WebApp Security/Hacking Workshop 101

N0L3ptr:

W. Owen Redwood

Ivan Lozano

Clark Wood

Disclaimer

This presentation is presenting ideas and topics that are common knowledge in the security industry, and can be found freely and easily.

You are solely responsible for any actions you take with the topics presented in this presentation.

BurpSuite

http://portswigger.net/burp/

A wonderful integrated platform for pen-testing security of web applications.

Key Features:

- Interception Proxy
- Application aware spider
- Vulnerability scanner (not free).... nikto is a good alternative
- And more

Common Alternative to BurpSuite is OWASP's WebScarab:
https://www.owasp.org/index.php/Category:OWASP_WebScarab_Project

Great Tutorials for Burp

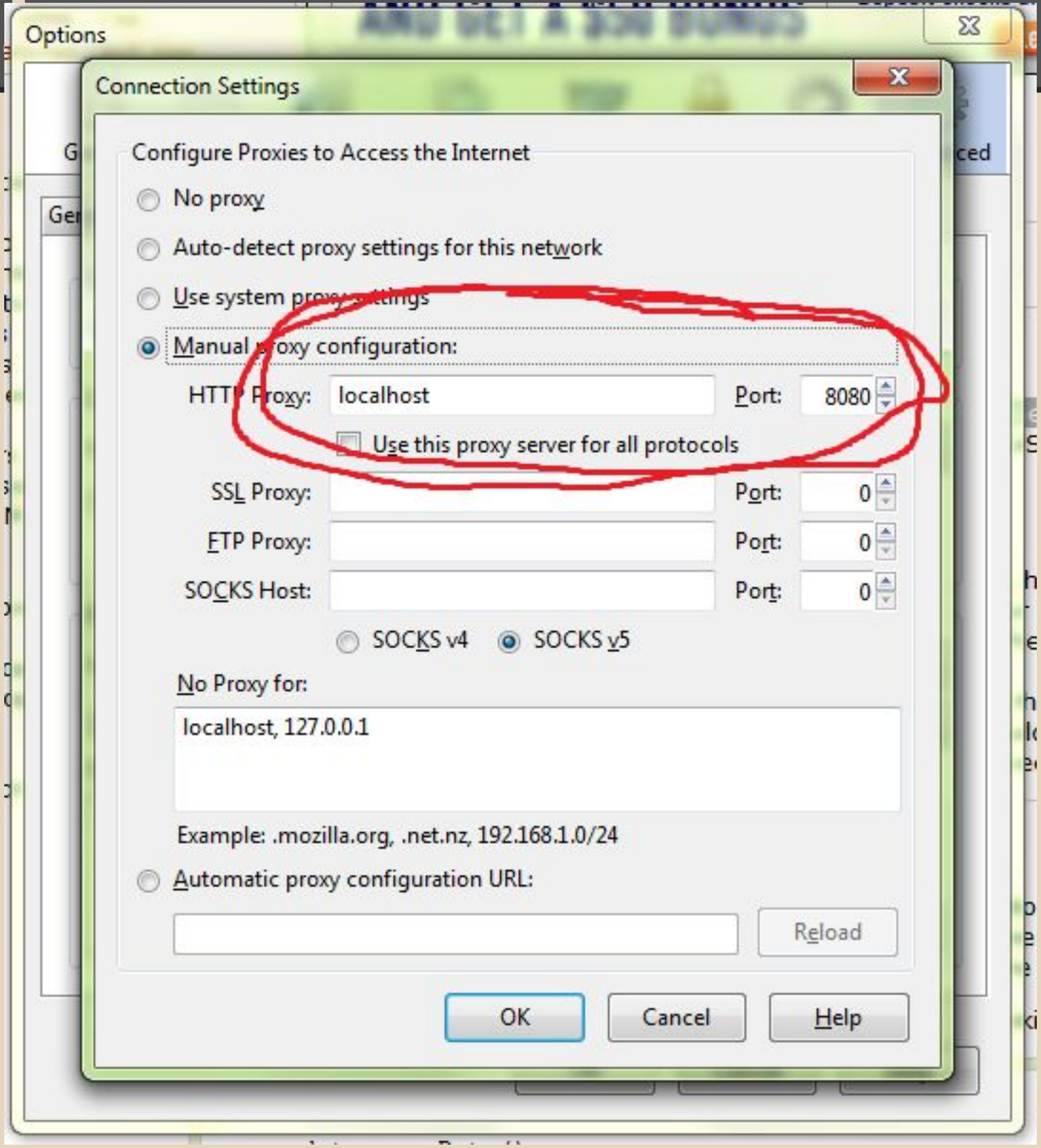
Configuring for firefox: http://www.youtube.com/watch?v=Fj0n17Jtnzw

General Use: http://vimeo.com/11553558

(Watch these if you cannot attend)

Set Up (Firefox)

Configure Browser to use it as a proxy:



Connection Settings

Configure Proxies to Access the Internet

- No proxy
- Auto-detect proxy settings for this network
- Use system proxy settings
- [✓ Manual proxy configuration:]

HTTP Proxy: localhost Port: 8080

Use this proxy server for all protocols

SSL Proxy: Port: 0

FTP Proxy: Port: 0

SOCKS Host: Port: 0

SOCKS v4 [✓ SOCKS v5]

No Proxy for:

localhost, 127.0.0.1

Example: .mozilla.org, .net.nz, 192.168.1.0/24

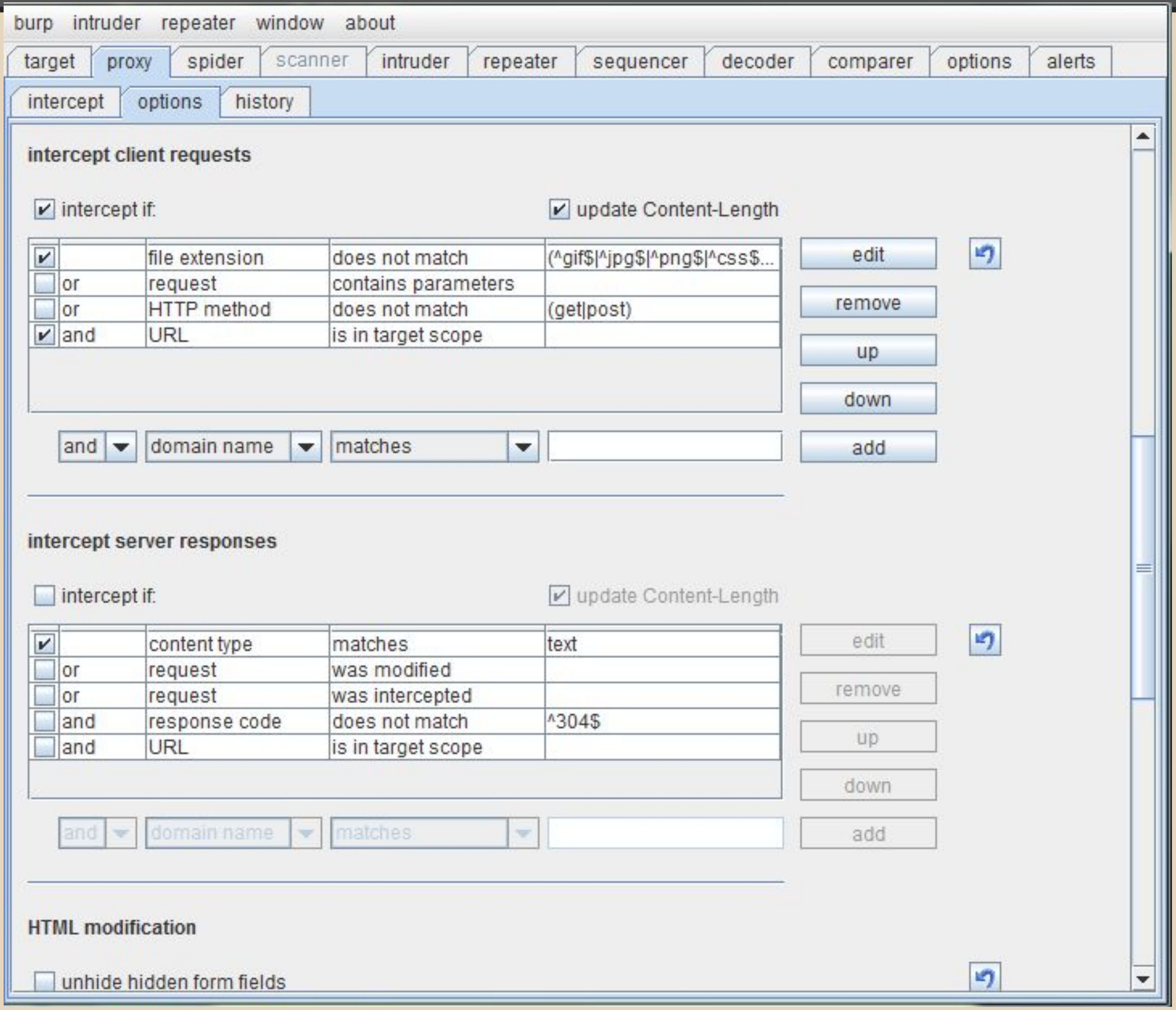
- Automatic proxy configuration URL:

Set Up (Burp)

Confine the scope to the website:

Goto Proxy->Options, and scroll down to Intercept client requests....

Check:
and URL.. is in target scope


burp intruder repeater window about
target proxy spider scanner intruder repeater sequencer decoder comparer options alerts
intercept options history
edit
remove
up
down
add

intercept client requests

☑ intercept if: ☑ update Content-Length

☑		file extension	does not match	(^gif$\|^jpg$|^png$\|^css$...
☐	or	request	contains parameters	
☐	or	HTTP method	does not match	(get|post)
☑	and	URL	is in target scope	

and | domain name | matches

intercept server responses

☐ intercept if: ☑ update Content-Length

☑		content type	matches	text
☐	or	request	was modified	
☐	or	request	was intercepted	
☐	and	response code	does not match	^304$
☐	and	URL	is in target scope	

and | domain name | matches

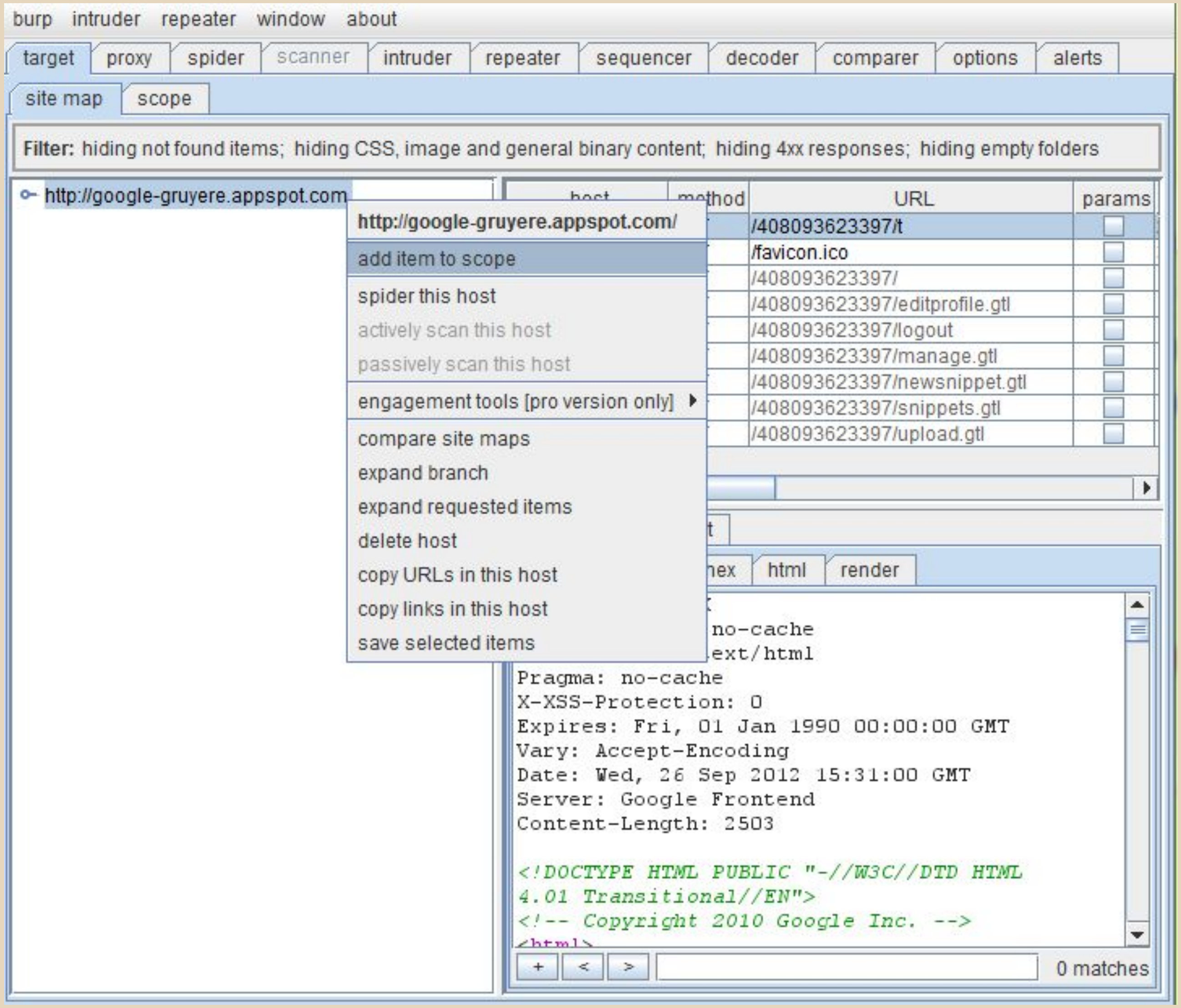
HTML modification

☐ unhide hidden form fields

Setting Scope (Burp)

Once we've done all that, the final step is to NAVIGATE to the target site.

- it should appear in the target->site_map tab
- Any other resources/sites accessed by the site will also appear (ads, etc...)
- Right click on site, then click add item to scope to set the scope to that site.
- Then we're ready to play!



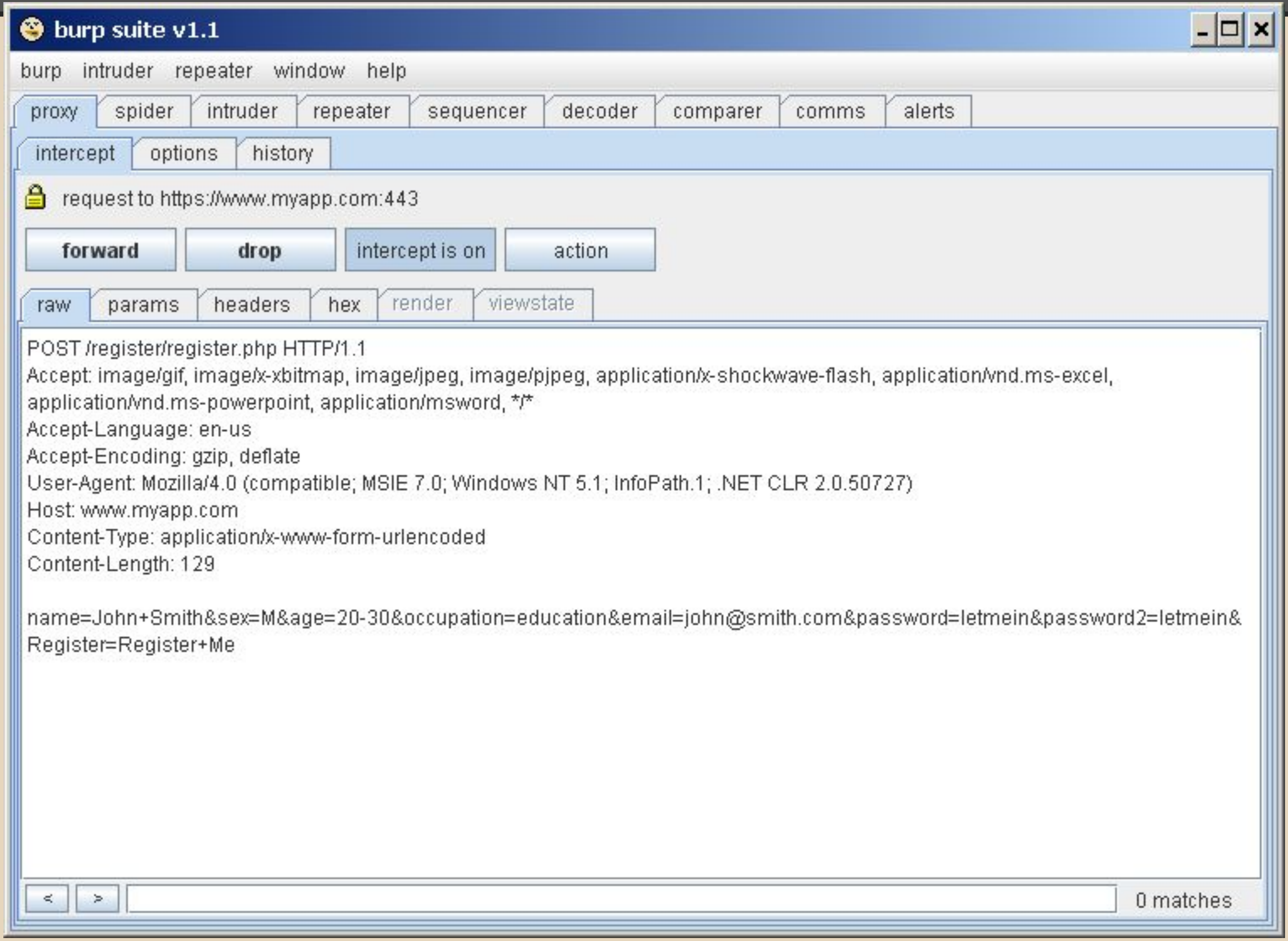
Filter: hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

URL
/408093623397/t
/favicon.ico
/408093623397/
/408093623397/editprofile.gtl
/408093623397/logout
/408093623397/manage.gtl
/408093623397/newsnippet.gtl
/408093623397/snippets.gtl
/408093623397/upload.gtl

```
no-cache
ext/html
Pragma: no-cache
X-XSS-Protection: 0
Expires: Fri, 01 Jan 1990 00:00:00 GMT
Vary: Accept-Encoding
Date: Wed, 26 Sep 2012 15:31:00 GMT
Server: Google Frontend
Content-Length: 2503

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN">
<!-- Copyright 2010 Google Inc. -->
```

Interception Proxy


burp suite v1.1
burp intruder repeater window help
proxy spider intruder repeater sequencer decoder comparer comms alerts
intercept options history
request to https://www.myapp.com:443
forward drop intercept is on action
raw params headers hex render viewstate
< >
0 matches

```
POST /register/register.php HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel,
application/vnd.ms-powerpoint, application/msword, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; InfoPath.1; .NET CLR 2.0.50727)
Host: www.myapp.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 129

name=John+Smith&sex=M&age=20-30&occupation=education&email=john@smith.com&password=letmein&password2=letmein&
Register=Register+Me
```

Disclaimer part 2

The site we will use:

http://google-gruyere.appspot.com/#0__hackers

^ All quoted material from this site

"**WARNING:** Accessing or attacking a computer system without authorization is illegal in many jurisdictions. While doing this codelab, you are specifically granted authorization to attack the Gruyere application as directed. You may not attack Gruyere in ways other than described in this codelab, nor may you attack App Engine directly or any other Google service. You should use what you learn from the codelab to make your own applications more secure. You should not use it to attack any applications other than your own, and only do that with permission from the appropriate authorities (e.g., your company's security team)."

Common Web Vulnerabilities

Cross Site Scripting (XSS)

Client-State Manipulation

Cross-Site Request Forgery (XSRF)

Cross Site Script Inclusion (XSSI)

Denial of Service (DoS)

Code Execution / Remote Code Execution

AJAX Vulnerabilities

Buffer/Integer Overflow, SQL Injection, Misconfiguration

Cross Site Scripting (XSS)

Vulnerability which permits an attacker to inject code (HTML or Javascript) into the contents of a website.

Bypasses browser's same origin policy and can steal cookies and user's private info on the website.

"Typically, if you can get Javascript to execute on a page when it's viewed by another user, you have an XSS vulnerability."

Same Origin Policy

http://en.wikipedia.org/wiki/Same_origin_policy

This is a browser-side policy that permits scripts running on pages originating from the same site to access each other's methods & properties/resources with no specific restrictions, but prevents access to most methods & properties/resources across pages on different sites [http://www.w3.org/Security/wiki/Same_Origin_Policy]

in other words: facebook.com shouldn't be able to access cookies and methods from gmail.com

Simple File Upload XSS Demo

Follow along with
http://google-gruyere.appspot.com/part2#2__xss_challenge

we'll upload a file with the following html code:

```
<html><head><script>
alert(document.cookie)
</script></head>
<body></body>
</hmtl>
```

Fun Facts

- Each time the same computer requests a page with a browser, it will send the cookie too.
  - Can have XSS attack send your cookies to another site
- Browsers may re-issue GET requests, which make actions get executed more than once, but browsers won't reissue POST requests without USER consent

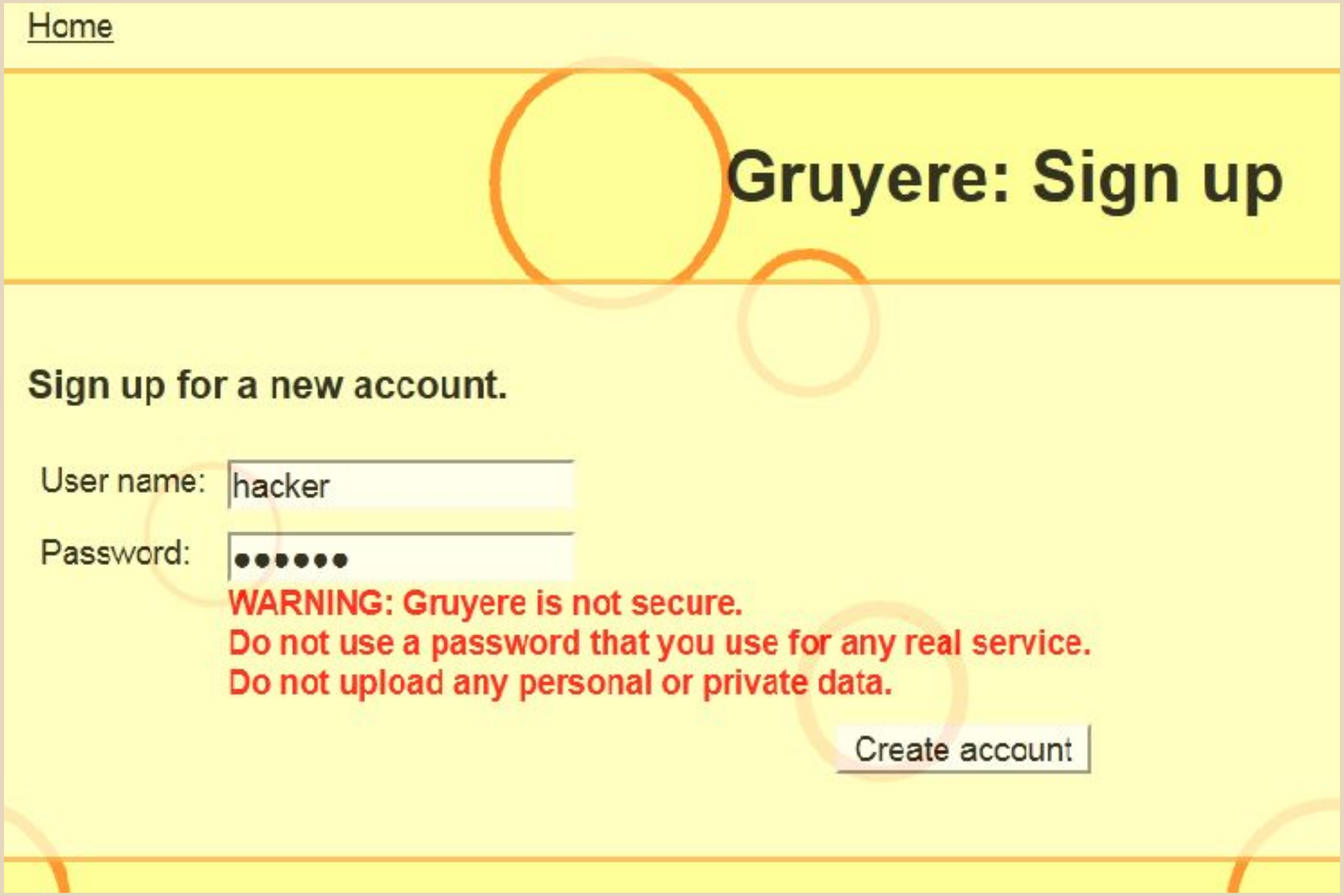
Client State Manipulation

Manipulate client resources (cookie, headers, etc).

Lets use BurpSuite to do this!

Lets make ourselves an admin :D

Lets create a new account


Home

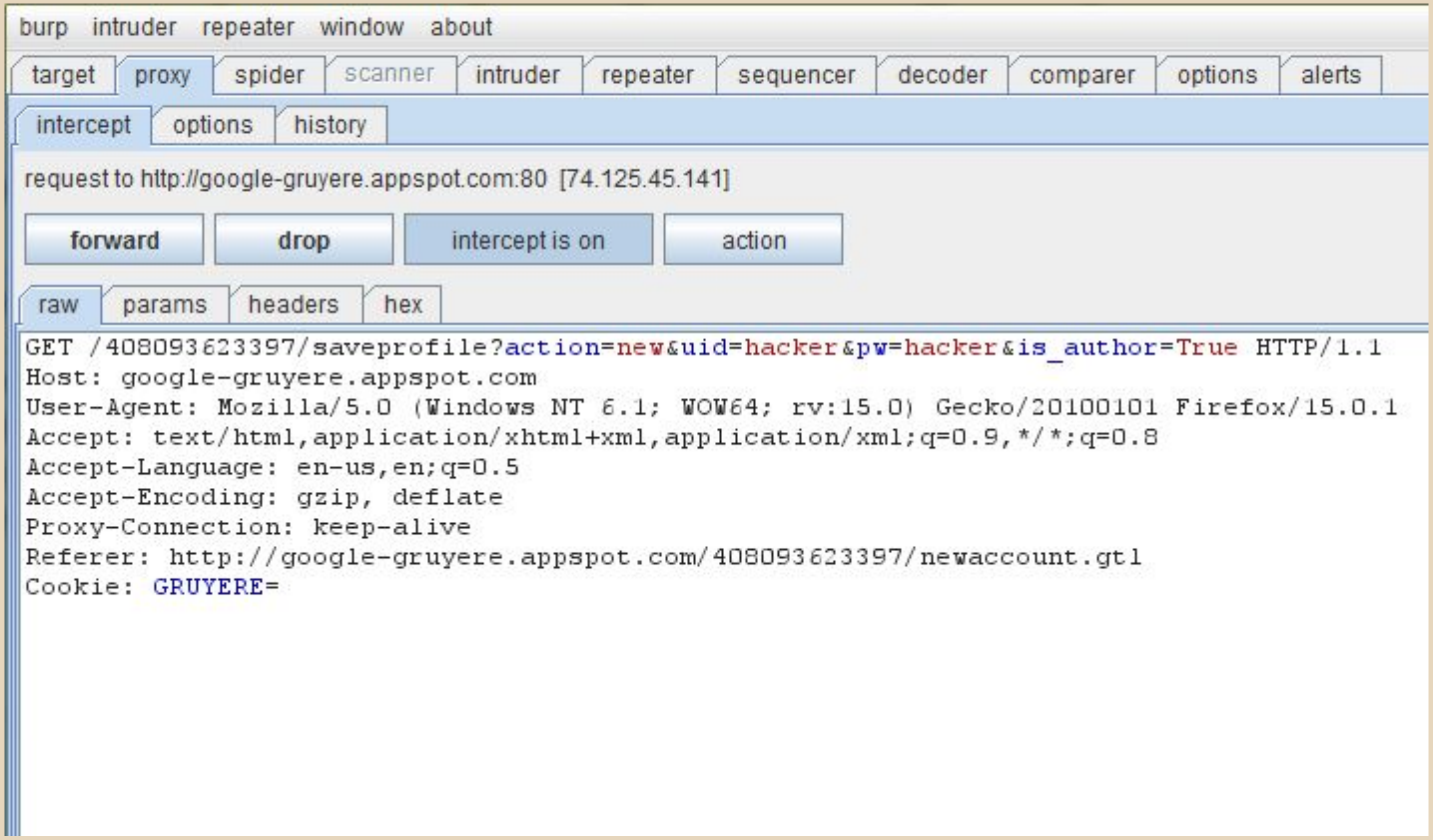
Gruyere: Sign up

Sign up for a new account.

User name:	hacker
Password:	●●●●●●

WARNING: Gruyere is not secure.
Do not use a password that you use for any real service.
Do not upload any personal or private data.

Intercept the packet


burp intruder repeater window about
target proxy spider scanner intruder repeater sequencer decoder comparer options alerts
intercept options history
forward
drop
intercept is on
action
raw params headers hex

request to http://google-gruyere.appspot.com:80 [74.125.45.141]

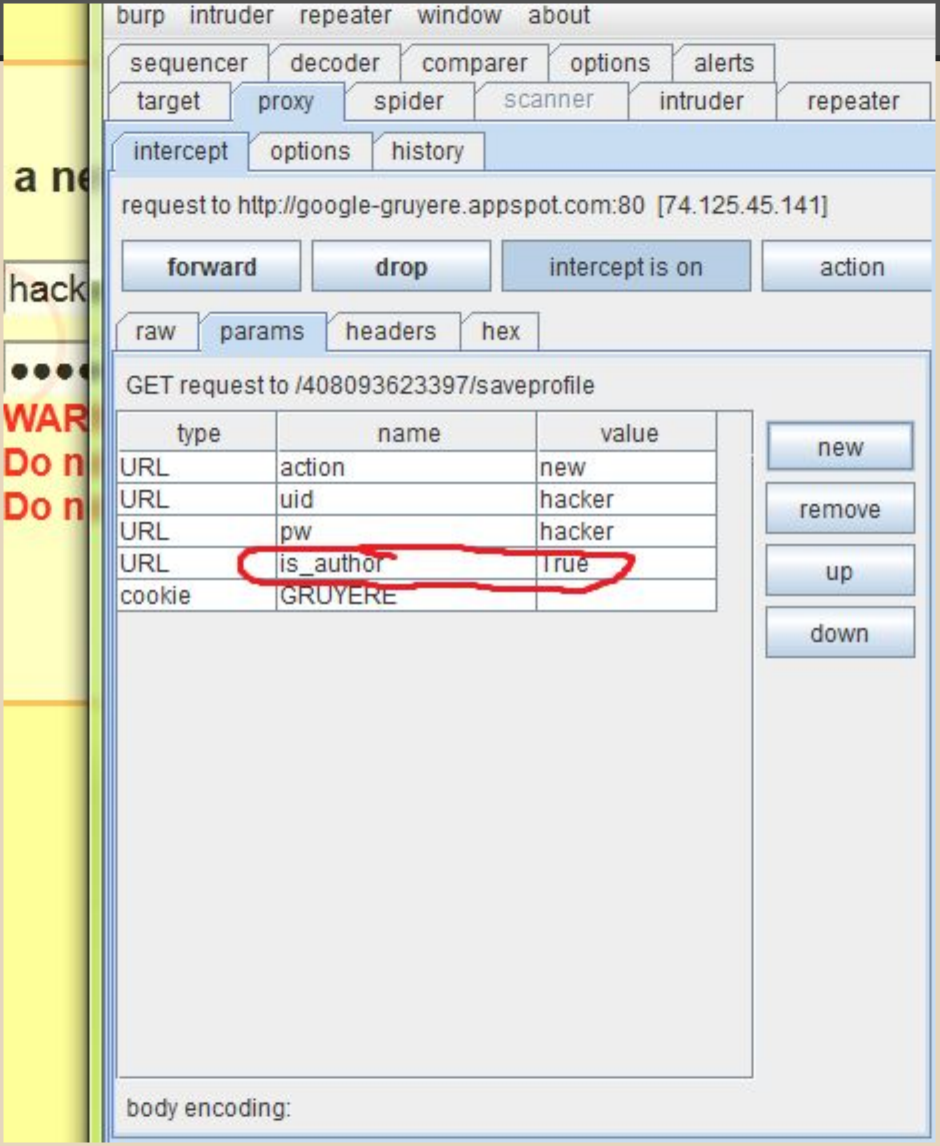
```
GET /408093623397/saveprofile?action=new&uid=hacker&pw=hacker&is_author=True HTTP/1.1
Host: google-gruyere.appspot.com
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:15.0) Gecko/20100101 Firefox/15.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip, deflate
Proxy-Connection: keep-alive
Referer: http://google-gruyere.appspot.com/408093623397/newaccount.gtl
Cookie: GRUYERE=
```

Inspect parameters

Looks like these are stored in the URL...

A common mistake.

Lets try adding a field:
is_admin


a ne
hack
WAR
Do n
Do n
burp intruder repeater window about
sequencer decoder comparer options alerts
target proxy spider scanner intruder repeater
intercept options history
forward
drop
intercept is on
action
raw params headers hex
new
remove
up
down
body encoding:

request to http://google-gruyere.appspot.com:80 [74.125.45.141]

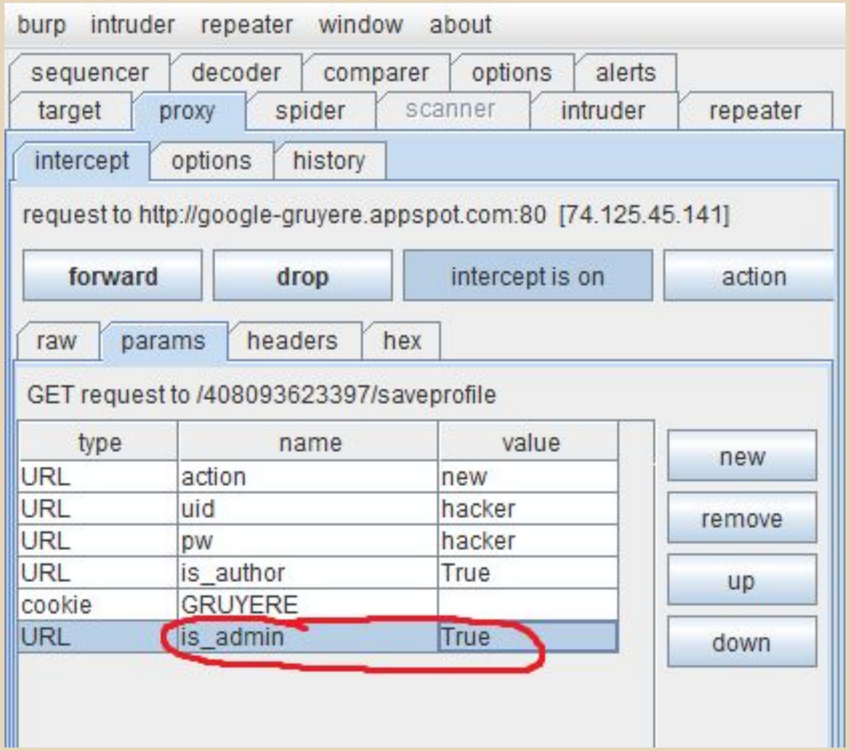
GET request to /408093623397/saveprofile

type	name	value
URL	action	new
URL	uid	hacker
URL	pw	hacker
URL	is_author	True
cookie	GRUYERE	

Manipulate Parameters

add is_admin
and set value: True

Looks good,
now click forward.


burp intruder repeater window about
sequencer decoder comparer options alerts
target proxy spider scanner intruder repeater
intercept options history
forward drop intercept is on action
raw params headers hex
new remove up down

request to http://google-gruyere.appspot.com:80 [74.125.45.141]

GET request to /408093623397/saveprofile

type	name	value
URL	action	new
URL	uid	hacker
URL	pw	hacker
URL	is_author	True
cookie	GRUYERE	
URL	is_admin	True

Cross Site Request Forgery (XSRF)

If a site is vulnerable to XSS, then an attacker can fake any form request as if it came from a victim user. This is XSRF.

"When a browser makes requests to a site, it always sends along any cookies it has for that site, regardless of where the request comes from. "

Lets exploit Gruyere to do some little malicious XSRF. We want an XSRF attack embedded in an image that deletes a victim's posts.

Malicious XSRF demo

Put the following in your icon URL (the gruyere session number matters):
http://google-gruyere.appspot.com/408093623397/deletesnippet?index=0

Now when <u>any</u> user visits the main page and your snippet + icon is visible, whichever snippet he/she posts that is index 0 will be deleted.

Its funny because each time the page is refreshed it deletes a new post on the victim account :D

Cross Site Script Inclusion (XSSI)

"Browsers prevent pages of one domain from reading pages in other domains. But they do not prevent pages of a domain from referencing resources in other domains"

See http://en.wikipedia.org/wiki/Cross-Origin_Resource_Sharing for the underlying security principle that allows this.

"For example, if www.evil.example.com includes a script hosted on www.google.com then that script runs in the evil context not in the google context. So any user data in that script will "leak.""

Denial of Service

A wide class of attacks, which achieve one or more of the following:

Deny

Destroy

Degrade

Deceive/Distract

Disrupt

and Deny

Code Execution / Remote Code Exec.

This is where an attacker can execute arbitrary code remotely on your server.

(Javascript is clientside)

Usually pretty bad and a near-game-over.

Lets do a demo :D

So I want a shell on this server

- Tried a PHP webshell from BackTrack5 R3, but the gruyere request handler disallows php (Unrecognized file type).
  - The file actually got uploaded, but the gruyere front end wont serve the request.
- Instead: python webshell, with a html front end. After uploading about 6 files, the result is the following *almost working* back door :D
  - http://google-gruyere.appspot.com/408093623397/hacker/templates/shell.html

Shell source code

http://code.google.com/p/google-app-engine-samples/downloads/detail?name=shell_20091112.tar.gz&can=2&q=

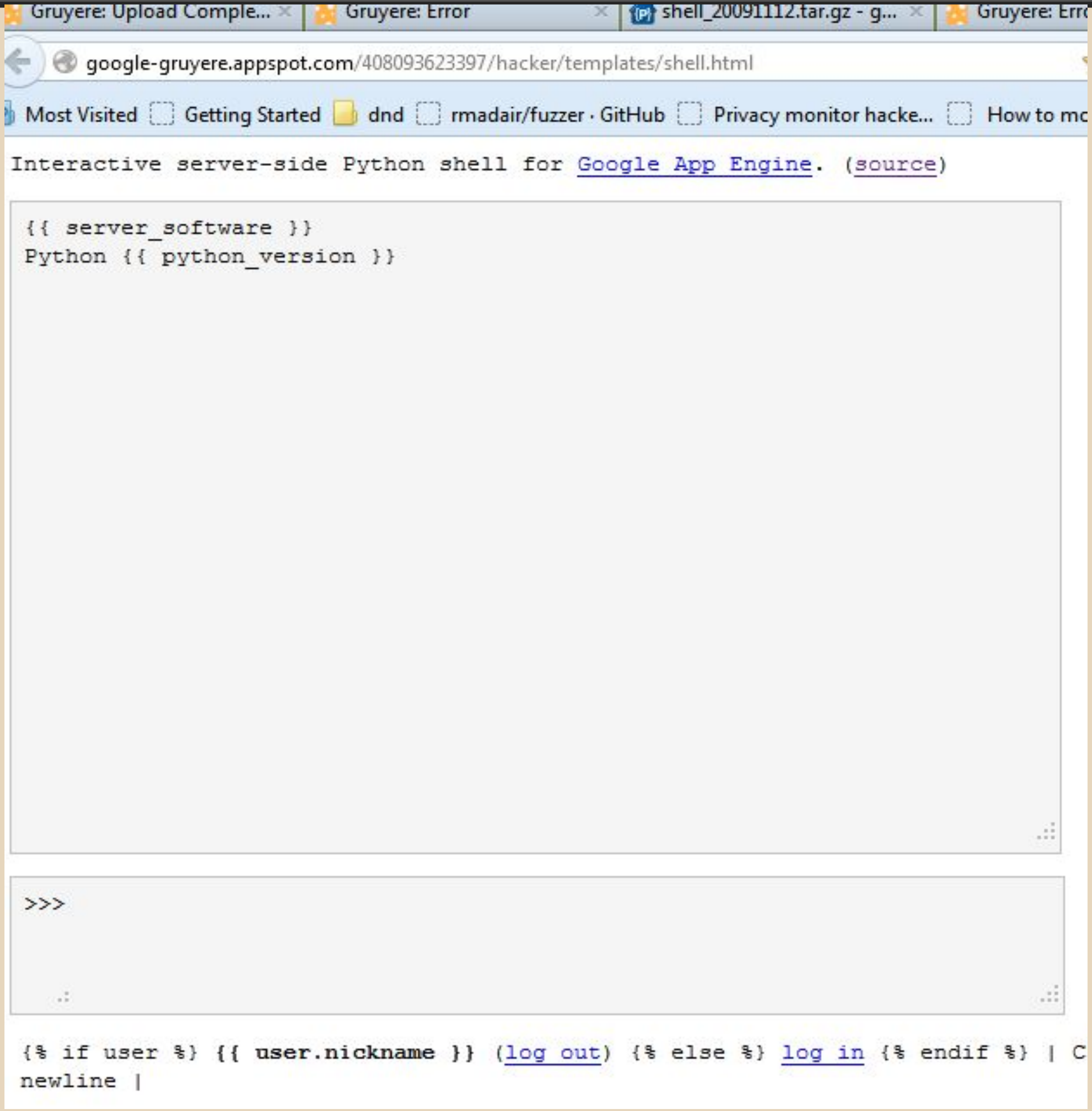
maybe you can get it or another shell working

Ok we've almost got a user-level shell on the server

Didn't have time to figure out what's keeping it from working. But its close.

We'll have another workshop on shellcode later:)

So stay tuned



Gruyere: Upload Comple...
Gruyere: Error
shell_20091112.tar.gz - g...
Gruyere: Err
google-gruyere.appspot.com/408093623397/hacker/templates/shell.html
Most Visited
Getting Started
dnd
rmadair/fuzzer · GitHub
Privacy monitor hacke...
How to mo

Interactive server-side Python shell for Google App Engine. (source)

```
{{ server_software }}
Python {{ python_version }}
```

```
>>>
```

```
{% if user %} {{ user.nickname }} (log out) {% else %} log in {% endif %} | C
newline |
```

Thats all for this time

Questions, comments.