Web client programming
JavaScript/AJAX
Web requests with JavaScript/AJAX

- Needed for reverse-engineering homework site
- Web request via jQuery JavaScript library
  
  ```javascript
  jQuery.ajax({
    'type': 'GET',
    'url': 'http://vulnerable/ajax.php',
    'success': function(data) {
      console.log(data);
    }
  });
  ```

  ```javascript
  jQuery.ajax({
    'type': 'POST',
    'url': 'http://vulnerable/ajax.php',
    'data': 'hello world',
    'success': function(data) {
      console.log(data);
    }
  });
  ```

- Example of form submission via embedded AJAX script (leForm)
  - [http://cs410.oregonctf.org](http://cs410.oregonctf.org)
Python Requests
Python Requests

- HTTP for humans
- Programmatically handle HTTP
  - Requests and responses
  - Authentication
  - Headers
  - Forms
  - Cookies
  - Sessions
  - JSON
- Can be used to solve each level
  - Submit solution scripts as part of lab notebook
Setting up

- Install python3, python-pip, virtualenv (apt-get)
- Then, set up a local python3 instance in directory `env` for use during the rest of the course
  ```
  mkdir env
  virtualenv -p /usr/bin/python3 env
  ```
- Enter the local python3 environment (always do this)
  ```
  source env/bin/activate
  ```
- Install requests into environment
  ```
  pip install requests
  ```
- Install beautifulsoup (bs4) into environment
  ```
  pip install bs4
  ```
- Run your scripts (either via interactive prompt or as a file)
  ```
  python 01.py
  ```
Requests and responses

- Methods in Python requests package map to HTTP methods (e.g. `requests.get => GET`)
- Simple HTTP request

```python
import requests
r = requests.get('http://thefengs.com')
print(r.text)
print(r.status_code)
print(r.headers)
```
Sessions

- Emulate web browser
- Accumulate cookies
- Remember header and authentication settings

```python
import requests
s = requests.Session()
print(s.cookies)
r = s.get('http://facebook.com')
print(s.cookies)
r = s.get('http://google.com')
print(s.cookies)
```
Forms

- Named parameter `data`
- Given as a dictionary
  - An associative array of key:value pairs in python
- Two possible methods: GET, POST
  - Examine form to find URL, method, and field name

```python
import requests
url = 'http://totoro.cs.pdx.edu/loginproc.php'
s = requests.Session()
formdata = {'username':'foo', 'password':'bar'}
r = s.post(url, data=formdata)
print(r.text)
formdata = {'username':'wuchang', 'password':'NjMxMjQx'}
r = s.post(url, data=formdata)
print(r.text)
```
Basic Authentication

- Named parameter `auth`
- Given as a tuple (an immutable list in python)

```python
import requests
url = 'http://natas0.natas.labs.overthewire.org'
r = requests.get(url)
print(r.status_code)
print(r.headers)
r = requests.get(url, auth=('natas0', 'natas0'))
print(r.status_code)
print(r.text)
```
Setting request headers

- Named parameter `headers` for both reading HTTP response headers and setting HTTP request headers
- Given as a dictionary
  - An associative array of key:value pairs in python
  - Can set per-request or across a session

```python
import requests
myheaders = {'referer':'http://natas5.natas.labs.overthewire.org/'}
url = 'http://natas4.natas.labs.overthewire.org'
r = requests.get(url,auth=('natas4','the_natas4_pass'),headers=myheaders)
print(r.text)
```

```python
import requests
s = requests.Session()
s.headers.update({'User-Agent':'Python Requests'})
url = 'http://natas25.natas.labs.overthewire.org/
r = s.get(url,auth=('natas25', 'the_natas25_pass'))
```
Setting cookies

- Named parameter `cookies` for both reading cookies in response and setting cookies in request
- Give as a dictionary
  - An associative array of key:value pairs in python
  - Encodes `key=value` in `Cookie:` field

```python
import requests
url = 'http://natas5.natas.labs.overthewire.org'
mycookies = {'loggedin': '1'}
r = requests.get(url, auth=('natas5', 'natas5_pass'), cookies=mycookies)
print(r.text)
```
Reading cookies

- Returned in response via a `CookieJar` named `cookies`
- Automatically added to session `CookieJar` if session is used
- Can be indexed similar to a dict()

```python
import requests
url = 'http://natas21-experimenter.natas.labs.overthewire.org/index.php'
r = requests.get(url)
sessionid = r.cookies['PHPSESSID']
print(sessionid)
```

```python
import requests
s = requests.Session()
r = s.get('http://espn.go.com/)
r = s.get('http://facebook.com/)
for cookie in s.cookies:
    print(cookie)
```
URL-encoding

- Encoding data for transmission in HTTP
  - Necessary for some injection levels to send a carriage return in form submission that is not interpreted by HTTP protocol

```python
import urllib.parse
print(urllib.parse.quote(': \r\n'))
```
HTML parsing

- BeautifulSoup

```python
import requests
from bs4 import BeautifulSoup
url = 'http://espn.go.com/
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')
for link in soup.find_all('a'):
    print(link.get('href'))
```
JSON and REST

- JSON (rather than XML) often returned when transmitting web objects
  - Encodes a serialized data structure to and from server
  - Typically translated to/from dictionaries in Python
  - Example sending a JSON object to a REST API call and receiving a JSON response

```python
# Set up the order
orders_url = "https://api.stockfighter.io/ob/api/venues/NYSE/stock/AAPL"
myorder = {
    'account': 3000001,
    'price': 4400,
    'qty': 100,
    'direction': 'buy',
    'orderType': 'limit'
}
r = requests.post(orders_url, data=json.dumps(myorder))
r_data = r.json()
print(r_data['id'])
```