디지털산업정보학회 논문지 제16권 제2호-2020년 6월
http://dx.doi.org/10.17662/ksdim.2020.16.2.053

악성 파일 업로드 공격 대응방안 연구

김 태 경*

Research on countermeasures against malicious file upload attacks

Kim Taekyung

〈Abstract〉

Malicious file upload attacks mean that the attacker to upload or transfer files of dangerous types that can be automatically processed within the web server's environment. Uploaded file content can include exploits, malware and malicious scripts. An attacker can user malicious content to manipulate the application behavior. As a method of detecting a malicious file upload attack, it is generally used to find a file type by detecting a file extension or a signature of the file. However, this type of file type detection has the disadvantage that it can not detect files that are not encoded with a specific program, such as PHP files.

Therefore, in this paper, research was conducted on how to detect and block any program by using essential commands or variable names used in the corresponding program when writing a specific program. The performance evaluation results show that it detected specific files effectively using the suggested method.

Key Words : Malicious File Upload Attack, Detection, Signature, Countermeasure

Ⅰ. 서론

악성 파일 업로드 공격은 웹 파일 업로드 기능이 존재하는 웹 애플리케이션에서 확장자 필터링이 제대로 이루어지지 않았을 경우 공격자가 악성 스크립트 파일(웹쉘)을 업로드 하여 웹을 통해 해당 시스템을 제어할 수 있어 명령어 실행 및 디렉터리 열람이 가능하고 웹 페이지 또한 변조가 가능한 취약점이다[1]. 악성 파일 업로드 공격은 직접적으로 파일을 업로드 하는 경우에도 발생할 수 있지만, RFI(Remote File Inclusion) 공격을 수행하는 경우에도 악성 파일을 업로드 하여 공격을 수행할 수 있다. 이러한 악성 코드를 이용한 공격은 계속적으로 증가하는 추세에 있다[2].

악성 파일 업로드를 이용한 공격의 일반적인 대응 방법으로는 첫 번째로 허용 가능한 파일의 확장자만 업로드 되도록 화이트리스트 기반 확장자 검사를 수행하는 것이 있으며[3], 두 번째로는 악성 파일이 업로드 되어도 실행을 할 수 없도록 웹서버의 설정을 통해 파일이 업로드되는 디렉터리에 스크립트 파일의 실행 권한을 제거하는 것이다. 마지막으로 악성

* 명지전문대학 인터넷응용보안공학과 교수

파일이 업로드 되었다고 하더라도 공격자가 웹을 통해 해당 파일에 접근할 수 없도록 디렉터리를 별도로 지정하는 방식이 있다[4].

본 논문에서는 악성 파일 업로드 공격 대응방법의 첫 번째인 허용 가능한 파일의 확장자만 업로드 되도록 하는 방식에서 공격자가 파일의 확장자를 변경하거나 프락시를 이용해 파일 정보를 변경해 필터링을 우회하는 방식에 대한 대응방법에 관한 연구를 수행하였다.

본 논문의 2장에서는 기존에 악성 파일 업로드 공격 방식인 MIME 타입 조작을 통한 악성 파일 업로드 방법과 RFI 공격을 통한 악성 파일 업로드에 대해서 분석을 수행하였으며, 3장에서는 파일의 특정 패턴을 이용하여 웹 서버에서 실행 가능한 파일의 탐지할 수 있는 방안의 제시 및 성능 평가를 수행하였다. 마지막으로 4장에서는 본 연구의 결론으로 구성하였다.

Ⅱ. 관련 연구

2.1 MIME 타입 조작을 통한 악성 파일 업로드

일반적으로 웹 서버 환경별로 실행 가능한 확장자로는 IIS 웹 서버 환경에서는 html, htm, asp, aspx, cgi, cer 등이 실행 가능하며, PHP 웹 서버 환경에서는 html, htm, php, php3, cgi, cer 등이 실행이 가능하다. 이외에도 JAVA 웹 서버 환경에서는 html, htm, jsp, cer 등이 실행이 가능하다. 따라서 일반적으로 웹 서버에서는 시큐어 코딩 또는 보안 설정을 통해 이러한 파일들이 업로드되지 않도록 설정하고 있다.

이러한 파일들이 업로드 되지 않도록 설정하는 방법중의 하나는 첨부된 파일의 MIME 타입을 검사하여 필터링하는 방법이 있다. 다음의 <그림 1>은 HTTP 프로토콜에서 info.php 파일을 업로드 하는 것을 BurpSuite Proxy 프로그램을 이용하여 캡처한 것이다[5].

```
Content-Disposition: form-data; name="uploaded"; filename="info.php"
Content-Type: application/x-php

<?php
  phpinfo();
?>
```

<그림 1> PHP 파일 MIME 타입

따라서 첨부된 파일의 MIME 타입을 검사하여 'Content-Type: application/x-php'인 경우 필터링을 수행하여 실행 가능한 파일의 업로드를 차단할 수 있다. 그러나 Proxy 프로그램을 이용하면 MIME 타입을 변경하여 필터링을 우회할 수 있다. 다음의 <그림 2>는 BurpSuite 프로그램을 이용하여 MIME 타입을 'Content-Type: image/jpeg'로 변경한 것이다.

```
Content-Disposition: form-data; name="uploaded"; filename="info.php"
Content-Type: image/jpeg

<?php
  phpinfo();
?>
```

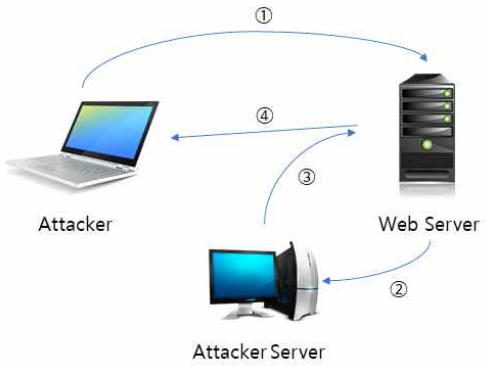
<그림 2> PHP 파일 MIME 타입 변경

MIME 타입을 허용하는 타입으로 변경하게 되면 MIME 타입을 통해 필터링하는 경우 악성 파일 차단을 우회하여 악성 파일을 업로드 할 수 있게 된다.

2.2 RFI 공격을 통한 악성 파일 업로드

RFI(Remote File Inclusion) 공격[6]은 파라미터를 통해 경로를 입력받아 사전에 의도한 파일이나 외부 모듈을 참조하는 웹 애플리케이션이 잘못된 경로 입력을 받아 권한이 허가되지 않은 파일의 내용을 노출하거나 외부의 악성 스크립트를 실행하게 되는 취약점이다.

RFI 공격의 개요는 <그림 3>과 같다. 일반적으로 URL에서 파라미터가 존재하는 경우 URI 다음에 '?'가 나오고 그 뒤에 파라미터가 오게 된다. 여기서 파라미터의 명이 'path'라고 하면, Attack Server의 웹



<그림 3> RFI 공격 개요

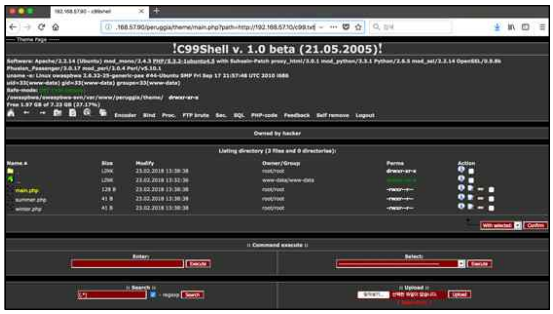
서버 홈 디렉터리에 웹쉘인 c99.php를 위치해 놓고 'http://웹서버?path=http://Attacker Server/c99.php' 형식으로 공격을 수행하게 된다. 그러나 일반적으로 '파일명.php'의 형식의 파일들은 웹서버에서 실행 가능하므로, 보안시스템을 통하여 차단을 수행하고 있다. 이러한 환경에서 RFI 공격을 수행하기 위해서는 웹쉘인 c99.php를 c99.txt로 변경해 다음과 같이 공격을 수행할 수 있다.

① 공격자가 웹 서버에 접속 시 정상적인 URL의 파라미터의 값에 공격자의 파일이 포함되도록 변경한다. (ex, http://웹서버?path=http://Attacker Server/c99.txt)

②, ③ Attacker Server의 c99.txt 웹쉘을 Web Server로 로드하여 Web Server에서 실행한다.

④ 공격자는 원격에서 웹쉘을 이용하여 Web Server에 공격을 수행한다.

<그림 4>는 Kali linux를 Attack Server로 사용하고, Cent OS 5.3을 Web Server로 사용하여 c99 웹쉘 [7]을 이용하여 원격에서 RFI 공격의 수행을 실행한 화면을 나타낸 것이다.



<그림 4> RFI 공격 수행 화면

또한, 블랙리스트 기반 악성 파일 업로드 필터링을 수행하는 경우 'aSP', 'AsP', 'pHp'와 같은 대소문자 조합을 사용하여 파일 이름을 지정하거나, 'php5', 'shtml', 'asa', 'cert'와 같은 서버가 실행할만한 파일의 확장자를 이용하여 공격을 수행할 수도 있다.

따라서 악성 파일 업로드 공격을 수행하기 위해 기존의 탐지 및 차단을 우회하여 공격하는 방법들에 대한 대응책을 수립하기 위해 본 연구를 수행하였으며, 실행 가능한 파일의 업로드 공격을 탐지하기 위한 방안은 3장에서 제시하였다.

Ⅲ. 시그니처 기반의 파일 확장자 필터링 방식

파일 유형을 식별하는 방법으로 가장 일반적인 방법은 파일 확장자를 활용하는 것이다. 그러나 파일의 확장자 및 MIME 타입 유형은 임의로 변경할 수 있으므로 파일의 유형을 탐지하는 방법을 우회할 수 있다. 따라서 파일의 확장자로 파일의 유형을 식별하는 것이 아니라 파일 고유의 데이터를 확인하여 파일의 종류를 식별하는 시그니처 분석을 통해 파일의 유형을 판별하여 접근제어를 수행할 수 있다.

3.1 제안 모델

모든 파일은 파일 고유 포맷의 특정 바이트로 이루어져 있는 파일 시그니처[8]가 존재하고, 일반적으로 '매직 넘버'라고도 한다. 보통은 파일의 첫 바이트로부터 일정 부분만큼을 파일의 시그니처로 지정하기도 하고 마지막 부분을 지정하기도 한다. 이를 헤더 시그니처와 푸터 시그니처라고 한다. 제안하는 모델에서는 이 시그니처를 이용하여 파일의 종류를 탐지하고 차단하는 방법에 대해서 제시하고자 한다.

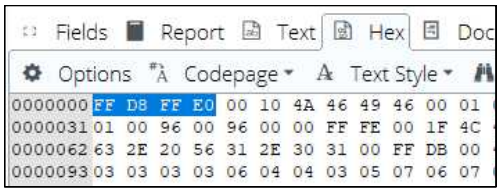
대표적인 파일들의 시그니처는 다음의 <표 1>과 같다.

<표 1> 파일 시그니처

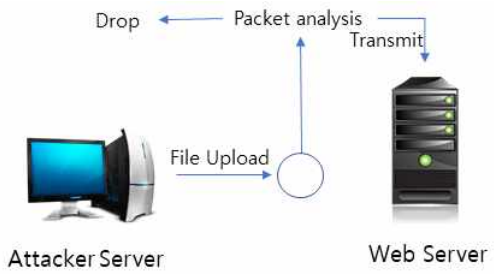
유형	확장자	16진수 데이터	Ascii 문자열
Bitmap	.bmp	42 4D	BM
GIF	.gif	47 49 46 38	GIF8
JPEG	.jpg	FF D8 FF E0	....
PNG	.png	89 50 4E 47	.PNG
Pkzip	.zip	50 4B 03 04	PK..
PDF	.pdf	25 50 44 46	%PDF
MS Office HWP	.doc .xls .ppt .hwp	D0 CF 11 E0 A1 B1 1A E1	...
MS Office	.docx .xlsx .pptx	50 4B 03 04	PK

예를 들면, JPEG 그림 파일을 Encase v8.08 버전으로 16진수의 값으로 해당 내용을 확인해 보면 파일의 시작 부분이 'FF D8 FF E0'로 시작하는 것을 알 수 있다.

파일의 시그니처는 디지털 포렌식에서 파일의 확장자를 변경하는 경우에 원래 어떠한 파일이었는지를 파악하는 데 도움을 줄 수 있다. 즉 시그니처를 분석함으로 어떠한 프로그램을 통하여 해당 파일이 인코딩 되었는지를 파악할 수 있다. 그러나 파일의 시그니처를 이용한 방식에서는 특정한 프로그램으로 인코딩한 경우에만 탐지할 수 있으며, 텍스트 파일 형식으로 작성된 PHP 파일의 경우 일반적인 시그니처 탐지 방법으로는 탐지할 수 없다는 단점을 가지고 있다. 따라서 본 논문에서는 해당 패킷을 분석하여 패킷의 내용을 통해 실행 가능한 파일들을 차단하는 방법을 제시하였다. 제안하는 모델은 다음의 <그림 6>과 같다.



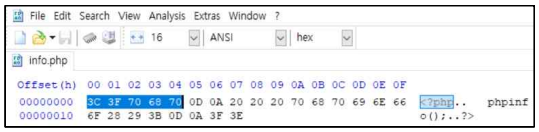
<그림 5> JPEG 파일 시그니처



<그림 6> 악성 파일 업로드 공격 대응 모델

공격자가 악성코드를 MIME 타입이나 파일 확장자를 변경하여 업로드 공격을 수행하는 경우에 이를 탐지하기 위해서는 파일에 대한 정보를 가지고 있는 메타정보를 이용해서는 효과적으로 악성코드들의 업로드를 차단할 수 없다. 이는 프락시 프로그램이나 파일 확장자의 변경을 통해서 해당 탐지를 우회할 수 있기 때문이다. 따라서 본 논문에서는 업로드되는 파일에서 특정 패턴을 탐색하여 파일에 대한 정보를 찾는 방법으로 악성코드 업로드 공격에 대한 대응방안을 제시하고자 한다.

2장에서 제시한 c99 웹쉘을 차단하기 위한 방안은 다음과 같다. PHP 파일을 작성할 때에는 기본적으로 '<?php'와 '?>'가 파일 안에 존재해야 한다. 즉 각각의 실행 가능한 파일에서는 일반적으로 사용하지 않는 특정한 패턴이 있으므로 이러한 패턴을 탐지하여 악성 파일을 탐지할 수 있다. 다음의 <그림 7>은 PHP 파일을 HxD를 이용해 확인한 것이다. '<?php'에 해당하는 16진수의 값은 '3C 3F 70 68 70', '?>'의 16진수의 값은 '3F 3E'인 것을 알 수 있다.



<그림 7> PHP 파일 시그니처

따라서, 파일의 업로드 시 패킷의 데이터를 검사하여 서버에서 실행 가능한 악성 파일을 탐지할 경우 패킷을 차단함으로 악성 파일 업로드 공격에 대응할 수 있다.

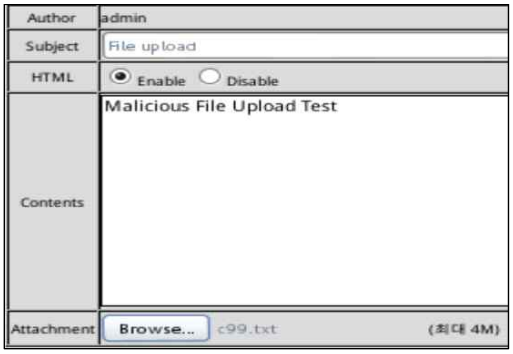
3.2 성능 평가

제안한 모델의 성능 평가를 수행하기 위하여 Kali 리눅스로 공격자의 역할을 수행하였고, Cent OS 5.3(IP 주소: 192.168.57.10)으로 웹서버(IP 주소: 192.168.57.80)를 구축하였으며 두 시스템 사이에 Cent OS 6.4로 snort(IP 주소: 192.168.57.2)를 설치하여 시뮬레이션을 수행하였다.

공격은 PHP 파일의 확장자를 변경하여 웹 서버에 올리는 것으로 하였으며, 해당 패킷을 탐지하여 업로드를 차단하는 실험을 수행하였다. 웹서버에 게시판을 구현하여 c99.php 파일의 확장자를 c99.txt로 변경하여 업로드 하였다. <그림 9>는 웹서버 게시판에 악성 파일을 업로드 한 화면이다.

스노트 시스템[9]에서 룰의 설정은 <그림 10>과 같

<그림 8> 시뮬레이션 환경



<그림 9> 악성 파일 업로드

이 설정을 하였다. 이 룰에서 content는 전송되는 데이터를 검색해서 지정한 문자열이 있는지 확인하는 옵션이며, 16진수로 검색할 대상을 지정하려면 "|"로 검색대상 앞뒤로 설정해야 한다.

```
alert tcp any any -> any any (content:"|3C 3F 70 68 70|";
msg:"PHP File Upload Detection";sid:1000001;gid:1;)
```

<그림 10> 스노트 룰 설정

다음의 <그림 11>은 스노트 룰에 의해 악성 파일이 Kali 리눅스에서 웹 서버로 업로드 될 때 탐지된 로그 화면을 나타낸 것이다.

```
[**] [1:1000001:0] PHP File Upload Detection [**]
[Priority: 0]
06/06-18:22:08.968938 192.168.57.10:41961 -> 192.168.57.80:80
TCP TTL:64 TOS:0x0 ID:9585 IpLen:20 DgmLen:244 DF
***AP*** Seq: 0x490618D  Ack: 0xA14AC694  Win: 0x1D  TcpLen: 32
TCP Options (3) => NOP NOP TS: 4294910167 1657720
```

<그림 11> 스노트 로그

특정한 프로그램에 의해 인코딩 된 파일의 경우에는 해당 파일에 있는 특정한 유형의 파일 시그니처를 이용하여 특정한 유형의 파일들을 탐지할 수 있다. 그러나 PHP 파일과 같은 경우에는 파일 시그니처가 존재하지 않으므로 해당 프로그램 파일에 존재하는 특정한 유형의 명령어나 변수명을 이용하여 어떠한 파일인지에 대한 탐지를 수행할 수 있다.

Ⅳ. 결론

악성 파일 업로드 공격은 계속해서 수행되고 있는 공격 중의 하나로 특히 웹서버에서 실행 가능한 파일들을 업로드 함으로 공격을 수행하고 있다. 이러한 공격을 탐지하는 방법으로는 일반적으로 파일의 확장자나 파일의 시그니처를 탐지하여 파일의 종류를 확인하는 방법이다. 그러나 이러한 방식의 파일 종류 탐지는 PHP 파일과 같이 특정 프로그램으로 인코딩 되지 않은 파일은 탐지할 수 없다는 단점을 가지고 있다.

따라서 본 논문에서는 특정 프로그램 작성 시 해당 프로그램에서 사용하고 있는 필수 명령어나 변수명들을 이용하여 어떠한 프로그램인지를 탐지하여 차단하는 방법에 관한 연구를 수행하였다. 성능 평가를 수행한 결과 효과적으로 특정 파일을 탐지하는 것으로 나타났다.

향후 연구계획으로는 기존의 파일 시그니처와 시그니처가 없는 파일들을 통합적으로 탐지하는 방법에 관한 연구를 수행할 계획이다.

참고문헌

[1] 김수정·하지희·오수현·이태진, “정적 분석 기반 기계학습 기법을 활용한 악성코드 식별 시스템 연구,” 정보보호학회논문지, 제29권, 제4호, 2019, pp.775-784.

[2] Cisco 2018 annual cybersecurity report, Cisco, 2018.

[3] 하정우·김휘강·임종인, “WhiteList 기반의 악성코드 행위분석을 통한 악성코드 은닉 웹사이트 탐지 방안 연구,” 정보보호학회논문지, 제21권, 제4호, 2011, pp.61-75.

[4] Zhi-Yong Li, Ran Tao, Zhen-He Cai, Hao Zhang. “A Web Page Malicious Code Detect Approach Based on Script Execution,” ICNC.09, Fifth International Conference, 2009, pp.308-312.

[5] 김광현, “웹 취약점 분석을 위한 프락시 시스템의 설계 및 구현,” 한국정보통신학회논문지, 제18권, 제9호, 2014, pp.1011-1018.

[6] 박재연·이송연·이하은·이종우, “리눅스 아파치 웹 서버 실시간 로그 분석을 통한 공격 탐지 프로그램 개발,” 정보과학회 컴퓨팅의 실제 논문지, 제24권, 제4호, 2018, pp.190-197.

[7] P. M. Wrench, B. V. W. Irwin, “Towards a PHP webshell taxonomy using deobfuscation-assisted similarity analysis,” 2015 Information Security for South Africa (ISSA), Johannesburg, 2015, pp.1-8.

[8] 이인복·오성일, “시그니처 기반 침입 탐지 시스템의 최적화,” 정보과학회지, 제33권, 제6호, 2015, pp.55-60.

[9] Snort. http://www.snort.org.

■ 저자소개 ■

김 태 경
Kim, TaeKyung

2017년 9월~현재
명지전문대학
인터넷응용보안공학과 교수
2008년 3월~2017년 8월
서울신학대학교 교수
2006년 3월~2008년 2월
서일대학 정보전자과 교수
2005년 8월 성균관대학교
전기전자및컴퓨터공학과(공학박사)

관심분야 : 네트워크보안, IoT 보안,
개인정보보호, 디지털포렌식
E-mail : tkkim@mjc.ac.kr

논문접수일 : 2020년 6월 8일
게재확정일 : 2020년 6월 17일