웹 에디터 보안 가이드

2022. 11

과학기술정보통신부

KISA 한국인터넷진흥원

중소기업 침해사고 피해지원 서비스 PLAINBIT 코어시큐리티(주)

목 차

< 주 의 >

본 가이드에 언급된 웹 에디터 별 대응방안을 운영 중인 자사 홈페이지에 적용할 경우, 담당자는 사전에 충분한 테스트와 소스코드를 백업 후 진행하시기 바라며 수정 이후 발생한 운영장애 관련사항에 대해선 책임지지 않음을 참고하시기 바랍니다.

1. 가이드 개요

1.1. 가이드 목적 및 구성

본 가이드는 중소기업 침해사고 피해지원 서비스 사업 수행 중 홈페이지를 기반한 악성코드 유포 및 개인정보 유출, 해킹 경유지 악용 등의 주요 원인이 웹 에디터 프로그램의 취약점으로 확인되어 이에 대한 내용과 대응방안을 상세히 기술하고 배포하여 국내 홈페이지 전반의 안전성을 확보하는데 목적이 있다.

앞으로 살펴 볼 웹에디터 목록은 아래와 같으며 가장 많이 악용된 순으로 정리하였다..

1. FCKeditor	개발사: CKSource
	최신버전: FCKeditor 2.6.11
	https://ckeditor.com/blog/FCKeditor-2.6.11-Released/
2. CKeditor	개발사: CKSource
	최신버전: CKeditor5 35.0.1
	https://ckeditor.com/ckeditor-5/download/
3. SmartEditor2.0	개발사: Naver
	최신버전: SmartEditor2 2.10.0
	https://github.com/naver/smarteditor2/releases
4. G-Editor	개발사 : 그누보드 기반으로 개인 개발
	최신버전: G-Editor 1.0.0
	https://sir.kr/geditor_pds/3
5. RainEditor	개발사: Outmind(Kimjonggab)
	최신버전: RainEditor 10.0
	https://phpschool.com/link/download/15553
6. DaumEditor	개발사: kakao(daum)
	최신버전: daumeditor-7.4.32
	https://github.com/kakao/DaumEditor/tree/gh-pages/download
7. TinyMCE Editor	개발사: Tiny
	최신버전: TinyMCE 6.1.2
	https://www.tiny.cloud/get-tiny/#self-hosted
8. EZEditor	개발사 : 오픈소스
	최신버전: Ezeditor 0.2.5
	https://github.com/erzu/ez-editor

최신버전 기준일 : 2022-10-31

2. FCKeditor

2.1. 개요

2.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "FCKeditor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응 방안을 안내하는 것을 목적으로 한다. 현재 "FCKeditor"는 2014년 06월 02일 2.6.11 버전을 마지막으로 업데이트가 중단되었으며, 해당 에디터를 사용하는 기업에서는 가급적이면 최신 버전의 "CKeditor"로 업그레이드를 권고하고 있다.

2.1.2. 주요 취약점

본 가이드에 대상의 되는 "FCKeditr"의 알려진 주요 취약점은 다음과 같다. 아래의 취약점은 설정 상의 보안 결함보다는 설계 상의 결함이므로, 영향을 받는 버전 이후의 버전으로 에디터를 업데이트하는 것을 추천하는 바이다.

[표 2-1] FCKeditor의 알려진 주요 취약점

CVE ID	취약점 유형	영향을 받는 버전
CVE-2009-2324	XSS	2.6.4 이하
CVE-2009-2265	파일 노출, 실행	2.6.4 이하
CVE-2008-6178	파일 실행	2.4.3 이하
CVE-2006-6978	XSS	모든 버전
CVE-2006-2529	파일 업로드	2.2 이하
CVE-2006-0921	파일 노출	2.0 Fc
CVE-2006-0658	파일 실행	2.0 및 2.2
CVE-2005-0613	파일 업로드	2.0 Rc2

※ XSS(cross-site scripting) : 웹 애플리케이션에서 많이 나타나는 취약점의 하나로 웹 페이지에 악성 스크립트를 삽입할 수 있는 취약점이다.

2.2. 취약점 및 대응방안

2.2.1. 파일 업로드 취약점

2.2.1.1. 개요

파일 업로드 취약점은 웹 에디터의 파일 첨부 기능을 이용하여 웹셸과 같은 실행 파일을 업로드 한 후 원격에서 실행시키는 취약점을 의미한다. 대부분의 경우 업로드 되는 파일에 대한 확장자 검사 과정이 없거나, 우회하기 쉬운 경우 발생한다.

[표 2-2] 웹서버 별 실행가능 확장자

웹서버 환경	실행가능 확장자
IIS	asp, aspx, cgi, cer
PHP	php, php3, cgi, cer
JAVA	jsp, cer

2.2.1.2. 영향 받는 버전

FCKeditor 2.2 버전 이하

2.2.1.3. 해결방안

A. 최신 업데이트 적용

알려진 취약점의 경우 제조사 및 개발사에서 배포하는 최신 보안 패치가 적용된 업데이트를 적용함으로써 해당 취약점의 악용가능성을 미연에 방지할 수 있다. 또한, 웹서버가 동작하는 환경(IIS, PHP, JAVA 등)에 대한 보안 패치도 주기적으로 수행하는 것을 권고한다.

B. 샘플파일 제거 및 경로 변경

FCKeditor를 기본 디렉터리에 설치하였을 경우 및 샘플 파일을 제거하지 않았을 경우 검색엔진을 이용한 무작위 공격과 샘플파일 접근으로 인한 주요 설정 변경 및 시스템 정보 노출이 일어날 수 있다. 해당 에디터를 웹 서버에 설치하여 운용 시 기본 경로를 변경하고 샘플파일을 삭제하는 것을 권고한다.

[표 2-3] FCKeditor 주요 경로

주요 경로		
/fck/editor/	/FCKeditor/	/js/fckeditor/
/feditor/editor/	/fckeditor/editor/filemanager/browser/default/browser.html	/fckeditor/editor/filemanager/connectors/test.html
/fckeditor/editor/filemanager/connectors/uploadtest.html	/editor/filemanager/browser/default/browser.html	/editor/editor/filemanager/browser/default/browser.html
/HtmlEditor/_samples/	/FCKedtior/editor/filemanager/upload/test.html	/fckeditor4/

C. 화이트리스트 기반 확장자 검사

파일 업로드 공격을 방지하기 위해 특정 파일 유형만 허용하도록 화이트리스트 방식으로 파일 유형을 제한하여야 한다. 이때 파일의 확장자 및 업로드 된 파일의 Content-Type도 같이 확인한다.

D. 디렉터리 실행권한 제거

임의의 파일이 업로드 되더라도 실행되지 않도록 파일이 저장되는 디렉터리의 실행 권한을 제거하는 것을 권고한다.

E. 파일명 랜덤 함수 추가

공격자가 업로드된 파일 이름을 유추하여 악용할 수 없도록, 업로드 된 파일 이름에 랜덤 값을 추가한다.

[FCKeditor 1] 파일명 랜덤 함수 예시

```
// Get the uploaded file name and extension.
// $sFileName = $oFile['name'] ;
$sFileName = 'file_'.date('YmdHis').'_'.uniqid().'.'.substr(strrchr($oFile['name'], '.'), 1);
$sOriginalFileName = $sFileName ;
$sExtension = substr( $sFileName, ( strrpos($sFileName, '.') + 1 ) ) ;
$sExtension = strtolower( $sExtension ) ;
```

2.2.2. XSS 취약점

2.2.2.1. 개요

원격 공격자가 지정되지 않은 벡터를 통해 임의의 웹 스크립트 또는 html을 삽입하여 사용자의 브라우저에서 이를 실행시키는 취약점이다.

2.2.2.2. 영향 받는 버전

모든 버전

2.2.2.3. 해결방안

A. 입출력값 검증 및 무효화

웹 문서에서 폼이 작성될 때는 실행될 수 있는 코드들은 실행될 수 없는 형태로 치환하여 저장하여야 한다. "<script>" 태그를 무효화하기 위해 "<script>"와 같이 특수문자로 치환을 수행한다.

데이터 형식	콘텍스트	코딩 예시	방어책
String(문자열)	HTML Body	악성 데이터	- HTML Entity 인코딩
String(문자열)	안전한 HTML Attributes	<input type="text" name="fname" value="악성 데이터">	- HTML Entity 인코딩 - 악성 데이터를 화이트리스트 방식 또는 안전한 속성에만 위치 - Background, id 및 name과 같은 안전하지 않은 속성을 철저히 검증
String(문자열)	GET 파라미터	dickme	- URL 인코딩
String(문자열)	SRC 또는 HREF 속성에 악성 URL	dickme <iframesrc="악성 URL" />	- 입력 값 정규화 - URL 검증 - URL 안전성 확인 - 화이트리스트된 http 및 https URL만 허용(새로운 창을 열기위해 자바스크립트 프로토콜 사용 금지) - 속성 인코더 사용
String(문자열)	CSS 값	Selection	- 엄격하게 구조적 검증 - CSS 16진수 인코딩 - CSS 기능 설계 강화

String(문자열)	자바스크립트 변수	<script>var currentValue='악성 데이터';</script> <script>함수("악성 데이터");</script>	- 자바스크립트 변수를 문자화 - 자바스크립트 헥스 인코딩 - 자바스크립트 16진수 인코딩 - 자바스크립트 유니코드 인코딩 - 백슬래쉬(\ ) 사용금지
HTML	HTML Body	악성 HTML	- HTML 검증 (JSoup, AntiSamy, HTML Sanitizer)
String(문자열)	DOM XSS	<script>document.write("악성 입력 값: " + document.location.hash);<script/>	- DOM 기반 XSS 예방

[표 2-4] 입출력값 검증 및 무효화 예시

B. 보안 라이브러리 사용

AntiXSS 라이브러리와 같이 입력값을 검증하여 악성 스크립트로 입력되지 못하는 기능과 문자를 인코딩하는 함수를 제공하는 라이브러리를 사용하여 XSS 취약점에 대한 시큐어코딩을 한다.

메소드	사용처	코딩 예시
HtmlEncode	신뢰할 수 없는 데이터가 HTML 바디 부분에 출력되는 경우	[악성 데이터] Hello [악성 데이터]
HtmlAttrEncode	HTML 속성(attributes)에 신뢰할 수 없는 데이터를 추가해야 하는 경우	<input type="text" name="fname" value="[악성 데이터]">
UrlQueryEncode	URL 내에 쿼리 문자열 값에 신뢰할 수 없는 데이터를 추가해야 하는 경우	clickme JavaScriptEncode
JavaScriptEncode	신뢰할 수 없는 데이터를 자바스크립트 변수에 지정할 경우	<script>var currentValue='[악성 데이터]';</script> <script>someFunction('[악성 데이터]');</script>
XmlEncode	XML 데이터 부분에 신뢰할 수 없는 데이터를 출력하는 경우	<name>[악성 데이터]</name>
XmlAttrEncode	XML 속성 값에 신뢰할 수 없는 데이터를 추가해야 하는 경우	<name firstName="[악성 데이터]"></name>
GetSafeHtml	HTML 바디에 신뢰할 수 없는 HTML을 출력하는 경우	[악성 HTML]

[표 2-5] AntiXSS 라이브러리 예시

2.2.3. 파일 노출 및 실행 취약점

2.2.3.1. 개요

파일 노출 및 실행 취약점은 웹 어플리케이션이 잘못된 경로를 입력받아 접근 권한이 허가되지 않은 디렉터리 및 파일의 내용을 노출시키거나 외부의 악성 스크립트를 실행하는 취약점이다.

2.2.3.2. 영향 받는 버전

FCKeditor 2.6.4 버전 이하

2.2.3.3. 해결방안

A. 사용자 입력값 검증

사용자의 입력값에서 확장자를 가진 파일명이 포함되어 있거나 경로를 표현하는 슬래시("/") 또는 닷(".") 문자열이 포함되어 있을 경우 이를 제거하거나 대체하도록 한다.

B. 파일 접근에 대한 하드코딩

접근할 리소스 파일에 대한 경로를 파라미터를 통해 전달받지 말고 소스코드에 하드코딩하여 처리한다.

C. 시스템 하드닝

PHP 환경의 register_globals 옵션, allow_url_fopen, allow_url_include 옵션을 비활성화하는 등 시스템 하드닝을 설정한다.

3. CKeditor

3.1. 개요

3.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "CKEditor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응방안을 안내하는 것을 목적으로 한다.

3.1.2. 주요취약점

[표 3-1] CKeditor의 알려진 주요취약점

CVE ID	취약점 유형	영향을 받는 버전
CVE-2018-17960	XSS	CKeditor 4.11.0 이전
CVE-2018-11093	XSS	CKeditor 5-link 10.0.1 이전
CVE-2018-9861	XSS	CKeditor 4.5.10 ~ 4.9.1
CVE-2020-27193	XSS	CKeditor 4.15.0
CVE-2021-21391	정규식 서비스 거부	CKeditor 5 26.0.0 이전
CVE-2021-33829	XSS	CKEditor 4.14.0 ~ 4.16.0
CVE-2022-24728	우회 공격 취약점	CKeditor 4.18.0 이전

3.2. 취약점 및 대응방안

3.2.1. Tabnabbing 취약점

3.2.1.1. 개요

탭내빙(Tabnabbing)이란 HTML 문서 내에서 링크(target이 _blank인 태그)를 클릭했을 때 새롭게 열린 탭(페이지)에서 기존의 문서인 location을 피싱 사이트로 변경해 정보를 탈취하는 공격 기술을 말한다.

이 공격은 메일이나 SNS와 같은 오픈 커뮤니티에서 사용될 수 있다. 사용자의 클릭을 유도하여 웹 브라우저의 탭을 피싱 사이트로 이동시키는 기존의 피싱 기법과 달리 탭내빙(Tabnabbing)은 사용자가 페이지에서 아무런 행위를 하지 않아도 사용자의 눈을 피해 열린 탭 중 하나를 피싱 페이지로 로드한다.

3.2.1.2. 영향 받는 버전

CKEditor 4.5.11 미만

3.2.1.3. 대응방안

링크 내에 noopener, noreferrer 속성 추가를 권고한다.

[표 3-2] noopener, noreferrer 속성 추가

A. noopener 속성 추가 noopener 속성은 헤더와 함께 참조자 정보를 보내지 않도록 하기 때문에 window.opener를 무효화 한다. 따라서 noopener 속성이 부여된 링크를 통해 열린 페이지는 window.opener를 통해 부모 탭을 참조할 수 없고, location과 같은 자바스크립트 요청을 거부한다. <a href="http://example.com" target=_blank rel="noopener">
B. noreferrer 속성 추가 noreferrer를 지정하여 noopener와 유사한 동작을 구현할 수 있는데, noreferrer는 링크를 통해 접근 시 포함된 referrer를 전송하지 않도록 하여 링크를 클릭하더라도 referrer 정보가 유출되지 않는다. <a href="http://example.com" target=_blank rel="noreferrer">

3.2.2. XSS 취약점 (CVE-2021-33829)

3.2.2.1. 개요

4.16.1 이전 CKEditor 4 (4.14.0 ~ 4.16.x)의 XSS(cross-site scripting) 취약성은 원격 공격자가 --!> 처리가 잘못되었기 때문에 조작된 코멘트를 통해 실행 가능한 JavaScript 코드를 주입할 수 있다. CKEditor 4 버전 4.16.1로 업데이트하는 것을 적극 권고한다.

3.2.2.2. 영향 받는 버전

CKEditor 4.14.0(포함) ~ 4.16.1(제외)

3.2.2.3. 대응방안

[표 3-3] XSS 취약점 대응방안

다음과 같은 특수 문자(?,&,/,<,>, 공백)는 각각의 HTML 또는 URL로 인코딩된 동등한 항목으로 변환한다.

3.2.3. 우회공격 취약점 (CVE-2022-24728)

3.2.3.1. 개요

HTML 처리 핵심 모듈에서 취약점이 발견되었으며 4.18.0 이전 버전인 CKEditor 4에서 사용하는 모든 플러그인에 영향을 줄 수 있다. 이 취약성을 통해 콘텐츠 검사를 받지 않고 잘못된 형식의 HTML을 주입할 수 있으며, 이로 인해 JavaScript 코드가 실행될 수 있다.

3.2.3.2. 영향 받는 버전

CKEditor 4.0(포함) ~ 4.18.0(제외)

3.2.3.3. 대응방안

[표 3-4] 우회공격 취약점 대응방안

4.18.0 이상 버전으로 패치 외에 현재까지 알려진 해결책은 없는 것으로 확인된다.

3.2.4. 이미지 업로드 취약점

3.2.4.1. 개요

웹 사이트의 미흡한 이미지 업로드 정책 설정을 이용하여 서버 측에서 실행될 수 있는 스크립트 파일(asp, jsp, php 파일 등)을 업로드하고 업로드한 파일을 통해 직접 시스템 내부명령어를 실행하거나 외부와 연결하여 시스템을 제어할 수 있는 취약점이다.

3.2.4.2. 영향 받는 버전

이미지 업로드 소스 코드가 미흡한 경우 (모든 버전에 한함)

3.2.4.3. 대응방안

CKEditor는 이미지 업로드 샘플 파일을 기본으로 제공하지 않는다. 사용자가 어느정도 소스코드를 구현하여 사용해야하기 때문에 특정 버전에서 발생되는 이미지 업로드 취약점은 존재하지 않는다. 따라서 CKEditor를 이용하는 웹사이트마다 이미지 업로드 취약점 대응방안이 다를 수 있으며 다음 소스코드는 참고용으로 기재했다.

[CKeditor 3-1] upload.php 소스코드 예시 (검증 코드 삽입 전)

```php
<?php
    $uploadfullPath = "/var/www/html/editor/images/";
    $imageBaseUrl = "/editor/images/";
    $CKEditor = $_GET['CKEditor'];
    $funcNum = $_GET['CKEditorFuncNum'];
    $langCode = $_GET['langCode'];
    $url = '';
    $message = '';

    if (isset($_FILES['upload'])) {
        $name = $_FILES['upload']['name'];
        move_uploaded_file($_FILES["upload"]["tmp_name"],
$uploadfullPath . $name);
        $url = $imageBaseUrl . $name ;
    } else {
        $message = '업로드된 파일이 없습니다.';
    }
    echo "<script type='text/javascript'>
window.parent.CKEDITOR.tools.callFunction($funcNum, '$url',
'$message')</script>";
?>
```

[CKeditor 3-2] upload.php 소스코드 예시 (검증 코드 삽입 후)

```php
<?php
    $uploadfullPath = "/var/www/html/editor/images/";
    $imageBaseUrl = "/editor/images/";
    $CKEditor = $_GET['CKEditor'];
    $funcNum = $_GET['CKEditorFuncNum'];
    $langCode = $_GET['langCode'];
    $url = '';
    $message = '';
// [파일 확장자를 화이트 리스트 방식으로 체크]
function check_ext($file_ext){
    $whiteList=array('gif', 'jpg', 'png');
    $lowerFileExt=strtolower($file_ext);
    for($i=0; $i<count($whiteList); $i++){
        $isExtSafe=eregi($whiteList[$i], $lowerFileExt);
        if($isExtSafe){
            break;
        }
    }
    return $isExtSafe; // 화이트 리스트 파일 타입일 시, true 반환
}
// [MIME 타입을 화이트 리스트 방식으로 체크하는 함수 생성]
function check_type($file_type){
    $white_list=array('image/gif', 'image/jpeg', 'image/png');
    // 대소문자로 인한 파일 타입 우회 방지를 위한 소문자 변환
    $lower_file_type = strtolower($file_type);
    for($i=0; $i<count($white_list); $i++){
        $isTypeSafe=eregi($white_list[$i],$lower_file_type);
```

```
28              if($isTypeSafe){
29                  break;
30              }
31          }
32          return $isTypeSafe; // 화이트 리스트 파일 타입일 시, true 반환
33      }
34      // [랜덤 문자열 생성 함수]
35      function GenerateString($length){
36          $characters   = "0123456789";
37          $characters .= "abcdefghijklmnopqrstuvwxyz";
38          $characters .= "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
39          $characters .= "_";
40          $string_generated = "";
41          $nmr_loops = $length;
42          while ($nmr_loops--){
43              $string_generated .= $characters[mt_rand(0, strlen($characters) - 1)];
44          }
45          return $string_generated;
46      }
47
48
49      if (isset($_FILES['upload'])) {
50          $file_name = $_FILES['upload']['name'];
51          //$type = explode(".", $file_name);
52          //$file_ext = $type[count($type)-1];
53          $file_ext = strtolower(array_pop(explode('.',$file_name)));
54          $upfile_type = $_FILES['upload']['type'];
55
```

```
56  // [파일 타입과 확장자 체크]
57      if(trim($file_name) != ""){
58          $isTypeSafe=check_type($upfile_type);
59          $isExtSafe=check_ext($file_ext);
60          if(!$isTypeSafe || !$isExtSafe){ // 허용하지 않는 파일 타입 및 확장자
61  차단
62              echo "<script>alert('업로드 불허용 타입.');history.go(-1);</script>";
63              echo "<script>alert('{$isTypeSafe}');</script>";
64              echo "<script>alert('{$isExtSafe}');</script>";
65              exit;
66          }
67      }
68      $rand_name = GenerateString(30).".".$file_ext;
69      move_uploaded_file($_FILES["upload"]["tmp_name"], $uploadfullPath .
70  $rand_name);
71      $url = $imageBaseUrl . $file_name ;
72  } else {
73      $message = '업로드된 파일이 없습니다.';
74  }
75  echo "<script type='text/javascript'>
76  window.parent.CKEDITOR.tools.callFunction($funcNum, '$url',
77  '$message')</script>";
78  ?>
```

4. 네이버 SmartEditor2.0

4.1. 개요

4.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "SmartEditor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응 방안을 안내하는 것을 목적으로 한다. 현재 "SmartEditor"는 2.10.0 버전이 가장 최신 버전이며, 해당 에디터를 사용하는 기업에서는 가급적이면 최신 버전으로 업그레이드를 권고하고 있다.

4.1.2. 주요취약점

[표 4-1] SmartEditor2.0 알려진 주요취약점

CVE ID	취약점 유형	영향을 받는 버전
-	XSS	모든 버전
-	file_uploader_html5.php의 null byte injection file_uploader.php의 리다이렉트	2.8.2.3 이하
-	사진첨부 샘플의 null byte injection	2.3.10 이하
-	파일 업로드	2.3.3 이하

4.2. 취약점 및 대응방안

4.2.1. 파일 업로드 취약점

4.2.1.1. 개요

파일 업로드 취약점은 SmartEditor를 수정없이 그대로 사용하는 경우 홈페이지 해킹에 의해 홈페이지 변조, 데이터베이스 정보 유출이나 공격자가 웹셸(webshell)을 업로드하여 시스템에 침투할 가능성이 있으므로 주의가 필요하다.

※ 웹셸: php, jsp, asp 등 서버사이드 스크립트 언어로 공격자가 원격에서 임의의 명령을 실행시킬 수 있는 악성코드

※ KISA 보안공지: https://www.krcert.or.kr/data/secNoticeView.do?bulletin_writing_sequence=2406

4.2.1.2. 영향 받는 버전

SmartEditor2.0 Basic (2.3.3) 및 이전버전

4.2.1.3. 대응방안

A. 샘플파일 제거 및 경로변경

SmartEditor를 기본 디렉터리에 설치하였을 경우 및 샘플 파일을 제거하지 않았을 경우 검색엔진을 이용한 무작위 공격과 샘플파일 접근으로 인한 주요 설정 변경 및 시스템 정보 노출이 일어날 수 있다. 해당 에디터를 웹 서버에 설치하여 운용 시 기본 경로를 변경하고 샘플파일을 삭제하는 것을 권고한다.

[표 4-2] SmartEditor 샘플 파일 경로

에디터	취약점 발생경로
SmartEditor	/js/se2/SmartEditor2.html
	nse/SmartEditor2.html
	/SmartEditor2.html
	/SmartEditorBasic/
	/SmartEditor2/
	/SmartEditorBasic/SEditorDemo.html
	SEditor/popup/quick_photo/imgupload.jsp
	/smarteditor/photo_uploader/popup/file_uploader_html5.php
	/SE2/photo_uploader/popup/file_uploader_html5.php
	/smarteditor2/photo_uploader/popup/file_uploader_html5.php
	/smarteditor/popup/quick_photo/FileUploader_html5.php
	/plugin/smarteditor2/photo_uploader/popup/file_uploader_html5.php

B. 업로드 파일(file_uploader_html5.php) 검증 코드 추가 및 수정

[Smarteditor 4-1] file_uploader_html5.php 소스코드 내용

```php
<?php
    $sFileInfo = '';
    $headers = array();

    foreach($_SERVER as $k => $v) {
        if(substr($k, 0, 9) == "HTTP_FILE") {
            $k = substr(strtolower($k), 5);
            $headers[$k] = $v;
        }
    }

    $file = new stdClass;
    $file->name = str_replace("₩0", "",
rawurldecode($headers['file_name']));
    $file->size = $headers['file_size'];
    $file->content = file_get_contents("php://input");

    $filename_ext = strtolower(array_pop(explode('.',$file->name)));
    $allow_file = array("jpg", "png", "bmp", "gif");

    if(!in_array($filename_ext, $allow_file)) {
        echo "NOTALLOW_".$file->name;
    } else {
        $uploadDir = '../../upload/';
        if(!is_dir($uploadDir)){
```

[Smarteditor 4-2] 업로드 파일 검증코드 추가

```php
<?php
    $sFileInfo = '';
    $headers = array();

    foreach($_SERVER as $k => $v) {
        if(substr($k, 0, 9) == "HTTP_FILE") {
            $k = substr(strtolower($k), 5);
            $headers[$k] = $v;
        }
    }

    $file = new stdClass;
    $file->name = str_replace("₩0", "",
rawurldecode($headers['file_name']));
    $file->size = $headers['file_size'];
    $file->content = file_get_contents("php://input");
    $filename_ext = strtolower(array_pop(explode('.',$file->name)));
    $allow_file = array("jpg", "png", "bmp", "gif");

    //[파일명 랜덤 저장 함수 추가]
    function GenerateString($length){
        $characters  = "0123456789";
        $characters .= "abcdefghijklmnopqrstuvwxyz";
        $characters .= "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        $characters .= "_";
```

```
26                  mkdir($uploadDir, 0777);
27              }
28
29              $newPath = $uploadDir.iconv("utf-8", "cp949", $file->name);
30
31              if(file_put_contents($newPath, $file->content)) {
32                  $sFileInfo .= "&bNewLine=true";
33                  $sFileInfo .= "&sFileName=".$file->name;
34                  $sFileInfo .= "&sFileURL=upload/".$file->name;
35              }
36
37              echo $sFileInfo;
38          }
39      ?>
```

```
26              $string_generated = "";
27              $nmr_loops = $length;
28              while ($nmr_loops--){
29                  $string_generated .= $characters[mt_rand(0,
30      strlen($characters) - 1)];
31              }
32              return $string_generated;
33          }
34          //[MIME TYPE 검증 함수]
35          function check_type($file_type){
36              $white_list=array('image/gif', 'image/jpeg', 'image/png');
37          //대소문자로 인한 파일 타입 우회 방지를 위한 소문자 변환
38              $lower_file_type = strtolower($file_type);
39              for($i=0; $i<count($white_list); $i++){
40                  $isTypeSafe=eregi($white_list[$i],$lower_file_type);
41                  if($isTypeSafe){
42                      break;
43                  }
44              }
45              return $isTypeSafe; // 화이트 리스트 파일 타입일 시, true 반
46      환
47          }
48          //header_type 변수에 mime type 저장 (ex. "image/jpeg")
49          $header_type = $headers['file_type'];
50          //mime type 최종 검증
51          if(!check_type($header_type)){
52              echo "NOTALLOW_".$header_type;
53          }
```

```
54
55      if(!in_array($filename_ext, $allow_file)) {
56          echo "NOTALLOW_".$file->name;
57      } else {
58          $uploadDir = '../../upload/';
59          if(!is_dir($uploadDir)){
60            //[디렉토리 생성 권한 수정]
61              mkdir($uploadDir, 0644);
62          }
63
64          //<변경 전>
65          //$newPath = $uploadDir.iconv("utf-8", "cp949", $file->name);
66          //<변경후>
67          // ex) shell.php%00.png -> 랜덤문자열.png 로 저장
68        // [파일명 랜덤문자열로 저장]
69          $rand_name = GenerateString(30).".".$filename_ext;
70          $newPath = $uploadDir.iconv("utf-8", "cp949", $rand_name);
71
72          if(file_put_contents($newPath, $file->content)) {
73              $sFileInfo .= "&bNewLine=true";
74              $sFileInfo .= "&sFileName=".$file->name;
75              $sFileInfo .= "&sFileURL=upload/".$file->name;
76          }
77
78          echo $sFileInfo;
79      }
    ?>
```

4.2.2. XSS 취약점

4.2.2.1. 개요

원격 공격자가 지정되지 않은 벡터를 통해 임의의 웹 스크립트 또는 html을 삽입하여 사용자의 브라우저에서 이를 실행시키는 취약점이다.

4.2.2.2. 영향 받는 버전

모든 버전

4.2.2.3. 대응방안

A. 최신 업데이트 적용

버전 2.8.2.3 이후부터 client-side xss filter라는 기능이 추가되었다. 보안을 강화하기 위한 조치이며, 외부 소스 코드 등이 초래할 수 있는 위험에 대비하기 위함이다. 다만, 업데이트 적용 후 iframe, embed 등 일부 태그를 제거하고 있는데, 이를 원하지 않는다면 아래와 같이 코드를 수정하면 된다.

[SmartEditor 4-3] 소스 코드 수정 예시

```
nhn.husky.EZCreator.createInIFrame({
    oAppRef: oEditors,
    elPlaceHolder: "ir1",
    sSkinURI: "SmartEditor2Skin.html",
    htParams : {
        bUseToolbar : true,             // 툴바 사용 여부 (true:사용/ false:사용하지 않음)
        bUseVerticalResizer : true,     // 입력창 크기 조절바 사용 여부 (true:사용/ false:사용하지 않음)
        bUseModeChanger : true,     // 모드 탭(Editor | HTML | TEXT) 사용 여부 (true:사용/ false:사용하지 않음)
        bSkipXssFilter : true,          // client-side xss filter 무시 여부 (true:사용하지 않음 / 그외:사용)
        //aAdditionalFontList : aAdditionalFontSet,         // 추가 글꼴 목록
        fOnBeforeUnload : function(){
            //alert("완료!");
        },
        I18N_LOCALE : sLang
    }, //boolean
    fOnAppLoad : function(){
        //예제 코드
        //oEditors
```

5. G-Editor

5.1. 개요

5.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "G-Editor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응 방안을 안내하는 것을 목적으로 한다. "G-Editor"는 '2007-11-13' 경 웹 사이트 'sir.kr'에서 version 1.0.0으로 공개되었으며, 이후로 업데이트가 없는 것으로 확인된다.

5.1.2. 주요취약점

[표 5-1] G-Editor 알려진 주요취약점

CVE ID	취약점 유형	영향을 받는 버전
-	파일 업로드 취약점	G-Editor 1.0.0

5.2. 취약점 및 대응방안

5.2.1. 파일 업로드 취약점

5.2.1.1. 개요

'G-Editor'의 업로드 파일 검사는 오직 파일 시그니처만을 근거로 진행하기 때문에 시그니처를 조작한다면 업로드 검사를 우회할 수 있는 취약점을 가지고 있다. 파일 검사는 'G-Editor' 디렉토리 내의 upload.php 내에서 처리되며, getImageSize() 함수를 통해 추출한 content-type 값을 'image/png', 'image/jpeg', 'image/gif'와 비교하는 방식으로 검사를 진행한다.

5.2.1.2. 영향 받는 버전

'G-Editor'은 '2007-11-13'에 공개된 최초 버전(v1.0.0)외에 추가 버전이 공개되지 않았기 때문에 총 1개 버전이 영향을 받고 있다.

5.2.1.3. 대응방안

기존의 시그니처 기반의 파일 검사 외에 확장자 검사 및 파일명 길이 검사, 그리고 null-byte 문자열 검사 등의 구문을 추가함으로써 기존 검사 방식을 보충했다. 업로드 파일 검증과 관련된 내부 구조는 하단 표 내에 주석을 통해 표시했다.

[G-Editor 5-1] upload.php 소스 코드 (검증 코드 삽입 전)

```
1  <?php
2  // 웹 사이트에서 POST 요청을 통해 전달한 파일 가져오기
3  $file = $_FILES[image];
4
5  // 입력된 파일의 크기, Content-Type 등 속성 정보 추출
6  $size = getImageSize($file[tmp_name]);
7  // 허용할 Content-Type 값을 화이트리스트로 생성
8  $mime = array('image/png', 'image/jpeg', 'image/gif');
9
10 if (!is_uploaded_file($file[tmp_name]))
11     alert("첨부파일이 업로드되지 않았습니다.₩₩n₩₩n$file[error]");
12 //업로드 파일의 Content-Type 을 화이트 리스트와 비교
13 if (!in_array($size['mime'], $mime))
14     alert("PNG, GIF, JPG 형식의 이미지 파일만 업로드
15 가능합니다.");
16 if (!is_dir($path))
17     alert("$path 디렉토리가 존재하지 않습니다.");
18 if (!is_writable($path))
```

[G-Editor 5-2] upload.php 소스 코드 (검증 코드 삽입 후)

```
1  <?php
2  // 웹 사이트에서 POST 요청을 통해 전달한 파일 가져오기
3  $file = $_FILES[image];
4  // 전달받은 파일의 마지막 확장자 값 추출
5  $ext = end(explode('.',$file['name']));
6
7  // 입력된 파일의 크기, Content-Type 등 속성 정보 추출
8  $size = getImageSize($file[tmp_name]);
9  // 허용할 Content-Type 값을 화이트리스트로 생성
10 $mime = array('image/png', 'image/jpeg', 'image/gif');
11
12 // 허용할 확장자에 대한 화이트 리스트 생성
13 $extarr=array('png', 'jpeg', 'gif');
14 if (!is_uploaded_file($file[tmp_name]))
15     alert("첨부파일이 업로드되지 않았습니다.₩₩n₩₩n$file[error]");
16 // 업로드 파일의 확장자와 확장자 화이트리스트 비교
17 if (!in_array($ext, $extarr))
18     alert("PNG, GIF, JPG 형식의 이미지 파일만 업로드 가능합니다.");
```

```
19      alert("$path 디렉토리의 퍼미션을 707 로 변경해주세요.");
20  add_index($path);
21  $path .= '/'.date('ym');
22  ?>
23
24
25
26
27
28
29
30
31
```

```
19  //업로드 파일의 Content-Type 을 화이트 리스트와 비교
20  if (!in_array($size['mime'], $mime))
21      alert("PNG, GIF, JPG 형식의 이미지 파일만 업로드 가능합니다.");
22  //[파일명 길이 확인]
23  if(strlen($file['name'])>255)
24      alert("파일명이 너무 깁니다.")
25  if (!is_dir($path))
26      alert("$path 디렉토리가 존재하지 않습니다.");
27  if (!is_writable($path))
28      alert("$path 디렉토리의 퍼미션을 707 로 변경해주세요.");
29  add_index($path);
30  $path .= '/'.date('ym');
31  ?>
```

6. RainEditor

6.1. 개요

6.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "RainEditor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응방안을 안내하는 것을 목적으로 한다. "RainEditor"는 '2004-12-06' 경 웹 사이트 'phpschool.com'에서 version 1.0으로 공개되었으며, 이후 '2008-03-26' 경 version 10.0으로 업데이트가 진행된 후 더 이상 진행사항은 없는 것으로 확인된다.

6.1.2. 주요취약점

[표 6-1] RainEditor 알려진 주요취약점

CVE ID	취약점 유형	영향을 받는 버전
-	파일 업로드 취약점	RainEditor 2.0 이하

6.2. 취약점 및 대응방안

6.2.1. 파일 업로드 취약점

6.2.1.1. 개요

"RainEditor"의 업로드 파일 검사는 업로드된 파일의 사이즈만을 검사하여 500Kbyte 이하의 파일이면 허용되는 것으로 확인된다. 웹셸은 일반적으로 파일 사이즈가 작아, 공격자는 웹셸을 업로드할 때 제약이 발생하지 않는다.

6.2.1.2. 영향 받는 버전

RainEditor 2.0 이하

6.2.1.3. 대응방안

공격자는 악성 프로그램을 업로드 시 파일 검사를 우회하기 위해 확장자 조작, 파일 시그니처 변조, Null-Byte 문자열 사용 및 다중 확장자 사용 등의 기법을 사용한다. 이에 이러한 우회 기법에 대응하기 위한 구문을 추가함으로써 파일 검사 방식을 보충하였다. 업로드 파일 검증과 관련된 내부 구조는 하단 표 내에 주석을 통해 표시했다.

[RainEditor 6-1] upload_file.html 취약한 소스 코드 (검증 코드 삽입 전)

```
1  <?
2  include("class.RainFile.php3");
3  $file =   new RainFile("../uploads/", "editor/uploads/");
4  $imgobj = array();
5  if ($_REQUEST[url_file] != '' && eregi("^http://|^ftp://",
6  $_REQUEST[url_file])) {
7              $file_name = basename($_REQUEST[url_file]);
8              $file_server = $file ->
9  file_Copy($_REQUEST[url_file],$file_name, "junk/");
10             $file_sorce = $_REQUEST[url_file];
11 } else if (!empty($_FILES[local_file][tmp_name]) &&
12 $_FILES[local_file][tmp_name] != 'none') {
13             $file_name = $_FILES[local_file][name];
14             $file_server = $file ->
15 file_Copy($_FILES[local_file][tmp_name],$_FILES[local_file][name],
16 "junk/");
17             $file_sorce = $_FILES[local_file][name];
18 }
19
20 if ($file_server != '') {
21             $server_file = $file -> base_root.$file_server;
22             if ($img_info = @getimagesize($server_file)) {
23                     $imgobj['width'] = $img_info[0];
24                     $imgobj['height'] = $img_info[1];
25
26             } else {
27                     $imgobj['width'] = 0;
```

[RainEditor 6-2] upload_file.html 취약한 소스 코드 (검증 코드 삽입 후)

```
1
2  <?
3  include("class.RainFile.php3");
4  $file =   new RainFile("../uploads/", "editor/uploads/");
5  $imgobj = array();
6  if ($_REQUEST[url_file] != '' && eregi("^http://|^ftp://", $_REQUEST[url_file])) {
7              $file_name = basename($_REQUEST[url_file]);
8              $file_server = $file -> file_Copy($_REQUEST[url_file],$file_name,
9  "junk/");
10             $file_sorce = $_REQUEST[url_file];
11 } else if (!empty($_FILES[local_file][tmp_name]) &&
12 $_FILES[local_file][tmp_name] != 'none') {
13             $file_name = $_FILES[local_file][name];
14             $file_server = $file ->
15 file_Copy($_FILES[local_file][tmp_name],$_FILES[local_file][name], "junk/");
16             $file_sorce = $_FILES[local_file][name];
17 }
18
19 if ($file_server != '') {
20             $server_file = $file -> base_root.$file_server;
21             if ($img_info = @getimagesize($server_file)) {
22                     $imgobj['width'] = $img_info[0];
23                     $imgobj['height'] = $img_info[1];
24                     $imgobj['mime'] = $img_info['mime'];
25             } else {
26                     $imgobj['width'] = 0;
27                     $imgobj['height'] = 0;
                       $imgobj['mime'] = $img_info['mime'];
             }
             $imgobj['size']= @filesize ($server_file);
```

```
28                      $imgobj['height'] = 0;
29
30              }
31              $imgobj['size']= @filesize ($server_file);
32              $imgobj['ext'] = $file -> name2Ext($file_name);
33
34              $imgobj['filename'] = $file_name;
35              $imgobj['sorce'] = $file_sorce;
36              $imgobj['server'] = $file_server;
37  }
38
39  ?>
40
41
42
```

```
28              $imgobj['ext'] = $file -> name2Ext($file_name); // 이중 확장자
29  필터링
30              // [허용할 확장자에 대한 화이트 리스트]
31              $imgext= array('.png', '.gif','.jpg');
32              // [시그니처 검사 화이트 리스트]
33              $imgmime= array('image/png', 'image/gif','image/jpg');
34              $imgobj['filename'] = $file_name;
35              $imgobj['sorce'] = $file_sorce;
36              $imgobj['server'] = $file_server;
37
38              if ($imgobj['size'] > 500000) {
39                      $error_msg = "500 Kb 이하의 파일만 첨부 가능합니다";
40                      $imgobj = array();
41                      @unlink($server_file);
42              }
43  }
44  //[확장자 검사 및 mime 타입 검사 함수]
45  if (!in_array($imgobj['ext'], $imgext)){
46              $error_msg = " PNG, GIF, JPG 형식의 이미지 파일만 업로드
47  가능합니다.";
48              $imgobj = array();
49              @unlink($server_file);
50  }
51  if (!in_array($imgobj['mime'], $imgmime)){
52              $error_msg = " PNG, GIF, JPG 형식의 이미지 파일만 업로드
53  가능합니다.";
54              $imgobj = array();
55              @unlink($server_file);
56  }
57  ?>
```

7. DaumEditor

7.1. 개요

7.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "DaumEditor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응방안을 안내하는 것을 목적으로 한다. 현재 "DaumEditor"는 2016년 03월 03일부로 마지막 업데이트가 중단되었다.

7.1.2. 주요취약점

[표 7-1] DaumEditor 알려진 주요취약점

CVE ID	취약점 유형	영향을 받는 버전
-	파일 업로드 취약점	모든 버전

7.2. 취약점 및 대응방안

7.2.1. 파일 업로드 취약점

7.2.1.1. 개요

파일 업로드 취약점은 웹 에디터의 파일 첨부 기능을 이용하여 웹셸과 같은 실행 파일을 업로드 한 후 원격에서 실행시키는 취약점을 의미한다. "DaumEditor"의 경우 파일 업로드 기능을 기본으로 제공하지 않는다. 사용자가 업로드 기능을 추가할 때 업로드 되는 파일에 대한 확장자 검사를 취약하게 수행하는 경우 발생한다.

7.2.1.2. 영향 받는 버전

모든 버전

7.2.1.3. 대응방안

DaumEditor의 경우 이미지 업로드 샘플 파일을 기본으로 제공하지 않는다. 사용자가 어느정도 소스코드를 구현하여 사용해야하기 때문에 특정 버전에서 발생되는 이미지 업로드 취약점은 존재하지 않는다. 따라서 DaumEditor를 이용하는 웹사이트마다 이미지 업로드 취약점 대응방안이 다를 수 있으며 다음 소스코드는 참고용으로 사용하기 바란다.

A. 파일 확장자 검증

업로드되는 파일의 확장자를 검증하여 서버에서 실행될 수 있는 실행 파일들의 업로드를 막아야 악성파일(ex. 웹셸)등이 서버에 업로드 및 실행되는 것을 막을 수 있다.

B. 업로드 파일명 난수화

업로드되는 파일의 이름을 난수화 후 서버에 저장하면 공격자에 의해서 업로드된 악성파일을 외부에서 호출하는 것을 막을 수 있다.

[DaumEditor 7-1] 검증코드 삽입된 upload 파일 소스코드

```php
<?php
// 업로드될 파일의 경로를 지정한다.
$targetFolder = "/custom/uploads";
if(!empty($_FILES)) {
        //업로드되는 파일명 난수화 기능
        $str = "ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890abcdefghijklmnopqrstuvwxyz";
        $str = str_shuffle($str);
        $str = substr($str, 0, 30);

        $upName = $str.$_FILES['CoverImage']['name'];
        $tempFile = $_FILES['CoverImage']['tmp_name'];
        $targetPath = $_SERVER['DOCUMENT_ROOT'] . $targetFolder;
        $targetFile = rtrim($targetPath, "/")."/".$upName;

        // 업로드 할 수 있는 파일의 확장자를 지정한다.
        $fileTypes = array('jpg', 'jpeg', 'gif', 'png');
        $fileParts = pathinfo($_FILES['CoverImage']['name']);

        if(in_array($fileParts['extension'], $fileTypes)) {
                move_uploaded_file($tempFile, $targetFile);
                echo $upName;
        } else {
                echo "업로드 할 수 없는 형식의 파일입니다.";
        }
}
?>
```

8. TinyMCE Editor

8.1. 개요

8.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "TinyMCE Editor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응방안을 안내하는 것을 목적으로 한다.

8.1.2. 주요취약점

[표 8-1] TinyMCE Editor 알려진 주요취약점

CVE ID	취약점 유형	영향을 받는 버전
-	파일 업로드 취약점	-

8.2. 취약점 및 대응방안

8.2.1. 파일 업로드 취약점

8.2.1.1. 개요

파일 업로드 취약점은 웹 에디터의 파일 첨부 기능을 이용하여 웹셸과 같은 실행 파일을 업로드 한 후 원격에서 실행시키는 취약점을 의미한다.

8.2.1.2. 영향 받는 버전

이미지 업로드 소스 코드가 미흡한 경우 (모든 버전에 한함)

8.2.1.3. 대응방안

TinyMCE Editor는 이미지 업로드 샘플 파일을 기본으로 제공하지 않는다. 사용자가 어느정도 소스코드를 구현하여 사용해야하기 때문에 특정 버전에서 발생되는 이미지 업로드 취약점은 존재하지 않는다. 따라서, TinyMCE Editor를 이용하는 웹사이트마다 이미지 업로드 취약점 대응방안이 다를 수 있으며 다음 소스코드는 참고용으로 기재했다.

[TinyMCE Editor 8-1] upload.php 소스코드 예시

```php
<?php
        // Allowed the origins to upload
        // $accepted_origins 변수 알맞게 변경 필요
        $accepted_origins = array("http://192.168.223.154");
        // Images upload dir path
        $imgFolder = "uploads/";
        reset($_FILES);
        $tmp = current($_FILES);
        if(is_uploaded_file($tmp['tmp_name'])){
            if(isset($_SERVER['HTTP_ORIGIN'])){
                if(in_array($_SERVER['HTTP_ORIGIN'], $accepted_origins)){
                    header('Access-Control-Allow-Origin: ' . $_SERVER['HTTP_ORIGIN']);
                }else{
                    header("HTTP/1.1 403 Origin Denied");
                    return;
                }
            }
            // check valid file name
            if(preg_match("/([^₩w₩s₩d₩-_~,;:₩[₩]₩(₩).])|([₩.]{2,})/", $tmp['name'])){
                header("HTTP/1.1 400 Invalid file name.");
                return;
            }
            // check and Verify extension
            if(!in_array(strtolower(pathinfo($tmp['name'], PATHINFO_EXTENSION)), array("gif", "jpg",
"png"))){
                header("HTTP/1.1 400 Invalid extension.");
                return;
            }

            // Accept upload if there was no origin, or if it is an accepted origin
            $filePath = $imgFolder . $tmp['name'];
            move_uploaded_file($tmp['tmp_name'], $filePath);
            // return successful JSON.
            echo json_encode(array('location' => $filePath));
        } else {
            header("HTTP/1.1 500 Server Error");
        }
?>
```

9. EzEditor

9.1. 개요

9.1.1. 가이드 목적 및 권고사항

본 웹 에디터 보안가이드는 "ezEditor"를 대상으로 작성되었으며, 해당 에디터의 주요 보안취약점 및 대응 방안을 안내하는 것을 목적으로 한다. 현재 "ezEditor"는 2015년 12월 23일 0.2.6 버전을 마지막으로 업데이트가 중단되었으며, 해당 에디터를 사용하는 기업에서는 가급적이면 최신 버전으로 업그레이드를 권고하고 있다.

9.1.2. 주요 취약점

본 가이드에 대상의 되는 "ezEditor"의 알려진 주요 취약점은 다음과 같다. 아래의 취약점은 설정 상의 보안 결함보다는 설계 상의 결함이므로, 영향을 받는 버전 이후의 버전으로 에디터를 업데이트하는 것을 추천하는 바이다.

[표 9-1] ezEditor의 알려진 주요 취약점

CVE ID	취약점 유형	영향을 받는 버전
-	파일업로드	모든 버전

9.2. 취약점 및 대응방안

9.2.1. 파일 업로드 취약점 (CVE-2009-2324)

9.2.1.1. 개요

웹 서비스 첨부 파일, 환경 설정 미흡 등을 악용하여 악의적인 스크립트가 포함된 파일을 업로드 한 후에 웹 서버에 침투를 할 수 있는 취약점이다. 공격자는 서버 전용 스크립트(PHP, JSP, .NET)를 이용하여 웹셸(WebSehll)을 제작하고 이를 게시판 첨부파일에 업로드 할 수 있다.

[표 9-2] 웹서버 별 실행가능 확장자

웹서버 환경	실행가능 확장자
IIS	asp, aspx, cgi, cer
PHP	php, php3, cgi, cer
JAVA	jsp, cer

9.2.1.2. 영향 받는 버전

ezEditor 모든 버전(0.1.1 ~ 0.2.5)

9.2.1.3. 해결방안

A. 파일 확장자 검증

첨부된 파일의 확장자를 검증하는 코드를 추가하여 공격자에 의해 악성파일이 업로드 되는 것을 사전에 예방할 수 있다. 검증하는 방식은 "BlackList 차단" 방식보다 "WhiteList 허용" 방식이 더 효과적이다.

[EzEditor 9-1] upload.js 취약한 소스코드(검증코드 삽입 전)

```
1  $.ajax({
2          url: self.settings.action,
3          type: 'post',
4          processData: false,
5          contentType: false,
6          data: form,
7          xhr: optionXhr,
8          context: this,
9          success: self.settings.success,
10         error: self.settings.error
11     })
12     return this
13   }
14   else {
15     // iframe upload
16     self.iframe = newIframe()
17     self.form.attr('target', self.iframe.attr('name'))
18     $('body').append(self.iframe)
19
20     self.iframe.one('load', function() {
21 $('<iframe src="javascript:false"></iframe>')
22           .appendTo(self.form)
23           .remove()
24       var response = $(this).contents().find('body').html()
25       $(this).remove()
26       if (!response) {
27         if (self.settings.error) {
```

[EzEditor 9-2] upload.js 취약한 소스코드(검증코드 삽입 후)

```
1  $.ajax({
2          url: self.settings.action,
3          type: 'post',
4          processData: false,
5          contentType: false,
6          data: form,
7          xhr: optionXhr,
8          context: this,
9          success: self.settings.success,
10         error: self.settings.error
11     })
12     return this
13   }
14   else {
15     // iframe upload
16     self.iframe = newIframe()
17     self.form.attr('target', self.iframe.attr('name'))
18     $('body').append(self.iframe)
19 $("input[id=nt_img_real").change(function(){
20     if($(this).val() != ""){
21         // 확장자 체크
22             var ext = $(this).val().split(".").pop().toLowerCase();
23             if($.inArray(ext, ["gif","jpg","jpeg",,"png"]) == -1){
24                 alert("gif, jpg, jpeg, png 파일만 업로드 해주세요.");
25                 $(this).val("");
26                 return;
27             }
```

```
                self.settings.error(self.input.val())
            }
        }
        else {
            if (self.settings.success) {
                self.settings.success(response)
            }
        }
    })
    self.form.submit()
  }
  return this
}
```

```
    self.iframe.one('load', function() {
$('<iframe src="javascript:false"></iframe>')
            .appendTo(self.form)
            .remove()
        var response = $(this).contents().find('body').html()
        $(this).remove()
        if (!response) {
            if (self.settings.error) {
                self.settings.error(self.input.val())
            }
        }
        else {
            if (self.settings.success) {
                self.settings.success(response)
            }
        }
    })
    self.form.submit()
  }
  return this
}
```

10. 마무리

지금까지 국내에서 가장 많이 사용되고 악용 중인 웹에디터 프로그램 별 취약점 및 대응방안에 대해 알아보았다. 위에서 언급한 취약한 웹에디터 프로그램을 사용 중인 홈페이지의 개발자나 담당자는 취약점을 보완하는 검증코드를 자사에 적용하기 전 충분한 테스트와 이전 소스코드의 백업 등으로 만약에 발생할 지 모르는 운영 장애에 대해 반드시 대비가 필요함을 잊지 말기를 바라며 가이드를 마무리 한다.