Intro to XSLT Injection

eXtensible Stylesheet Language Transformation (XSLT) is a language enabling the transformation of XML documents. For instance, it can select specific nodes from an XML document and change the XML structure.

eXtensible Stylesheet Language Transformation (XSLT)

Since XSLT operates on XML-based data, we will consider the following sample XML document to explore how XSLT operates:

Code: xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<fruits>
    <fruit>
        <name>Apple</name>
        <color>Red</color>
        <size>Medium</size>
    </fruit>
    <fruit>
        <name>Banana</name>
        <color>Yellow</color>
        <size>Medium</size>
    </fruit>
    <fruit>
        <name>Strawberry</name>
        <color>Red</color>
        <size>Small</size>
    </fruit>
</fruits>
```

XSLT can be used to define a data format which is subsequently enriched with data from the XML document. XSLT data is structured similarly to XML. However, it contains XSL elements within nodes prefixed with the `xsl`-prefix. The following are some commonly used XSL elements:

- `<xsl:template>`: This element indicates an XSL template. It can contain a `match` attribute that contains a path in the XML document that the template applies to
- `<xsl:value-of>`: This element extracts the value of the XML node specified in the `select` attribute
- `<xsl:for-each>`: This element enables looping over all XML nodes specified in the `select` attribute

For instance, a simple XSLT document used to output all fruits contained within the XML document as well as their color, may look like this:

Code: xslt

```xslt
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/fruits">
        Here are all the fruits:
        <xsl:for-each select="fruit">
            <xsl:value-of select="name"/> (<xsl:value-of select="color"/>)
        </xsl:for-each>
    </xsl:template>
</xsl:stylesheet>
```

As we can see, the XSLT document contains a single `<xsl:template>` XSL element that is applied to the `<fruits>` node in the XML document. The template consists of the static string `Here are all the fruits:` and a loop over all `<fruit>` nodes in the XML document. For each of these nodes, the values of the `<name>` and `<color>` nodes are printed using the `<xsl:value-of>` XSL element. Combining the sample XML document with the above XSLT data results in the following output:

```
Here are all the fruits:
    Apple (Red)
    Banana (Yellow)
    Strawberry (Red)
```

Here are some additional XSL elements that can be used to narrow down further or customize the data from an XML document:

- `<xsl:sort>`: This element specifies how to sort elements in a for loop in the `select` argument. Additionally, a sort order may be specified in the `order` argument
- `<xsl:if>`: This element can be used to test for conditions on a node. The condition is specified in the `test` argument.

For instance, we can use these XSL elements to create a list of all fruits that are of a medium size ordered by their color in descending order:

Code: xslt

```xslt
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/fruits">
        Here are all fruits of medium size ordered by their color:
        <xsl:for-each select="fruit">
            <xsl:sort select="color" order="descending" />
            <xsl:if test="size = 'Medium'">
                <xsl:value-of select="name"/> (<xsl:value-of select="color"/>)
            </xsl:if>
        </xsl:for-each>
    </xsl:template>
</xsl:stylesheet>
```

This results in the following data:

```
Here are all fruits of medium size ordered by their color:
        Banana (Yellow)
        Apple (Red)
```

XSLT can be used to generate arbitrary output strings. For instance, web applications may use it to embed data from XML documents within an HTML response.

XSLT Injection

As the name suggests, XSLT injection occurs whenever user input is inserted into XSL data before output generation by the XSLT processor. This enables an attacker to inject additional XSL elements into the XSL data, which the XSLT processor will execute during output generation.