인식하여 서버에서 WebShell을 실행 한 뒤, 공격자에게 결과를 반환한다.




[Fig. 2] Flow of PHP Data

그 다음 단계는 공격장의 권한 상승을 위해 백도어를 설치한다. 기존의 백도어은 Bind TCP이며 Bind TCP를 응용하여 공격자는 Reverse TCP 기술을 사용한다. Bind TCP는 공격 대상 서버의 포트를 오픈하고, 침입에 용의하게 하는 장점이 있는 반면에 IDS, IPS에서 필터링 하는 경우가 있다. 서버의 인바운드 규칙은 외부에서 내부로 들어오는 모든 포트를 필터링 한다. IDS와 IPS은 서버의 인바운드 규칙을 준수하는 것으로 백도어의 침입을 차단하고 방지 하도록 되어있다. 문제점은 IDS, IPS에

서 서버의 아웃바운드 규칙에는 특별한 규칙을 설정 하지 않는다. 보통의 서버의 아웃바운드 규칙은 내부에서 외부로 나가는 모든 포트를 의미한다. [Fig. 3]에서 공격자는 서버의 아웃바인드 규칙을 준수하여 공격을 시도하기 때문에 IDS, IPS와 같은 방화벽을 우회하여 성공한다. 그리고 [Fig. 3]에서 오른쪽 부분은 공격자가 공격 대상의 서버 쉘을 원격에서 획득한 것을 보여주고 있다.




[Fig. 3] Reverse Connection

PHP WebShell의 패턴을 분석한다. [Fig. 4]에서 공격자은 임의의 서버 명령어를 post형식으로 PHP WebShell에 전송한다. PHP WebShell에서는 전송받은 데이터 중 mode값을 특정 변수에 할당하고, mode값이 특정조건과 일치 여부를 검사한다. 만약 mode의 데이터값이 upload일 경우, upload 루틴을 시작하게 된다. 업로드를 하기 위해 uploadfile변수를 할당하고 업로드가 될 경로를 생성한다. 파일 업로드의 함수 move _uploaded_file를 사용하여 업로드 된 임시파일을 공격자가 지정한 경로로 파일을 옮긴다. 그리고 mode의 데이터가 upload가 아닐 시 echo shell_exec 로 post로 전송받은 명령어를 서버에서 실행하고 echo 명령어로 출력한다.

```
<?
$mode = $_POST['mode'];
if($mode == 'upload'){
$uploaddir = './';
$uploadfile = $uploaddir. basename($_FILES['file']['name']);
if(move_uploaded_file($_FILES['file']['tmp_name'],$uploadfile))
{echo "upload success";}
else{
echo "faile";
}
}else{
echo shell_exec($_POST['cmd']);
}
?>
```

[Fig. 4] PHP WebShell Analysis

## 3. JSP 기반의 WebShell 공격

WebShell 공격을 분석하기 위해서 JSP 기반의 CPCU 웹 사이트을 구축하고, WebShell 공격을 하기 위해서 공격대상인 JSP의 정보를 수집 한다. 공격자는 수집된 정보를 바탕으로 공격방법을 구현한다. 구현된 방법으로 WebShell를 제작하고 파일을 업로드 시켜 공격을 시도한다. 공격시도에 성공한 공격자는 내부 네트워크 침투와 권한상승을 위해 특정 악성코드(이하 넷캣)을 해당 서버에 설치한다. 설치된 악성코드는 백도어 기능으로 공격자에게 해당 서버의 모든 권한을 넘겨주게 되어, 공격자는 내부정보를 취득할 수 있다.

본 장에서는 WebShell 공격을 수행하기 위해서 정보를 수집하고, JSP의 파일 업로드 취약점을 이용하여 패턴분석을 한다.

### 3.1 정보수집 단계

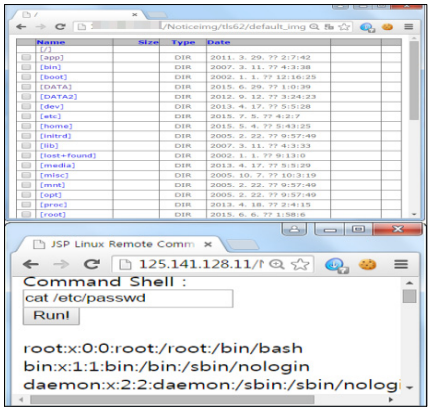
OWASP Top 10은 3년 주기로 보안의 10대 위험요소를 공개하고 있다. OWASP Top 10중에는 웹서버 관리자의 보안설정 미흡으로 발생되는 취약점인“Security Misconfiguration”이 포함되고 있다. 웹 공격을 하기 위해서 공격대상 서버의 버전과 서버종류를 수집한다. 공격자는 서버의 에러코드를 확인하고 반환된 에러코드에서 서버의 종류를 유추 할 수 있다. 유추된 내용으로 서버의 구조를 파악하고 전체적인 공격 방법을 구현 할 수 있다. 공격자는 수집된 정보를 분석하여 한가지의 방법뿐 만 아니라 다양한 공격 방법을 할 수 있다. 정보를 수집하는 방법은 다양하지만 본 논문에서는 JSP를 이용하여 정보를 수집한다.

〈Table 1〉 Information of JSP Server

| Information | Stat | Exploit | Risk |
|---|---|---|---|
| Server Operation System | Microsoft Windows \|\| CentOS Linux \|\| Ubuntu Linux \|\| Redhat Linux | N | Low |
| Server WebApplication | Apache Tomcat | Y | High |
| Server Version | 4.1.34 | Y | High |
| Server CGI Langage | jsp | Y | High |
| http status | 404 | N | Low |
| Vulnerability name | File Upload | Y | High |
| Backdoor | Bind TCP | Y | High |

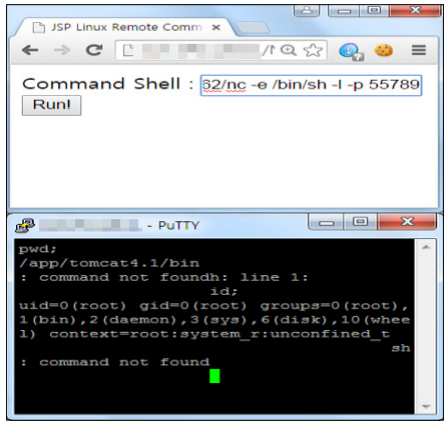
### 3.2 JSP 공격 단계

정보수집단계에서 취득한 정보를 활용하여 WebShell의 공격방법을 분석한다. JSP공격 대상은 CPCU사이트이다. 공격자는 서버의 권한을 원격에서 취득하기 위하여 이미지 파일 대신 CGI언어로 작성된 jsp파일을 커뮤니티게시판의 관리자 페이지에서 업로드 한다. 공격자는 업로드된 파일의 절대경로를 유추하고, 웹브라우저를 이용하여 업로드 한 파일의 절대경로에 접근을 시도한다. 파일업로드 취약점은 웹서버가 사용자의 파일을 신뢰하여 파일검증과 파일확장자에 대한 필터링을 하지 않기 때문에 발생한다. JSP 공격은 파일업로드, WebShell 접근 및 실행, 명령어 전송, 백도어 설치, 권한상승 등으로 실행한다. 공격자는 파일의 절대경로에 접근하여 WebShell을 실행한다. 공격자는 실행된 WebShell로 공격대상 서버에 명령어를 전송한다. 공격자는 공격대상 서버의 디렉터리 목록을 확인할 수 있고, 공격자로부터 “cat /etc/passwd” 명령어를 전송받은 공격대상 서버는 모든 계정의 정보를 공격자에게 보낸다. 이와 같은 과정은 [Fig. 5]과 같다.




[Fig. 5] File Browser & Shell

루트권한을 획득하기 위한 방법으로는 “NetCat”을 사용해야 한다. 여기서 "NetCat"은(이하 nc)은 Network connection 에서 raw-data read, write를 할 수 있는 유틸리티 프로그램이다. [Fig. 6]에서 명령어 “nc -e /bin /sh -l -p 55789” 를 분석하면 “nc는 55789포트를 대기하고 만약 공격자가 nc에 연결이 되었을 경우 서버의 권한을 공격자가 획득한다.” 공격자는 raw socket으로 해당 서버의 55789포트로 연결을 하는 동시에 서버의 권한을 넘겨받아 원격에서 실시간으로 명령어를 내릴 수 있다.




[Fig. 6] Remote Shell Connection

### 3.3 JSP WebShell 패턴 분석

공격자는 임의의 명령어를 post형식으로 jsp WebShell로 전송을 한다. 명령어를 전송받은 jsp WebShell은 request.getParameter 메소드로 String 형식으로 command 변수에 저장한다. 그리고 Runtime rt = Runtime. getRuntime();메소드를 정의하여 전송받은 명령어를 실행할 준비를 한다. jsp WebShell에서 중요한 부분인 ps = rt.exec (command);는 공격자로부터 전송받은 명령어를 서버에서 실행 한 후 결과를 ps에 반환 한다. rt.exec로부터 반환된 결과값을 읽어오기 위해 BufferedReader 메소드를 사용한다. 반복문 while ((line=br.readLine())!=null)을 사용하여 Buffered Reader 값이 null이 아닐 때 까지 결과값을 리턴한다. 최종적으로 모든 결과값을 리턴 한 뒤, br.close 메소드를 사용하여 버퍼를 닫는다.

〈Table 2〉 Information of JSP Command Pattern

| Information | Command | Risk |
|---|---|---|
| System Command Function | Runtime.getRuntime. exec | High |
| Code Encryption | URLEncoder, URLDecoder, base64Encode, base64Decode | High |
| File Create Function | FileWriter, FileReader, File, delete, | High |

```
<%
    request.setCharacterEncoding("UTF-8");
    String command = request.getParameter("cmd");
    int lineCount = 0;
    String line="";
    Runtime rt = Runtime.getRuntime();
    Process ps = null;
    try{
    ps = rt.exec(command);
    BufferedReader br =
    new BufferedReader(
    new InputStreamReader(
    new SequenceInputStream(ps.getInputStream(), ps.getErrorStream())));
     while((line = br.readLine()) != null){
%>  <%=line%><br>
<%
      }
      br.close();
   }catch(IOException ie){
      ie.printStackTrace();
   }catch(Exception e){
      e.printStackTrace();
   }
%>
```

[Fig. 7] JSP WebShell Analysis

# 4. WebShell 대응방안

## 4.1 대응 방안

WebShell 공격 방법 중에는 파일 업로드와 RFI (Remote File Include)를 사용한다. 파일 업로드는 공격자가 악성 애플리케이션을 제작하여 서버에 업로드를 하여 권한을 취득하는 방식이다. RFI방식은 관리자의 보안설정 미흡과 개발자의 실수로 인해 생기는 취약점을 이용한 PHP에서만 사용가능한 WebShell 업로드 방식이다. WebShell 공격의 대응방안으로 파일확장자 체크, 서버 보안 설정, 업로드 폴더의 실행권한 제거 등으로 취약점을 보완할 수 있다.

첫 번째, 파일확장자 체크는 공격자가 악성 애플리케이션을 업로드 할 수 없도록 필터링을 한다. 기존 파일 확장자 체크방식은 개발자들이 javascript를 이용하여 확장자 체크를 한다. 기존의 javascript는 Client-Language 이므로 공격자에 의해 충분히 변조가 가능하다. 기존의 javascript를 이용하지 않고 CGI단계에서 확장자 필터링 방법은 [Fig. 8]과 같다.

```
<script language="javascript">

function check()
{
        var filename =
        document.getElementById("file").value;
        var filetext =
        filename.substr(filename.length-3).toLowerCase();
        if(filetext == 'asp'
        || filetext == 'php'
        || filetext == 'jsp')
        {
                alert("Warning");
        }
}
```

[Fig. 8] Javascript Analysis

기존 방식의 취약점을 보완하기 위해서 서버 인증방식을 제안한다. 서버 인증방식은 Client에서 업로드를 시도하고 업로드 요청을 받은 서버는 임의 폴더에서 파일의 확장자를 추출하여 필터링을 한다. 서버 인증방식은 "pathinfo" 함수를 사용하여 파일의 절대경로를 배열로 변환한다. 변환된 배열 중 확장자를 추출 한 뒤, strotolower 함수를 사용하여 할당된 변수의 모든 데이터를 소문자로 변환하고, if 조건문을 사용하여 확장자 필터링을 한다. 서버 인증방식을 사용하면 공격자가 임의로 필터링을 우회할 수가 없으므로 기존의 취약점을 보완할 수 있다.

```
$path = pathinfo($uploadfile);
$ext = strtolower($path['extension']);
if(@ereg($ext,"php|php3|php4|htm|html")){
        echo "Warning";
}
```

[Fig. 9] PHP Upload Check

두 번째, 서버 보안설정은 관리자의 보안설정 미흡으로 인해 발생하는 취약점이다. php 관리자가 보안설정 미흡으로 RFI(Remote File Include)과 같은 취약점이 발생한다. 취약점인 RFI는 공격자가 공격대상 웹서버에 파일 업로드를 하지 않고, 공격대상 웹서버에서 공격자가 임의로 지정한 WebShell을 이용하여 공격하는 방식이다. RFI에 대한 대응방안은 서버의 보안설정을 변경하여 취약점을 보완할 수 있다. “php.ini” 파일을 편집하여 “allow_url_ fopen = On”, “allow_url _inclue = On”을 “allow_ url_fopen = OFF”, “allow_ url_inclue = OFF”로 비활성화 한 후, 서버를 다시 부팅한다.

세 번째, 실행권한 제거는 업로드 폴더의 실행권한을 제거하여 WebShell의 실행을 방지할 수 있다. 대응방법은 “Apache” 웹서버를 대상으로 업로드폴더의 CGI파일에 대한 실행권한을 제거하여 공격자의 WebShell 실행을 방지하도록 한다. 먼저 실행권한을 제거할 폴더에서 “.htaccess(hypertext access)” 파일을 생성한다. “.htaccess”은 웹서버 구성의 분산된 관리를 위해, 여러 웹서버가 지원하는 디렉터리 수준의 설정파일이다. 즉, 특정 디렉터리 접근제어 여부를 설정한다. “.htaccess”을 [Fig. 10]과 같이 편집한 후 저장한다.

```
<FilesMatch "\. (html|php|txt|jsp)">
Order allow, deny
Deny from all
</FilesMatch>
```

[Fig. 10] Server Exec Prevention

## 4.2 분석

<Table. 3>은 php와 jsp의 정보수집단계에서 비교분석을 했다. Vulnerability은 취약점의 상세정보이다. 공격자는 php의 RFI(Remote File Include)취약점, JSP에서는 FileUpload 취약점을 사용했다. RFI(Remote File Include)는 공격자가 공격대상 웹서버에서 원격의 파일을 이용하여 WebShell을 실행하고, FileUpload는 공격자가 파일을 업로드 할 경우, 일반 파일이 아닌 특수하게 조작된 악성 웹 어플리케이션을 업로드하여 WebShell을 실행한다. Server 웹 어플리케이션을 공격자가 알 수 있는 경우, Server CGI Langage 와 Server Operation System 까지 공격자가 유추 할 수 있다. Backdoor부분에서는 PHP는 Reverse TCP로 공격대상 웹서버에서 공격자 PC의 특정포트로 TCP연결을 하여 서버의 권한을 원격에서 획득할 수 있고, JSP는 Bind TCP로 공격자 PC에서 공격대상 웹서버의 특정포트로 원격 접속하여 서버의 권한을 원격에서 획득 할 수 있다.

〈Table 3〉 Information of Web Server

| | PHP | JSP |
|---|---|---|
| Server Operation System | Microsoft Windows | Redhat Linux |
| Server WebApplication | Apache | Apache Tomcat |
| Server Version | 5.2.12 | 4.1.34 |
| Server CGI Langage | PHP | jsp |
| http status | 200 | 404 |
| Vulnerability | RFI (Remote File Include) | File Upload |
| Backdoor | Reverse TCP | Bind TCP |

<Table 4>을 보면 php와 jsp의 정보수집단계에서 비교분석은 다음과 같다. System Command Function에서 PHP는 system, passthru, popen, shell_exec,exec ,proc_open 과 Runtime.getRuntime.exec 이 함수가 존재한다. php에서는 system와 exec 명령어를 WebShell 패턴에서 자주 쓰인다. 그 외 passthru, popen, proc_open 의 함수들도 잠재적 위험요소 이므로 패턴 탐지에서 중요하다. Code Encryption에서는 PHP 또는 JSP의 WebShell에서 코드 일부를 패턴탐지 우회를 위해 난독화를 한다. php의 eval, assert 와 JSP base64Encode, base64Decode 등의 함수는 난독화를 위해 사용하는 함수이므로 패턴탐지에서는 해당 함수들이 필터링이 필요하다. 마지막으로 File Create Function에서의 php는 require, require_once, fwrite, fputs 와 JSP는 FileWriter, FileReader, File, delete 등의 함수를 사용하여 파일을 생성 및 삭제를 시도한다. 이러한 함수들의 사용은 웹 서버의 민감한 부분이므로 파일의 생성, 삭제를 할 경우 서버에서 검증이 필요하다.

〈Table 4〉 Information of Command Pattern

| | PHP | JSP |
|---|---|---|
| System Command Function | system, passthru, popen, shell_exec,exec,proc_open | Runtime.getRuntime.exec |
| Code Encryption | eval, assert, gzdecode base64_decode, str_rot13 gzinflate, gzuncompress, | URLEncoder, URLDecoder, base64Encode , base64Decode |
| File Create Function | require, require_once, include, include_once, file_get_contents, fputs file_put_contents, fwrite | FileWriter, FileReader, File, delete |

## 5. 결론

WebShell 공격은 다양한 변종이 많이 존재하고 있다. 기존의 대응방법에는 javascript를 이용하여 사용자의 업로드 파일에 대한 제한을 한다. javascript는 클라이언트 언어이다. javascript는 공격자가 임의로 조작할 수 있는 취약점이 존재한다. javascript를 이용하는 대부분의 개발자의 취약점은 공격자가 중간에서 MIMT(man in the middle attack)즉, 중간자 공격이 가능하다는 것에 취약

점이 존재 한다. 중간자 공격에서는 공격자가 임의의 프록시 서버를 이용하여 현재 전송하는 모든 패킷을 변조가 가능하다. 서버에서는 클라이언트에서 전송하는 모든 패킷을 신뢰하지 않고 항상 검증을 해야 한다. 이에 본문은 파일업로드 부분에서 클라이언트 부분에서 인증이 아닌 서버에서 인증하여 보다 안정적인 서비스 보안을 제안한다. 서버에서는 javascript가 아닌 CGI 단계에서 검증하기 때문에 공격자는 변조할 수 있는 가능이 적다. 본문에서 제안하는 CGI단계에서의 문제점은 모든 Client의 검증을 서버에서 하기  때문에 서버에 대한 많은 요청을 전송할 경우 서버의 과 부화 상태가 될 가능성 존재한다. 과부화의 가능성은 존재하지만 현재 문제점은 서버에 대한 많은 요청이 존재 하지 않을 경우 안전하다. 개발자는 서버에서 중요한 검증일 경우  서버인증 방식을 사용해야 한다. 그리고 서버 인증 방식의 문제점으로는 정적인 방식으로 특정 확장자에 대한 검증을 하기 때문에 이에 대한 문제점을 보완해야한다. 특정 확장자란 php, html, htm, jsp, asp 등과 같은 CGI 단계의 언어로 구성된 웹 어플리케이션 악성코드를 말한다. 공격자는 해당 웹 어플리케이션 악성코드는 파일업로드부분을 이용하여 서버의 권한을 획득, 소스코드 유출 등 심각한 문제점을 유발한다. 이에 개발자는 본문에서 제안하는 파일 업로드 부분에서 클라이언트 부분(javascript)의 인증이 아닌 서버인증으로 사용해야한다. 이전의 많은 워드프레스 플러그인에서 파일 업로드가 아닌 RFI(Remote File Include)가 존재했다. RFI취약점은 외부의 URL을 허용, 참조를 하기 때문에 발생하는 것으로 사용자에는 이에 대한 보안패치가 존재할 수 가 없다. RFI 취약점은 사용자가 아닌 웹 어플리케이션 관리자가 패치를 해야 한다.

WebShell에서는 많은 패턴과 변종이 존재하지만 해당 업로드 폴더에 대한 실행권을 제거하여 모든 WebShell을 차단할 수 있다.

## ACKNOWLEDGMENTS

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

## REFERENCES

[1] In-hyu Jung, Soo-jin Jun, “Study in Image-Based Social Curation Interface Improvement to Build Self Identity", pp. 164-167, 2013.

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

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

[4] Haesung Lee, Joonhee Kwon “The Survey about the Personalized Curation in the Age of Big Data”, JAITC, pp. 124-126, 2013,

[5] Department of Education, Creativity and personality education plan. Seoul: Department of Education. 2009

[6]. N. Kim, Elementary school teachers' conception and management conditions on creativity and character education. Master's Thesis, Ewha Womans University. 2012.

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

[8] Mirtalebi, A.; Khayyambashi, M.R., "Enhancing security of Web service against WSDL threats," Emergency Management and Management Sciences (ICEMMS), 2011 2nd IEEE International Conference on , vol., no., pp. 920-923, Aug. 2011

[9] Darrell M. Kienzle and Matthew C. Elder. Recent worms: A survey and trends. In Proceedings of the 2003 ACM Workshop on Rapid Malcode, Washington, DC, pp. 40-49, October 2003.

[10] David Moore, Vern Paxson, Stefan Savage, Colleen Shannon an d Stuart Staniford, and Nicholas Weaver. Inside the slammer worm. IEEE Security and Privacy, 1(4), pp. 33-39, July 2003.

[11] Prabhat K. Singh and Arun Lakhotia. Analysis and detection of computer viruses and worms: An annotated bibliography. ACM SIGPLAN Notices, 37(2) pp. 29-35, February 2002.

[12] Shaikh, F.B.; Haider, S., "Security threats in cloud computing," Internet Technology and Secured Transactions (ICITST), 2011 International Conference, vol., no., pp.214-219, Dec. 11-14, 2011.
[13] C. Kaufman, M. Spiciner, and R. Perlman, Network Security Private Communication in a PUBLIC World, 2nd Edition, Englewood Cliffs, NJ : Prentice Hall, 2002.
[14] D.-Y. Kim, "Vulnerability Analysis for Industrial Control System Cyber Security," J. of the Korea Institute of Electronic Communication Sciences, vol.9, no.1, 2014, pp. 137-142.
[15] D.-K. Kang, M.-Y. Hyun, and C.-S. Kim, "Cyber trap : Unknown Attack Detection System based on Virtual Honeynet," J. of the Korea Institute of Electronic Communication Sciences, vol.8, no.6, 2013, pp. 863-871.
[16] Buehrer. G, Weide. B. W, Sivilotti. P A, "sing Parse Tree Validation to Prevent SQL Injection Attacks" In Proceedings of the 5th international Workshop on Software Engineering and Middleware, pp. 105-113, 2005.
[17] Wei. K, Muthuprasanna. M, Kothari. S, "reventing SQL injection attacks in stored procedures" Software Engineering Conference 2006. Australian, pp. 18-21, 2006.

## 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

**이 준 연(Jun-Yeon Lee)** [정회원]

- 1992년 8월 : 중앙대학교 대학원 컴퓨터공학과 (공학석사)
- 1992년 8월 ~ 1995년 8월 : Microsoft Ltd. Developer
- 2000년 2월 : 중앙대학교 대학원 컴퓨터공학과 (공학박사)
- 2000년 3월 ~ 현재 : 동명대학교 미디어공학과 교수

<관심분야> : 오피니언 마이닝, 큐레이션 시스템