**XXE(XML 实体注入)漏洞攻防分析**

**Auth:Cryin’**

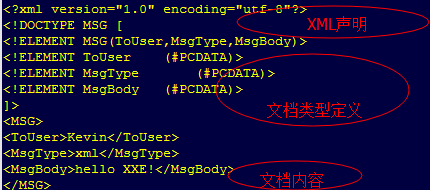
**Date:2015/10/21**

**概述**

XML可扩展标记语言 (Extensible Markup Language, XML) ，用于标记电子文件使其具有结构性的标记语言，可以用来标记数据、定义数据类型，是一种允许用户对自己的标记语言进行定义的源语言。 XML是标准通用标记语言 (SGML) 的子集，非常适合 Web 传输。XML 提供统一的方法来描述和交换独立于应用程序或供应商的结构化数据。安全研究者发现一些普遍使用XML的场景中都存在一种XML实体注入漏洞(即XML External Entity Injectio简称XXE漏洞)，当允许引用外部实体时，攻击者通过构造恶意内容，可导致读取任意文件、执行系统命令、探测内网端口、攻击内网网站等危害。

**XML格式**

XML文档结构包括XML声明、DTD文档类型定义（可选）、文档元素。



文档类型定义(DTD)的作用是定义 XML 文档的合法构建模块。DTD 可以在 XML 文档内声明，也可以外部引用。

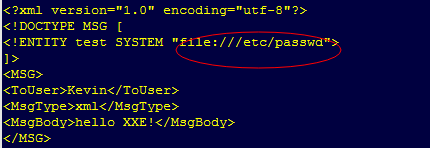
内部声明DTD:

<!DOCTYPE 根元素 [元素声明]>

引用外部DTD:

<!DOCTYPE 根元素 SYSTEM "文件名">

当允许引用外部实体时，恶意攻击者即可构造恶意内容访问服务器资源,如读取passwd文件：



**典型漏洞案例**

XML 实体注入漏洞从原理上讲，只要程序允许引用外部实体，攻击者通过构造恶意内容，就可能导致读取任意文件、执行系统命令、探测内网端口、攻击内网等攻击。所以XXE和xss，sqli 等一样都是比较广泛的漏洞，所以从以往发现的漏洞可以看到很多厂商都存在这种漏洞，包括邮箱、门户、通用CMS等，如

网易、腾讯邮箱XXE可读取文件

<http://www.wooyun.org/bugs/wooyun-2010-073321>

<http://www.wooyun.org/bugs/wooyun-2014-073439>

百度XXE实体注入

<http://www.wooyun.org/bugs/wooyun-2010-058381>

DZ xxe 可破坏数据库结构，导致脏数据进入

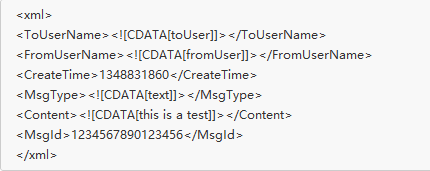
<http://www.wooyun.org/bugs/wooyun-2014-076041>

Oracle盲注结合XXE漏洞远程获取数据

<http://drops.wooyun.org/papers/6035>

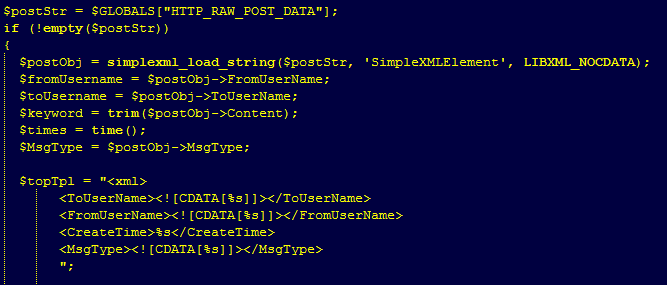
**漏洞分析**

随着微信公众号的广泛使用，一些通用CMS程序都利用微信的公共接口增加了与微信用户交互的插件或者独立的功能模块。而这些公共的接口正是通过XML数据包进行数据传输。见<http://mp.weixin.qq.com/wiki/10/79502792eef98d6e0c6e1739da387346.html>



就是说这些通用的CMS通过引入微信公众平台这个XML数据结构传递相应的消息及事件。如果程序对于XML数据包未经过确认是否来自与微信服务器并允许引入外部实体，那攻击者就可以构造恶意内容，并获取服务器更多资源。

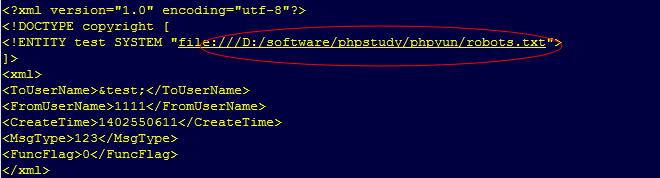
这里以PHPYUN(PHP云人才招聘系统)为例，源文件代码weixin/model/index.class.php中相应代码调用微信开发者接口获取数据。



获取POST过来的数据通过simplexml\_load\_string解析后的内容未经任何处理，直接带入了$topTpl，并在最后将数据输出。

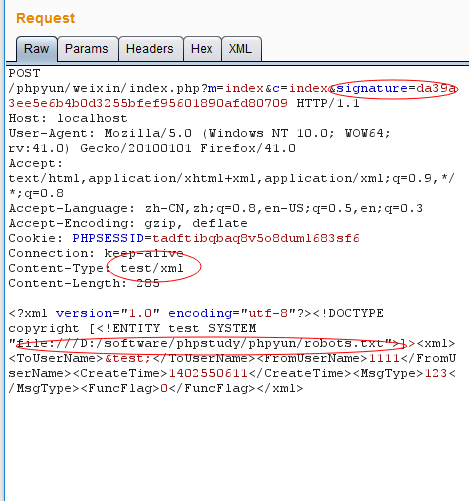


在整个过程中都将POST得到的XML数据作为可信数据，并直接使用。所以我们尝试构造一个读取服务器文件的XML即可验证是否存在XML实体注入漏洞！构造的文件如下图:

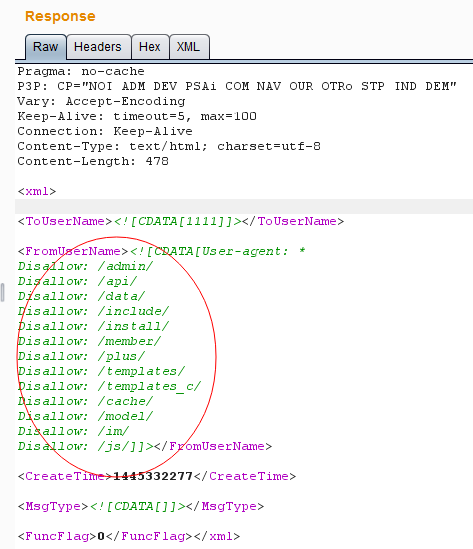


发送上述POST请求并查看返回内容，如图:

POST请求截图:

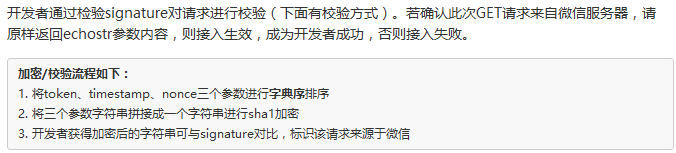


返回消息截图:



可以看到已经成功读取到了robots.txt文件的内容。说明该程序存在XXE漏洞。

该漏洞的成因在于PHPYUN 在请求XML数据时并未对来源身份进行验证(虽有进行校验但token值存在缺省值并已泄露)，在后续数据处理时也 未对数据进行任何校验和过滤。我们查看[微信开发者接口的指导文档](http://mp.weixin.qq.com/wiki/17/2d4265491f12608cd170a95559800f2d.html)中对于接入的说明:



所以，在新版的PHPYUN相关源代码中可以看到如下代码:



即对外部实体身份进行校验，确认为微信服务器时才使用该post请求得到的XML数据。

**XXE防御措施**

因为不同的程序支持的协议不同,比如java、php、python等，所以防御措施也有不同，不过总体可以将防御措施分为两类：

(1)使用开发语言提供的禁用外部实体的方法

PHP：

libxml\_disable\_entity\_loader(true);

JAVA:

DocumentBuilderFactory dbf =DocumentBuilderFactory.newInstance();

dbf.setExpandEntityReferences(false);

Python：

from lxml import etree

xmlData = etree.parse(xmlSource,etree.XMLParser(resolve\_entities=False))

(2)校验XML数据请求身份或过滤用户提交的XML数据

关键词：<!DOCTYPE和<!ENTITY，或者，SYSTEM和PUBLIC等。

**结束语**

XML 实体注入漏洞从原理上讲，只要程序处理XML文件允许引用外部实体，攻击者通过构造恶意内容，就可能导致读取任意文件、执行系统命令、探测内网端口、攻击内网等攻击。所以XXE和xss，sqli 等一样都是比较广泛的漏洞，甚至在一些客户端程序中也有该漏洞，所以开发人员在处理XML时需谨慎，在用户可控的XML数据里禁止引用外部实体，同时有必要的话需要对XML数据身份进行确认并过滤外部实体包含的恶意关键词。

**参考**

[1] <http://www.80sec.com/xml-entity-injection.html>

[2] <http://security.tencent.com/index.php/blog/msg/69>

[3] <http://www.wooyun.org/bugs/wooyun-2010-064637>

[4] <http://defcon.org.ua/data/2/2_Vorontsov_XXE.pdf>

[5] <https://media.blackhat.com/eu-13/briefings/Osipov/bh-eu-13-XML-data-osipov-wp.pdf>