Constricting the Web

Offensive Python for Web Hackers
Yes, We are Weird

Nathan Hamiel
Principal Consultant

Marcin Wielgoszewski
Security Engineer
GOTHAM DIGITAL • SCIENCE

Associate Professor at UAT
This is Important

- Reliance on tools can = Fail!
  - Many more people testing web apps
  - Vendors play catch-up
  - Success is on your shoulders

- Difficult cases
  - APIs and specialized data formats
  - Sequenced operations
  - Randomized data
An AppSec Intervention

I SEE A BUNCH OF PEOPLE WHO LOVE YOU LIKE CRAZY

Defcon 18
Why Python?

• Language specific
  – Object-oriented
  – Byte compiled
  – Fast

• Wide support
  – Many security tools written in Python
  – Plenty of help available
  – Plenty of resources for learning available
Where Does Python Fit?

Defcon 18
A Few Tools

Scapy
w3af
sqlmap
Peach
Pcapy
DeBlaze
MyNav

sulley
Pyscan
MonkeyFist

SpikeProxy

wapiti
Canvas

ProxyStrike

Idapython

Defcon 18
Python Implementations

• CPython
  – http://python.org
• Jython
  – http://jython.org
• IronPython
  – http://ironpython.net
Want To learn Python

• Start with http://python.org
  – http://docs.python.org/
  – http://docs.python.org/tutorial/index.html

• Google’s Python Class
  – http://code.google.com/edu/languages/google-python-class/

• There are differences between Python 2.x and 3.x
First Things First

• Walk like a duck and quack like a duck
# Helpful Modules

<table>
<thead>
<tr>
<th>Standard Lib</th>
<th>3rd Party</th>
</tr>
</thead>
<tbody>
<tr>
<td>• httpplib</td>
<td>• httpplib2</td>
</tr>
<tr>
<td>• urllib / urllib2</td>
<td>• lxml</td>
</tr>
<tr>
<td>• urllibparse</td>
<td>• zsi / suds</td>
</tr>
<tr>
<td>• HTMLParser</td>
<td>• PyAMF</td>
</tr>
<tr>
<td>• struct</td>
<td>• pydermonkey</td>
</tr>
<tr>
<td>• xml</td>
<td>• Twisted</td>
</tr>
<tr>
<td>• json (Python 2.6)</td>
<td></td>
</tr>
<tr>
<td>• difflib</td>
<td></td>
</tr>
</tbody>
</table>

---

*Defcon 18*
Capabilities of HTTP Modules

- **httpplib**
  - Standard HTTP Module
  - Good for GETs and POSTs
  - HTTP / HTTPS support

- **httpplib2**
  - Expanded HTTP method support
  - Supports various auth methods
  - Automatically follows 3xx redirects
More Modules

• urllib
  – High level module for opening resources
  – Has URL encoding capabilities

• urllib2
  – Expanded support for handlers

• Merged in Python 3 along with urlparse
Basic HTTP Clients

- Examples
Encoding and Data Types

• Perform transition magic
  – URL encoding and Escaping
  – String methods (base64 / hex / rot13, etc)
  – Data representations (decimals / entities / etc)

• DharmaEncoder
  – Provides methods to encode and wrap values
  – http://hexsec.com/labs
DharmaEncoder

Defcon 18
Fuzz Cases

• Do the legwork
  – Know your app
  – Know your parameters
  – Know your data

• Work smarter
  – Create accurate ranges
  – itertools methods
  – Don’t empty the clip
• Web fuzzing lib for Python
  – http://code.google.com/p/pywebfuzz/
  – Usable in Python 2.x
  – Easy to distributable and repeat tests
• Convenience
  – Fuzzdb values accessible through classes
  – Request Logic
  – Range generation and encoding /decoding
pywebfuzz Examples

- Basic request fuzzing
- Finding an error condition

Defcon 18
First things first

- Determine content type, use appropriate parser
- Don’t use HTMLParser

```python
if html:
    use lxml.html
elif xhtml:
    use lxml.etree
elif xml:
    use lxml.etree
elif json:
    use json
```
Sequenced Operations

Defcon 18
Sequence Difficulties

• State Issues
  – Account login / logout
  – Randomized values
  – Maintaining proper state while testing

• Request
  – Process headers (referer and cookies)
  – Unable to parse content properly
  – Resort to regular expressions

Defcon 18
Test Driving the Browser

- Selenium
  - http://seleniumhq.org/
- Windmill
  - http://www.getwindmill.com/
Browser Integration

- Firefox / XULRunner
  - pyxpcomext
    - http://pyxpcomext.mozdev.org/no_wrap/tutorials/pyxulrunner/
      python_xulrunner_about.html

- Webkit
  - PyGtk / PyWebKitGtk
    - http://code.google.com/p/pywebkitgtk/
  - PyQt
    - http://wiki.python.org/moin/PyQt4
  - PySide (Official Support from Nokia)
    - http://www.pyside.org/

Defcon 18
Webviews

• Render returned requests from other libs in just a couple of lines of code

```python
from PyQt4.QtGui import *
from PyQt4.QtWebKit import *
import httplib2

http = httplib2.Http()
headers, content = http.request("http://python.org", "GET")
app = QApplication(sys.argv)
web = QWebView()web.setHtml(content)
web.show()
sys.exit(app.exec_())
```
Example

Python Programming Language -- Official Website

Tribon uses Python...

... joining users such as Rackspace, Industrial Light and Magic, AstraZeneca, Honeywell, and many others.

What they are saying...

Thawte Consulting:

"Python makes us extremely productive, and makes maintaining a large and rapidly evolving codebase relatively simple," said Mark Shuttleworth.

more...

Using Python For...

- Web Programming
- CGI, Zope, Django, TurboGears, XML
- Databases
- ODBC, MySQL
- GUI Development
- wxPython, Tkinter, PyGtk, PyQt
- Scientific and Numeric
- Bioinformatics, Physics
- Education
- pypiloto, Software Carpentry Course

Defcon 18
Web Services

• Traditional
  – ZSI
  – Suds

• RESTful
  – Both High and Low Rest
  – httpplib
  – httpplib2
Web Services Examples

• Example

Defcon 18
Passive Content Analysis

• Identify issues passively
  – Cookie issues
  – Cache-control
  – Encoding issues

• Augment other tools
  – Perform inspection on captured data
  – Use your favorite inspection proxy
  – No need to send data to endpoint
Working with Flex

- PyAMF is most popular
- Action Message Format encoder/decoder
- Create remoting clients, gateways
- Bind client-side classes to server-side POJOs
Object Factories

• Start with a simple Python design pattern

```python
class Factory(object):
    def __init__(self, *args, **kwargs):
        self.__dict__.update(kwargs)

pyamf.register_class(Factory, "namespace.of.object.Class")
```
Binary Protocols

• You’re presented with an app that communicates via a custom binary protocol
• Oh what to do without my scanner...
Intro to Struct Module

- Convert between Python values and C structs
Example Binary Protocol

U8 = unsigned 8-byte integer
U16 = unsigned 16-byte integer
UTF-8 = U16 * (UTF8-char) ; as defined in RFC3629
DOUBLE = 8-byte IEEE-754 double precision
; floating point in network byte order

msg = message-count parameters
message-count = U16
parameters = number-type | boolean-type | string-type
number-marker = 0x00
boolean-marker = 0x01
string-marker = 0x02
number-type = number-marker DOUBLE
boolean-type = boolean-marker U8
string-type = string-marker UTF-8
Working with Numbers

- Write the appropriate type-marker to buffer
- Followed by the value as a Double

```python
buf.write("\x00")
buf.write(struct.pack("!d", val))
```
Working with Numbers

• Reading is just the opposite
• Struct unpacks into a Tuple

```python
while pos < len(buf):
    ..snip..
    if buf[pos] == '\x00':
        pos += 1
        val = struct.unpack('!d', buf[pos:pos+8])[0]
        pos += 8
```
Booleans

• Writing a Boolean

def write_bool(buf, val):
    buf.write("\x01")
    buf.write(struct.pack("?", val))
• Parsing a Boolean

```python
while pos < len(buf) + 1:
    ..snip..
    if buf[pos] == '\x01':
        pos += 1
        val = struct.unpack('?\x00', buf[pos])[0]
        pos += 1
```
Strings

• Writing a String

def write_string(buf, val):
    u = val.encode("utf-8")
    strlen = len(u)
    buf.write("\x02")
    buf.write("H%ds" % strlen, strlen, u)
while pos < len(buf) + 1:
  ..snip..

if buf[pos] == '\x02':
  pos += 1
  s_len = struct.unpack("H", buf[pos:pos+2])[0]
  pos += 2
  val = struct.unpack("%ds" % strlen, buf[pos:pos+s_len])[0]
  pos += s_len
Congratulations!

• You may have noticed that we wrote a simple state-machine

• A while loop that iterates over a buffer, keeping track of the state it’s in

• Here’s a cookie:  <cookie pic here>
def decode(buf):
    state = "START"

    while pos < len(buf):
        if state == "START":  # get message count
            #
            get message count
        elif state == "MARKER":  # parse marker
            # parse marker
        elif state == "NUMBER":  # parse number
            # parse number
        elif state == "BOOL":  # parse boolean
            # parse boolean
        elif state == "STRING":  # parse string
            # parse string
Questions?

Nathan Hamiel
http://www.neohaxor.org/
Twitter
https://twitter.com/nathanhamiel

Marcin Wielgoszewski
http://www.tssci-security.com/
Twitter
https://twitter.com/marinw