security-assessment.com

XPath Injection Attacks

Of the Awesome Advanced Automated Kind

Paul Haas

Kiwicon 7

Agenda

security-assessment.com

- Who, What, Why, How, Where, When

Who

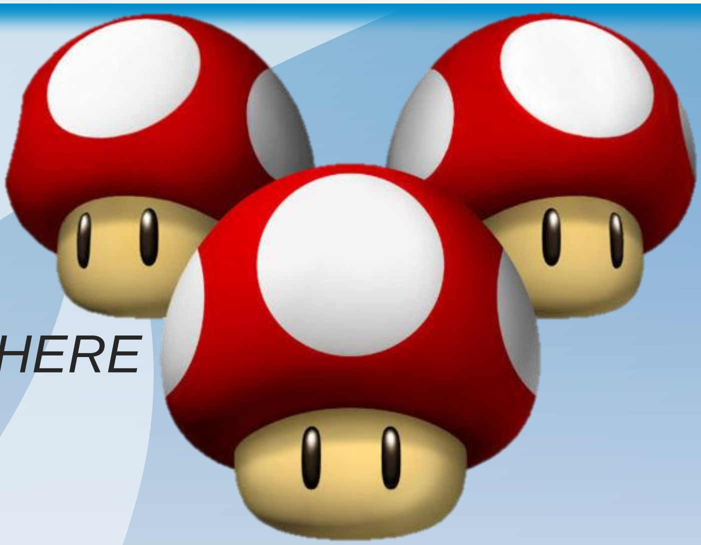
security-assessment.com

- **Paul ‘sss’ Haas**
  - Security Consultant @ Security-Assessment.com in Wellington
- **Experience**
  - 10 years in computer security, hailing from California, living in NZ
  - Expertise across entire pentest spectrum: App, Net, WIFI, DB, etc.
  - Talks: OWASP Day NZ 2013, sec. training classes, Defcon 2010
  - Bash-Fu Master, XPath Ninja, CTF Winner, Psychic Beach bum
- **Passions**
  - Solving complex problems (the hack)
    - *Alternately*: making them more complex
  - Mario Kart duals at sunset on the *beach*

What

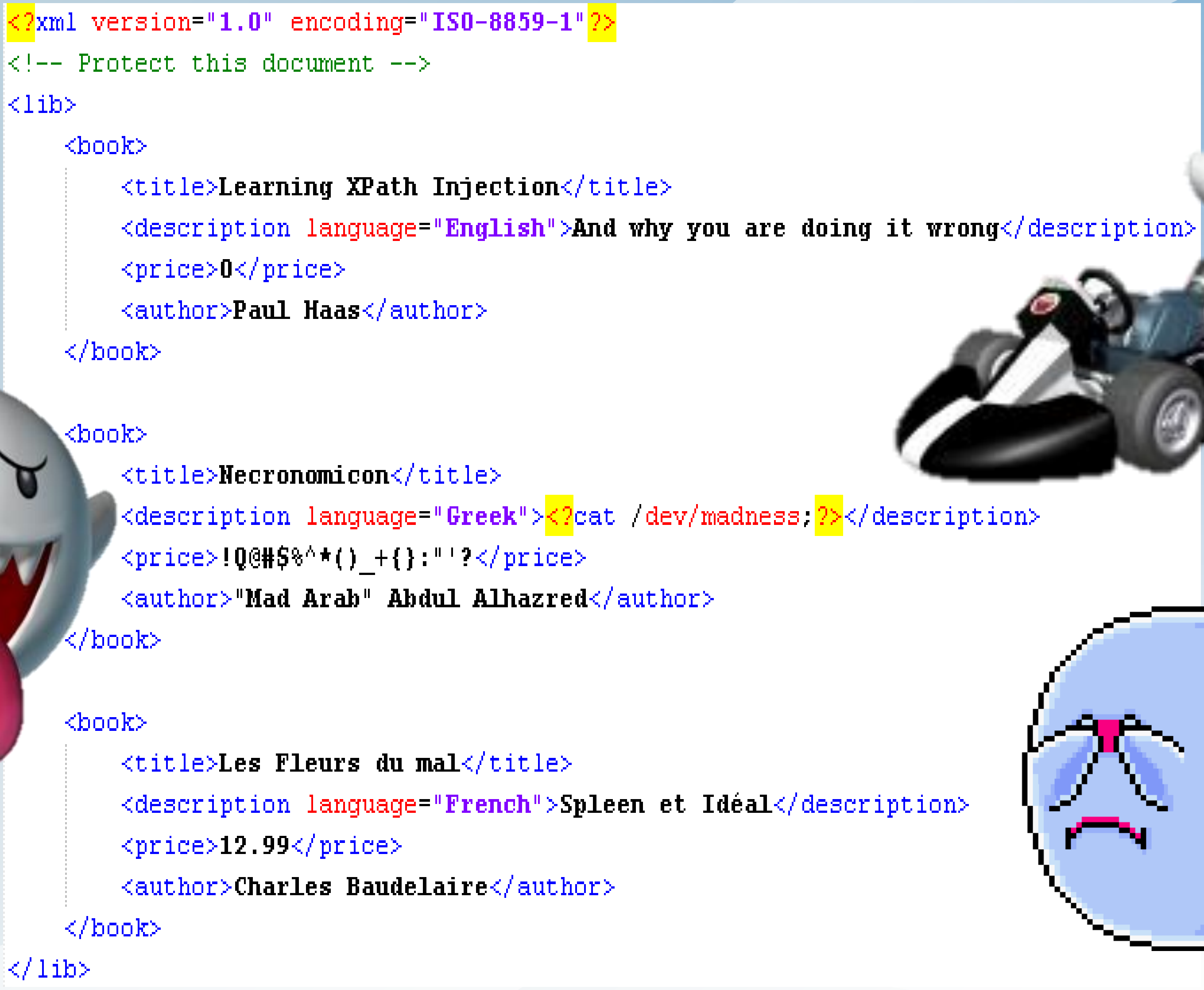
security-assessment.com

- **What is XPath?**
  - Like SQL but for XML Documents
    - SQL: *SELECT book FROM bookstore WHERE title='Test'*
    - XPATH: */library/book/[title='Test']*
  - Uses File System Folder/Path syntax with slashes '/'
    - *Parent, Ancestor, Sibling, Descendants, nodes*
  - Based on standards we don't really care about
    - W3C: XQuery, XLink, XSLT
    - Guaranteed universal implementation

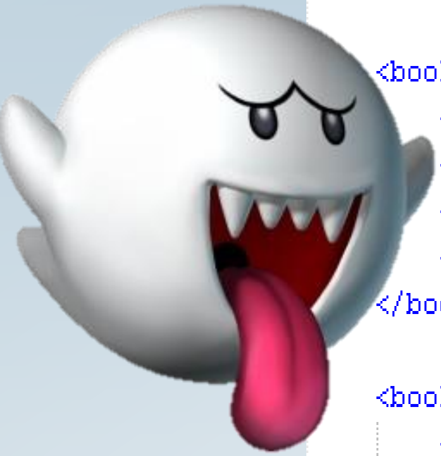


What

security-assessment.com

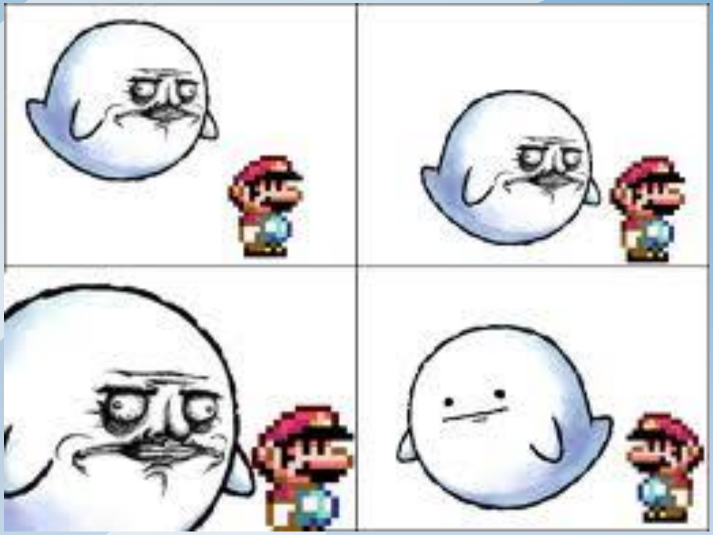


```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Protect this document -->
<lib>
    <book>
        <title>Learning XPath Injection</title>
        <description language="English">And why you are doing it wrong</description>
        <price>0</price>
        <author>Paul Haas</author>
    </book>

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

    <book>
        <title>Les Fleurs du mal</title>
        <description language="French">Spleen et Idéal</description>
        <price>12.99</price>
        <author>Charles Baudelaire</author>
    </book>
</lib>
```



What

security-assessment.com



- **Examples**
  - count(/lib/book)
  - /lib/book[1]/price
  - //book[last()]/description
  - /lib/book[title='Learning XPath']
- **Elements**
  - Node, @attribute, ‘//’ anywhere, ‘.’ Current, ‘..’ Parent, ‘*’ wildcard
- **Functions**
  - name, count, string-length, translate, concat, contains, substring
- **Operators**
  - +-/*, *div*, =, !=, <, <=, >, >=, [ ], *or*, *and*, *mod*, | as a union operator

Wut

security-assessment.com

- **XPath 1 introduced in 1999**
  - Built-in and included with most XML frameworks/libraries
  - All features should be present in any XPath implementation
- **XPath 2 'Working Draft' introduced in 2007**
  - Introduces powerful functions useful for hacking
  - Not common in wild or fully implemented in most libraries
- **XPath 3 in candidate status as of January 2013**
  - No known implementations

Why

security-assessment.com

- **Why XPath**
  - XPath allows queries to read from a ‘sensitive’ backend database
  - Used in variety of web frameworks as a replacement for SQL
  - Commonly used to provide dynamic user interaction/search
  - Certain characters can modify purpose and function of query
  - Modified query can access other part of database
    - Including arbitrary XPath functions
- **Risk**
  - XPath 1: Retrieve the entire database
  - XPath 2: Access remote files on the server
- **Why does this sound familiar**
  - What alarms are going off?

Why

security-assessment.com

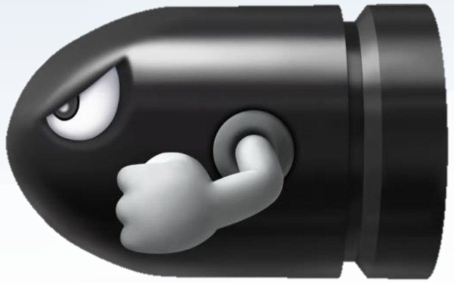
- **XPath Injection (XPi)**
  - Similar risk as SQL Injection
    - Much less awareness
  - Only a couple of tools
  - Plenty of vulnerable frameworks

- **Similar Injection Techniques**
  - If you know SQLi, you can do XPi
  - Single **'** and double **"** quotes escape strings
  - Spaces escape numerical input
  - Brackets **[ ]** used to escape XPath predicates
  - Error, union, time-based, blind techniques
  - Still works: ***x' OR '1'='1***
  - Even better: ***x' AND 1=0] | //*["1"="1***

Why

security-assessment.com

- **Penetration Testing**
  - Need to be aware of emerging technologies and vulnerabilities
  - XML technologies on the rise, more ‘enterprise’
  - Increased number of applications using XPath
  - Lack of techniques, tools and cheat sheets
- **Existing Work**
  - Various presentations and whitepapers about injection techniques
  - XPath-blind-explorer: Windows binary to perform blind injection
  - xcat.py: Blind XPath injection with focus on XPath 2 techniques
    - Both tools designed by same author for Blackhat



Why

security-assessment.com

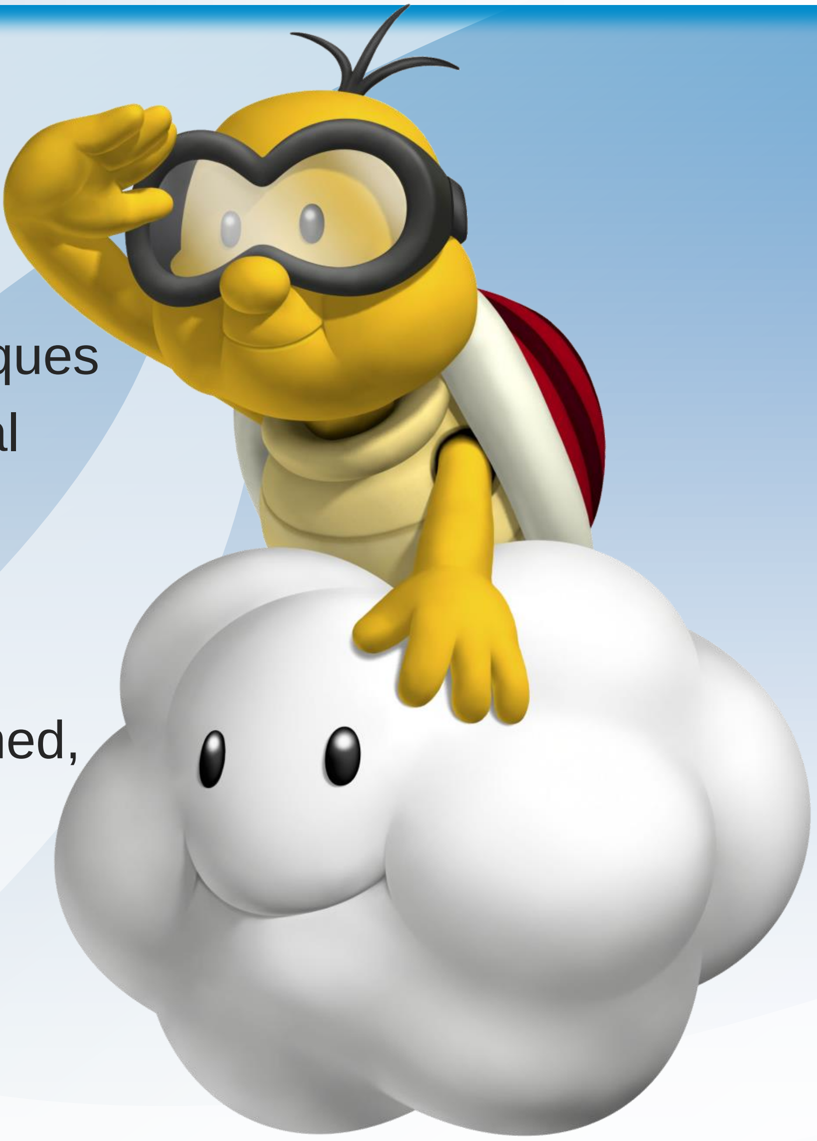
- **xcat Advantages**
  - Reconstruct a remote XML database using blind XPi
  - Replaces Windows binary with open source Python implementation
  - Includes both XPath 1 and 2 techniques
  - Uses threading and other optimization techniques

- **xcat Disadvantages**
  - Best optimizations only work in XPath 2
  - Version 1 falls back to slow linear methods
  - Threading makes improvements impossible
  - Cannot customize retrieval content

- **Can do better**

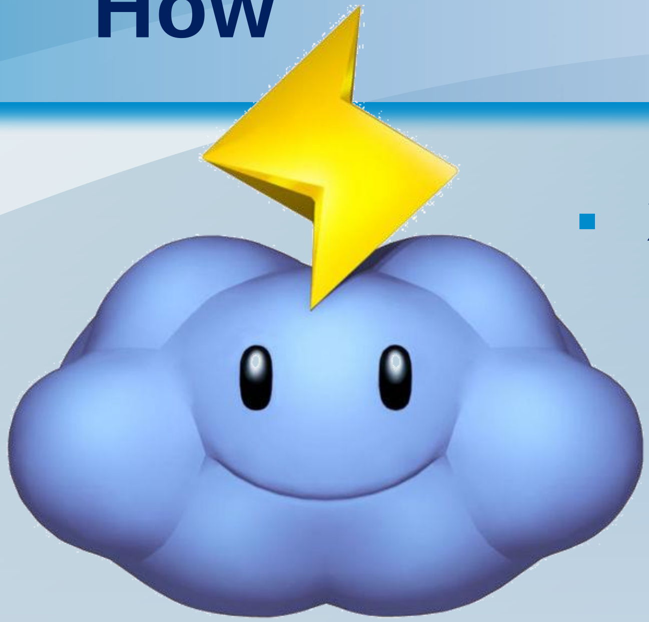
y?

- **Better Faster Stronger**
  - Use xcat as a starting point
  - Open source, allow future improvements
  - Focus solely on XPath 1 injection techniques
  - Use blind injection so method is universal
  - Allow customisation of retrieval content
- **Blind Injection**
  - Does not rely on XPath data being returned, errors or speed of response
  - Ask yes/no question about the database
  - Distinguish if true/false using response
  - Repeat until no questions remain



How

security-assessment.com



- **XPath Injection : A Brief Primer**
  - Find your own vulnerable application
  - Test all locations of dynamic input : GET, POST, HTTP Headers, cookies, etc.
  - Identify ‘SQL flaw’ using basic injection
  - Discover complex SQL injection isn’t working

- **Injection Comparisons**
  - **' OR '1'='1** – Supported in both SQLi and XPi
  - **' OR user() AND '1'='1** – Works in SQLi only
  - **' OR count(//*) AND '1'='1** – Works in XPi only
  - **' OR lower-case('A') AND '1'='1** – Works in XPath 2
  - **' OR kart() AND '1'='1** – Doesn’t work anywhere

How

security-assessment.com

- Demo

How

security-assess .com

Reconstructing an XML database using XPath

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

Tedious, hence the need to automate the attack

Needs to be further simplified for blind injection

How

security-assessment.com

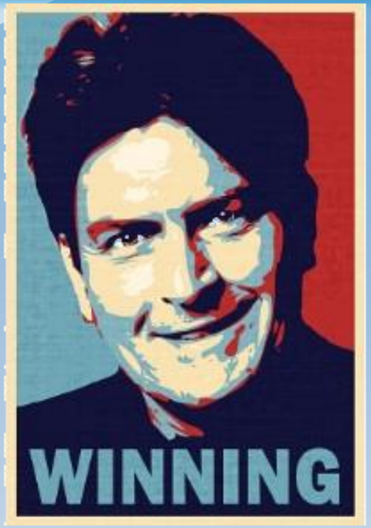
- **Blind Injection : “The question game”**
  - To recover a number, need to ‘guess’ using yes/no
  - For strings, ‘guess’ length & ‘guess’ each character
  - Must be repeated for everything in the database

- **xcat XPi Version 1 Blind Retrieval**
  - Guesses numbers by starting from **0** and going up
  - Guess characters by starting at ‘**a**’ and ending at ‘**Z**’
  - Only correct guess returns a valid injection result
  - Threaded to speedup slow guessing process
  - “You’re doing it wrong”

How

security-assessment.com

- **Search Techniques**
  - xcat uses a linear search method for blind retrieval
  - There are faster search algorithms, implement these
  - Determine if XPath 1 has necessary functionality



- **Binary Search**
  - Keeps dividing problem in half until single answer is found
  - IE: **Is character in the first or second half of the alphabet?**
  - Requests = *ln(size of alphabet)*, 8 requests for entire ASCII set

- **Numerical Binary Conversion**
  - Convert number to binary and check value of each bit individually
  - IE: **56 = 0b00111000, 8 requests to reconstruct numbers <256**

How

security-assessment.com

- **XPath has a minimal function set**
  - No direct method to determine if a character is present in a set
  - No method to convert a number to binary, or character to number
  - Recreate using SCIENCE

- **Binary Search**
  - Use **contains** function while dividing set in half until match
  - **contains([A..Z], character), contains([A..M], character), contains([A..G], character), contains([A..D], character), contains([A..B], character), character = 'A'**

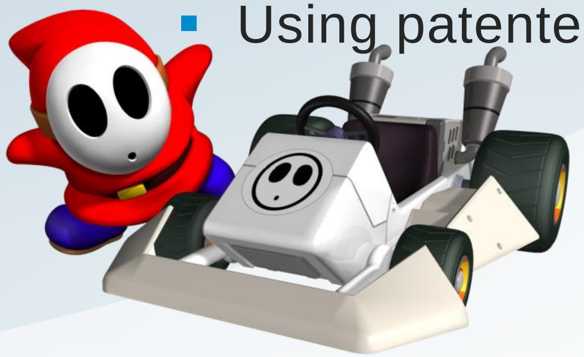
- **Numerical Binary Reconstruction**
  - Recreate bit shift/2binary using floor, division and modulus
  - **for n in range(0,8): bit[n] = floor(number div 2**n) mod 2**

How

security-assessment.com

- **Better search algorithms**
  - Adds code/query complexity
  - More difficult to thread
  - Need additional XPath 1 functions
    - *Not present in xcat*
  - ~6-8x speedup (logarithmic)

- **"BUT WAIT, there's more"**
  - There are additional tricks to speedup retrieval
  - To reach XPath 2 speeds with XPath 1 at no additional cost
  - Using patented backend logic and XPath black magic



How

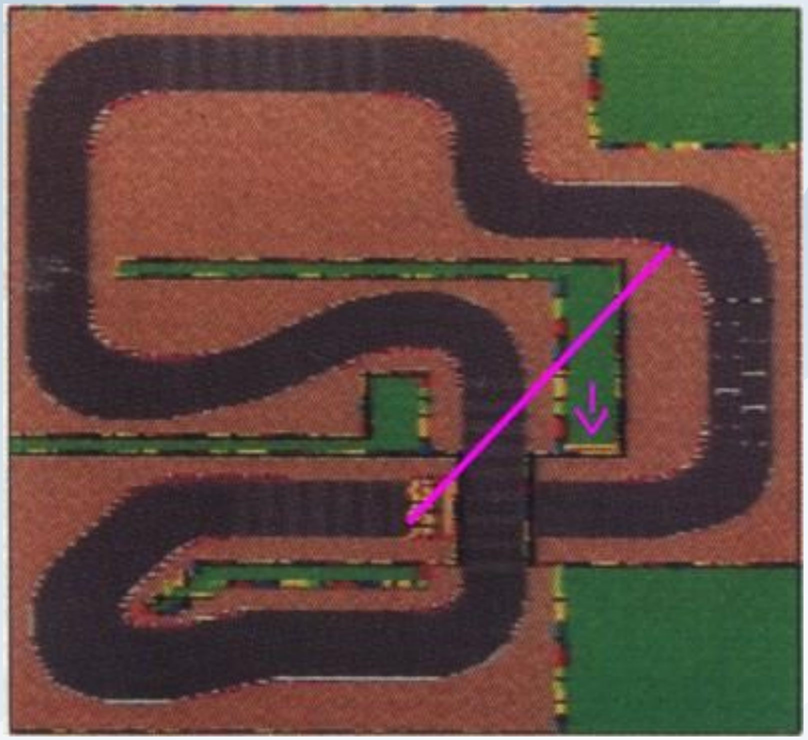
security-assessment.com

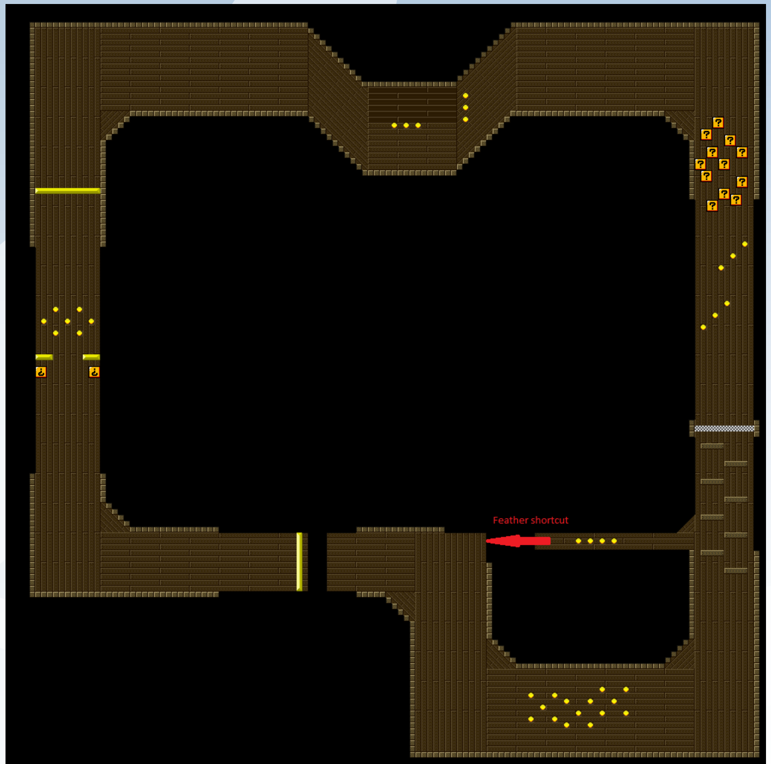
- **Improvement: Case Sensitive -> Insensitive Match**
  - xcat provides a lower-case match for XPath 2 only
  - Recreate XPath 2 lower-case() function in XPath 1
    - ***translate(character, [A-Z], [a-z])***
  - Slight improvement in number of XPath queries (<1%)
    - Only efficient for very large databases, not looking for passwords
    - Matching case after fact less efficient than just using Binary Search

How

security-assessment.com

- **Improvement: Normalize Whitespace**
  - Eliminate unnecessary whitespace before reconstruction
    - XPath 1: **normalize-whitespace(string)**
    - Eg: *[Space] [Space]** = *[Space]*
  - Significant improvement for 'text like' databases (<15%)





How

security-assessment.com

- **Improvement: Maintain Global Count**
  - Get initial count of each type of node, attribute, text, comment, processing instruction
    - **count(//*), count(//@*), count(//comment()), count(//text()), …**
  - Decrement count when accessing that type
  - Stop accessing that type when count is zero
  - Useful for top-heavy documents (comments only at top)
    - Slight speed improvement at small cost of initial requests (1-5%)
  - Very useful for documents missing a node type
    - 5-10% speed improvement for each missing type

How

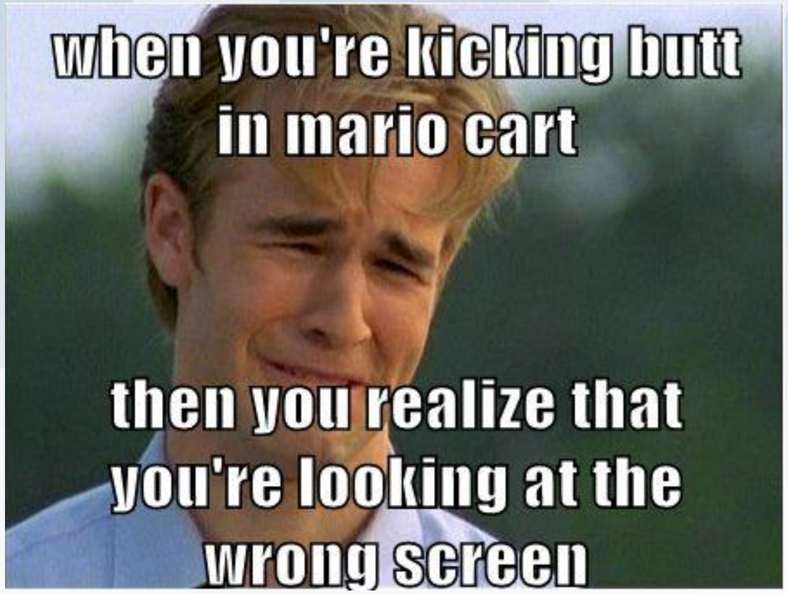
security-assessment.com

- **Improvement: Eliminate Non-Existent Characters**
  - Given set of all possible characters, determine if they are present anywhere in the database using a global search
    - **for c in [A..Z]: node_exists[c] = count(//*[contains(name(), *c*)])**
    - **for c in [A..Z]: attr_exists[c] = count(//*@[contains(name(), *c*)])**
    - Allows us to shrink our character set to stuff that exists in the DB
  - Speedup based on how many characters removed (10-25%)
  - Can also be used to identify Unicode and other strange encodings

How

security-assessment.com

- **Improvement: Customized Retrieval**
  - Using global count improvements we have rough idea of size of database, number of characters
  - For large document we may only want to extract 'interesting' parts
    - Skip comments, attributes, text nodes, or limit depth
  - Used to get basic idea of database structure for focused attacks
  - Variable speedup (10-50%), leads well into the next improvement


when you're kicking butt in mario cart
then you realize that you're looking at the wrong screen

How

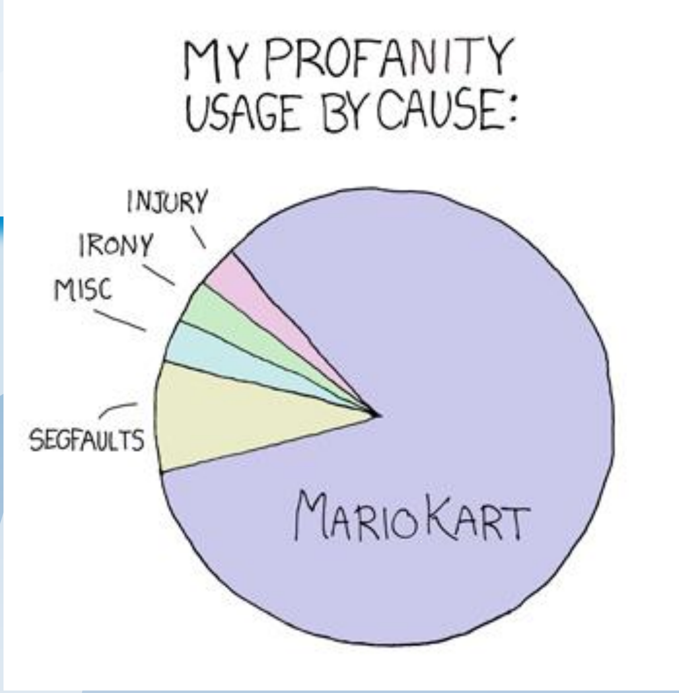
security-assessment.com

- **Improvement: String Search**
  - Perform a global search for string
  - Extract usernames, passwords, other sensitive data using optimizations
    - **//*[contains(.,“admin")]**
    - **//@*[contains(name(),“pass")]**
    - **//text()[contains(.,“secret")]**
  - Useful for open-source, known databases and finding credentials
    - Takes only as long as it needs

YO DAWG, I HERD YOU LIKE MARIO KART
SO WE PUT MARIO KART IN YOUR BIKE SO YOU CAN KART WHILE YOU BIKE
memegenerator.net

How

- **Improvement: Smart Reconstruction**
  - Useful portion of XML database is unique
    - Yet large amount of XML is structure
  - XML databases follow a predictable format
    - Sibling nodes have similar children
    - Use previous node to guess future ones
  - Significant speed improvement (80%) for ‘well-formed’ databases
    - Done by comparing new data to saved node and attributes values
  - Challenges
    - Requires knowledge using incomplete XML document
    - Additional logic required to prevent speedup inefficiencies



How

security-assessment.com

- **Improvement: Threading**
  - xcat uses threading across a linear search
    - Cannot easily thread advanced searches as they use conditional statements based on old results for future ones
  - Largest amount of time is spent reconstructing strings
    - Assign a thread to each character in string reconstruction
    - Allows use of all speedup techniques without additional complexity

I'LL HAVE YOU KNOW, I PLAYED ON RAINBOW ROAD
AND I ONLY FELL OFF 20 TIMES

How

security-assessment.com

- **Future Improvements:**
  - HTTP Keep Alive
    - Keep connections open to prevent round trip TCP setup time
  - Retrieval Resume
    - Keep information about current reconstruction, allowing restart
  - Compare/Update SQLmap
    - Compare features/Push XPath techniques back to SQLi
  - Namespace checks
  - Additional Unicode checks

- **So without much further ado**
  - The tool you've been waiting for

How

security-assessment.com

xxxpwn : “For when SQLmap isn’t working”

How

security-assessment.com

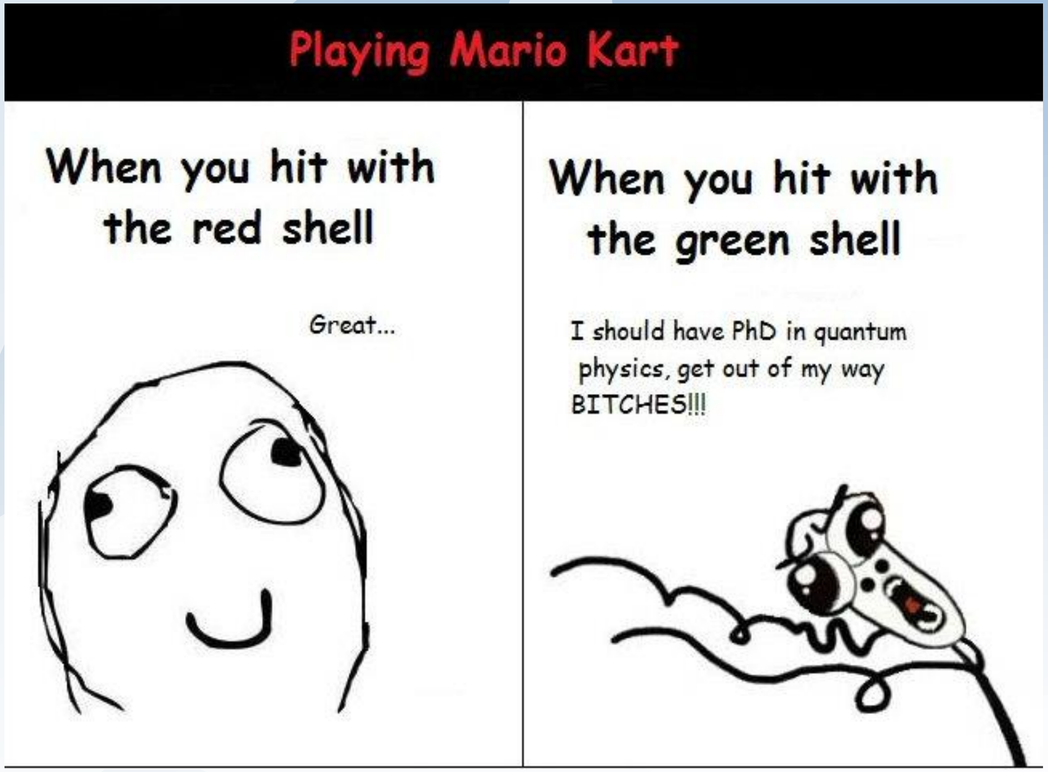
When you get hit with a green shell in mario kart
EVERY FREAKIN TIME
very often
often
somtimes
hardly ever
never
somone else sent it
After you send it and it bounces off a wall

- **xxxpwn – XPath eXfiltration eXploitation Tool**
  - Designed for blind XPath 1 injection
  - Open source, python, no dependencies
  - Almost as fast as fastest XPath 2 techniques
  - Also sounds like the title of cool hacker porn
- **Running xxxpwn for maximum satisfaction**
  - **xxxpwn.py host port** and REQUIRED flags below
  - **--match MATCH** : Keyword to match on successful blind injection
  - **--inject INJECT_FILE** : File containing valid HTTP Request
    - **$INJECT** string in file contains location of injection
  - Use **--urlencode** for GET and **--htmlencode** for POST requests
  - HTTP Host and Content-Length headers are automatically updated

How

security-assessment.com

- Speedup Improvements implemented as optional flags
  - --summary
  - --no_{root,comments,values,attributes,etc.}
  - --lowercase
  - --global_count
  - --normalize_space
  - --optimize_charset
  - --xml_match
  - --threads THREADS
  - --search SEARCH

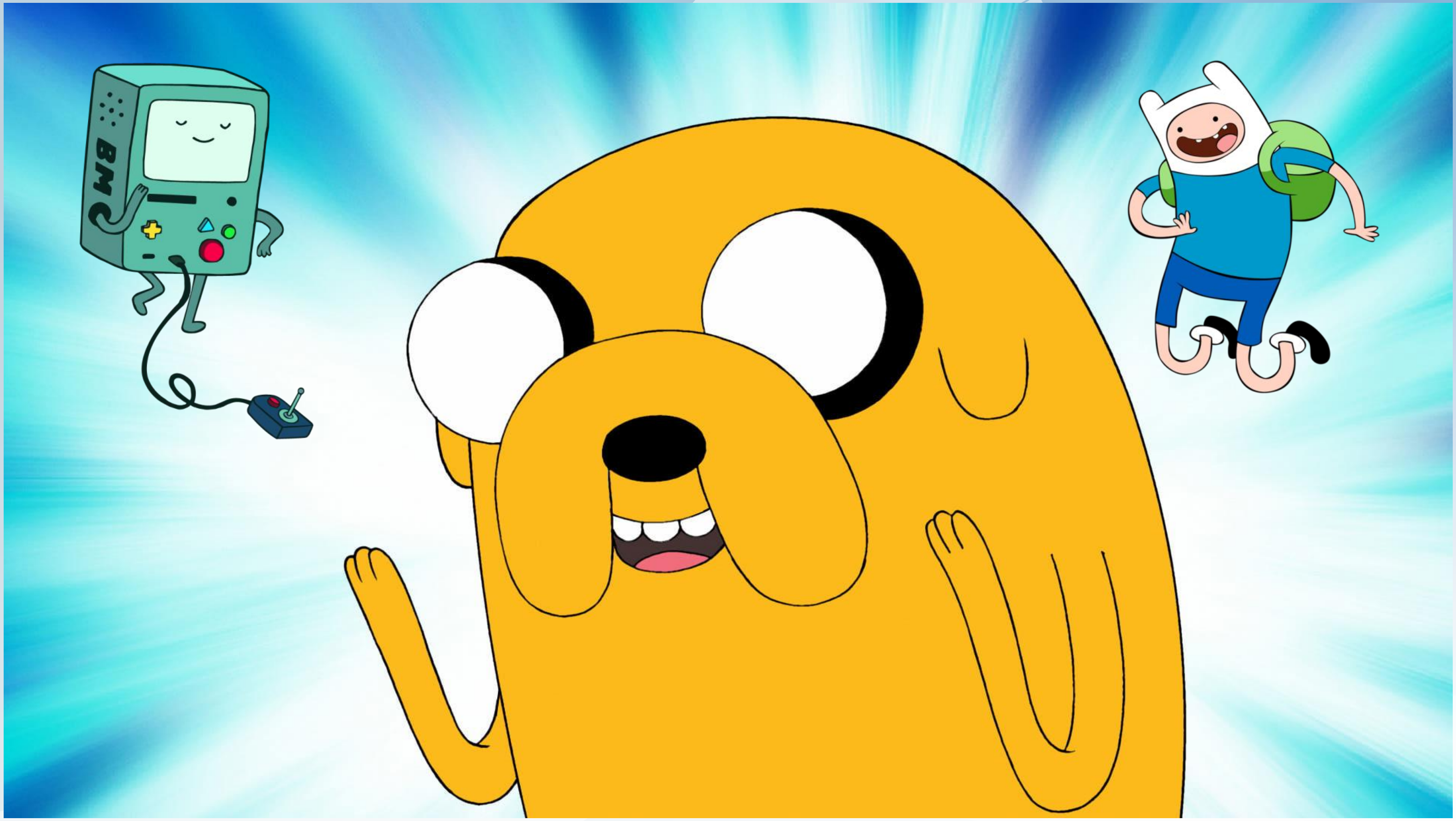

Playing Mario Kart
When you hit with the red shell
Great...
When you hit with the green shell
I should have PhD in quantum physics, get out of my way BITCHES!!!

- Additional Flags
  - IE: --ssl

How

security-assessment.com

- Adventure Time! (Tool Demo)



Ho-ow

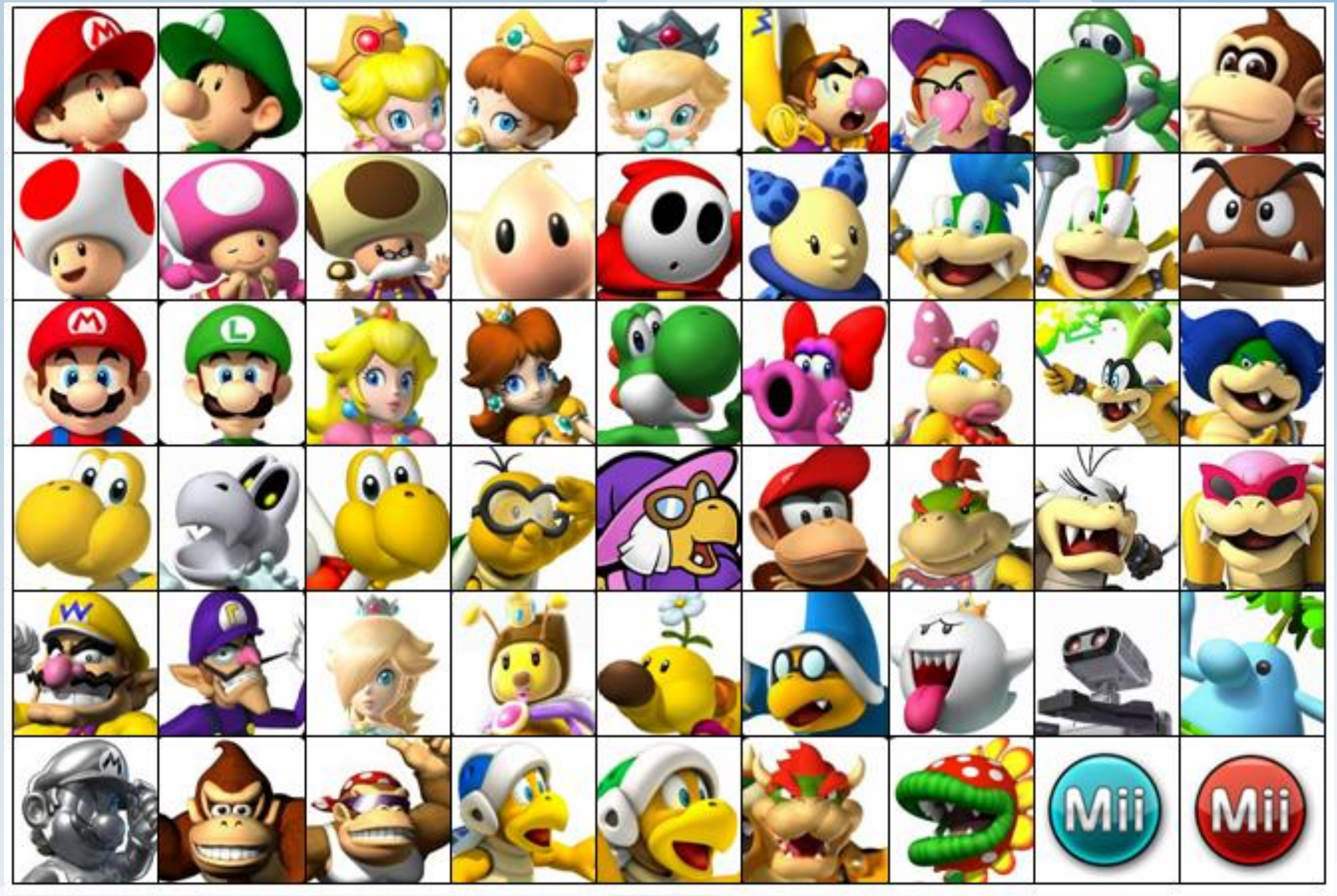
security-assessment.com

- **Retrieval Speed Comparison Results**
  - xcat version 1 - **82.37** seconds with missing characters & elements
  - xcat version 2 - **100.48** seconds with missing root comment
  - xxxpwn w/no optimizations - **12.14** seconds with missing Unicode é
  - xxxpwn w/all optimizations - **6.16** seconds complete
  - xcat autopwn – **5.33** (**7.16** with initialization) missing root comment
    - Requires XPath 2 & local HTTP server to receive results

Where

security-assessment.com

- What good is a tool without something to use it on?



Where

- **Umbraco**
  - Described as ‘The open source ASP.NET CMS’
  - Discovered by SA team during yearly hackathron
  - Vulnerable at /umbraco/dashboard.aspx?app=$INJECTION
  - No sensitive information in XML database, POC only
  - As long as they don't update to XPath 2 they will be safe
  - Payload provided in xxxpwn

+

=

Where?

security-assessment.com

- **Sitecore 6.0.0.081203 and below**
  - Described as the ‘Best .NET CMS for ASP.NET’
  - Discovered by me during a penetration test, high risk
  - SOAP methods at /sitecore/shell/WebService/service.asmx
  - Vulnerable to blind XPath injection in <vis:databaseName> field
    - Can be used to retrieve database information including username and password from the Sitecore XML database
  - Payload already loaded in xxxpwn
- **Demo**

When

security-assessment.com

- **xxxpwn available soon on Github**
  - https://github.com/feakk/xxxpwn
- **This presentation will be available on the SA website after the talk**
  - **http://security-assessment.com/**
- **I will be around the con for questions**
  - May require Mario Kart for the answer
- **Let me know if you find any vulnerabilities**
  - With responsible disclosure of course
  - Then I can feed them back into xxxpwn

Conclusion

- Security-Assessment.com is hiring, come work for us

Thanks

security-assessment.com

TOTAL POINTS

—	1	Json	69pts
—	2	CPU3	62pts
—	3	CPU2	60pts
—	4	Speed	35pts
—	5	CPU4	28pts
↑	6	CPU1	26pts
↓	7	betty♥	25pts
—	8	Fulgore	1pt