SecurityTube Python Scripting Expert (SPSE)

SecurityTube Python Scripting Expert

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 1: Fetching Web Pages

http://www.securitytube.net

Vivek Ramachandran

Course Instructor

Fetching Web Pages

- Most basic of functionality to fetch data
- urllib, urllib2
- Allows for argument encoding
- Note: Please install apache2 on our Ubuntu Server

Exercise

If you try and download a very large file, then how do you monitor the progress?

Research on urllib.urlretrieve() to solve this problem

Exercise

- Urlencode() does a bad job in handling special characters in the URL

Research on .quote() and .quote_plus() and illustrate how they can help

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 2: Parsing HTML

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Understanding Data on the Web

- Web Data is primarily:
  - HTML
  - XHTML
  - XML
  - JSON

- Need a mechanism to receive this data and parse

- Need a mechanism to generate this data and send

Parsing HTML

- Hierarchical Data

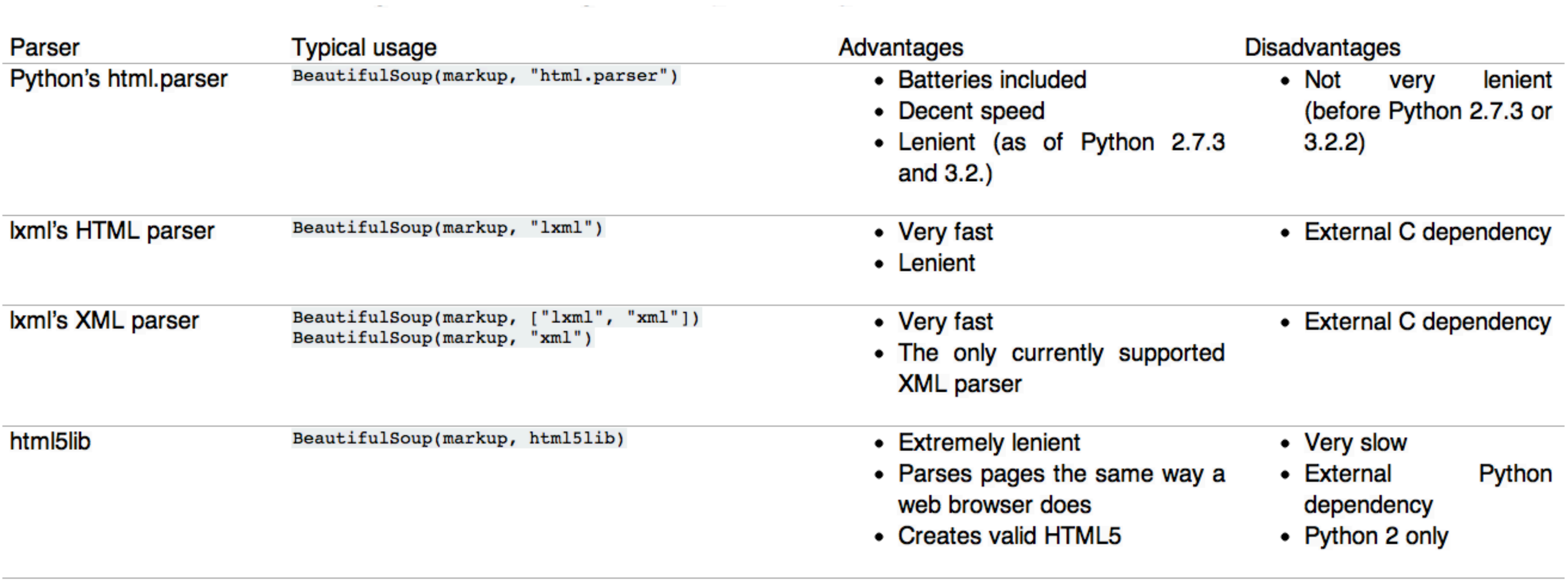
- Multiple Parsers
    - LXML
    - BeautifulSoup
    - HTMLParser
    - …

- Challenges in HTML Parsing
    - non adherence to standards
    - most websites have broken HTML documents

BeautifulSoup

- Fantastically easy to use
- Version 4 onwards allows use of lxml and html5lib – handles bad HTML better
- Till version 3 was not so great at handling bad HTML
- Handles encoding very very well!

Parser Comparison



Parser	Typical usage	Advantages	Disadvantages
Python's html.parser	`BeautifulSoup(markup, "html.parser")`	• Batteries included • Decent speed • Lenient (as of Python 2.7.3 and 3.2.)	• Not very lenient (before Python 2.7.3 or 3.2.2)
lxml's HTML parser	`BeautifulSoup(markup, "lxml")`	• Very fast • Lenient	• External C dependency

lxml's XML parser	`BeautifulSoup(markup, ["lxml", "xml"])`
	`BeautifulSoup(markup, "xml")`
		• Very fast
		• The only currently supported XML parser
			• External C dependency

html5lib	`BeautifulSoup(markup, html5lib)`	• Extremely lenient • Parses pages the same way a web browser does • Creates valid HTML5	• Very slow • External Python dependency • Python 2 only

http://www.crummy.com/software/BeautifulSoup/bs4/doc/

Exercise

- Read the documentation of BeautifulSoup 4 and find other ways to iterate through tags and get to the juicy information

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

End of Part 2: Parsing HTML

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 3: Coding a Screen Scraper

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Screen Scraper

- Very Very Dependent on the HTML structure
- Slight change might break scraper depending on how you’ve coded it

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

End of Part 3: Coding a Screen Scraper

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 4: Form Parsing and Submission with Mechanize

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Mechanize

- Based on the Perl module WWW:Mechanize
- Allows for stateful programming and browser emulation
- Very powerful yet easy way to work through a website

Exercise

- In the example shown we did not try and modify the hidden fields. Try to see how you can do it and send arbitrary data ☺

Exercise

- Install a vulnerable web application such as DVWA, OWASP Web Goat or other
- Use mechanize to try SQL Injection on form fields and deduce which fields are vulnerable to SQL Injection

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

End of Part 4: Form Parsing and Submission with Mechanize

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 5: Stateful Web Application Browsing with Mechanize

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

The Stateful Web

- Mechanize handles cookies by itself ☺
- We need to understand how to “browse” the application
- “click links”, “fill and submit forms”, “maintain state”

Exercise CookieJAR

- Explore the concept of mechanize.CookieJar
- Why is it useful?
- Sample code to illustrate its functionality

Exercise

- Explore http://seleniumhq.org/support/

- Can you automate the current example in it?

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

End of Part 5: Stateful Web Application Browsing with Mechanize

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 6: XML Parsing and Web Services

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

XML Parsing

- Can you the lxml parser
- Can use BeautifulSoup as well

Exercise

- Web Services are an important part of web communication now

- Zolera Soap Infrastructure

http://pywebsvcs.sourceforge.net/zsi.html

Attack on WebGoat

http://yehg.net/lab/pr0js/training/webgoat.php#Web_Services

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

End of Part 6: XML Parsing and Web Services

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 7: Exercise Series 1

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Proxy Support Exercise

- Investigate on how you can use Proxy support with:
  - BeautifulSoup
  - urllib
  - mechanize

Web Spider Exercise

- Create a Multi-Threaded Web Spider which
  - takes a website and depth of spidering as input
  - download the HTML files only
  - Inserts the HTML into a MySQL Database
    - Design the Schema
  - It also parses the Forms on each page
    - inserts into DB with details of Form fields

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

End of Part 7: Exercise Series 1

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 8: OWASP Top 10 Attack Scripting Exercise

http://www.securitytube.net

Vivek Ramachandran
Course Instructor

OWASP Top 10

- A1: Injection
- A2: Cross-Site Scripting (XSS)
- A3: Broken Authentication and Session Management
- A4: Insecure Direct Object References
- A5: Cross-Site Request Forgery (CSRF)
- A6: Security Misconfiguration
- A7: Insecure Cryptographic Storage
- A8: Failure to Restrict URL Access
- A9: Insufficient Transport Layer Protection
- A10: Unvalidated Redirects and Forwards

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Massive Exercise ☺

- For each of the OWASP Top 10 create Python scripts which can automate the testing of the vulnerability

- Vulnerable software to use: **Mutillidae**

http://www.irongeek.com/i.php?page=mutillidae/mutillidae-deliberately-vulnerable-php-owasp-top-10

Further Study

Offensive Python for Web Hackers talk at Blackhat 2010 by Nathan Hamiel and Marcin Wielgoszewski

Video: http://www.securitytube.net/video/1142

Module 4: Attacking Web Application

SecurityTube Python Scripting Expert

Part 8: OWASP Top 10 Attack Scripting Exercise

http://www.securitytube.net

Vivek Ramachandran
Course Instructor