The principle of XPath injection is very similar to SQL injection. The goal of the attack is very similar too. The only difference between these attacks is that XPath injection uses an XML file for data storage instead of a database. One way to get data from an XML file is to use a special query language called XPath.

Example of XPath injection

Let's say that a developer stores authentication data in an XML file with the following structure:

```
...
<user>
        <name>UserName</UserName>
        <password>Password</password>
</user>
...
```

On authentication, the developer builds an Xpath expression this way:

```
string//user[name/text()='"txtUserName.Text"' and password/text()='" txtPassword.
Text"'])
```

The txtUserName and txtPassword variables are standard ASPX textboxes. When the attacker inserts an expression with an apostrophe (') to one of the textboxes, the attacker terminates the string and is able to write his own XPath expression. The scenario is basically the same as with SQL injection.

What can XPath injection attack do

An attacker can get, modify or delete anything stored in the given XML file.

Finding XPath injection vulnerabilities

The first technique is based on trying. Try to insert strings like:

- 'whatever – basic test
- DROP
- Something

to all inputs/URL parameters/whatever. If you see any error related to classes which provide manipulation with XMLs in ASP.NET, you have probably found an XPath injection threat.

The second way is to search for vulnerabilities in code. You can search for the following strings:

- Xpath - many classes which work with XPath have the 'xpath' string in their name.
- **SelectSingleNode()** and **SelectNodes()** - methods used in Kentico for getting data from XML files via XPath.

Avoiding XPath injection in Kentico

We do not store sensitive data in XML files in Kentico at the moment. We also do not have any code where an XPath expression is built as described in the provided example. But this may change in the future. You can avoid XPath injection by following these rules:

- Validate input from external sources before you put it into XPath expressions.
- For characters like ', <, >, etc., use replace entities. "'" is a replace entity for an apostrophe.