Introduction to OWASP ZAP

Overview

This lab walks you through using ZAP by OWASP. ZAP is a vulnerability analysis tool used to scan Web applications for possible software flaws. As an introduction to using ZAP, you will scan and interrupt http protocols in PHP code we developed in week 4. You will also run the attack scanner on code you developed in week 4.

Important: Do not attempt to use these tools against any live Web site. It is illegal to do so. You can only scan sites you have written permission to scan. You should use the virtual machine on applications you developed running on the localhost and disconnect from the Internet when running ZAP.

Learning Outcomes:

At the completion of the lab you should be able to:

1. Launch ZAP and view Web sites history and input parameters
2. Use ZAP to intercept http messages and change their content to Identify possible vulnerabilities
3. Read and analyze reports produced from ZAP and prioritize and fix alerts associated with software issues

Lab Submission Requirements:

After completing this lab, you will submit a word (or PDF) document that meets all of the requirements in the description at the end of this document. In addition, your associated files should be submitted. You can submit multiple files in a zip file.

Virtual Machine Account Information

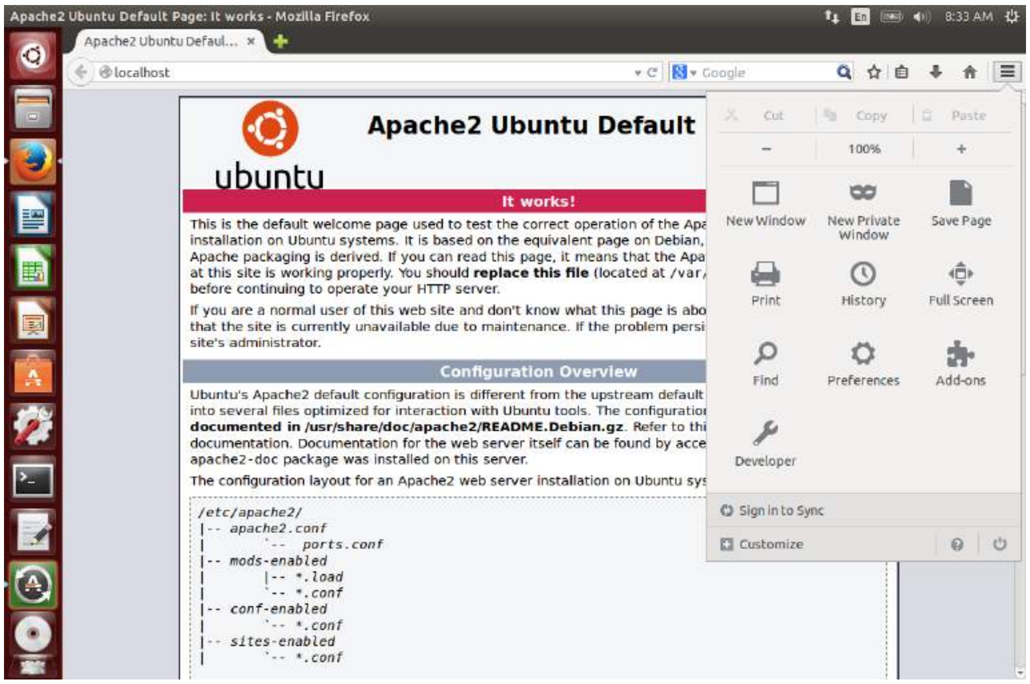
Your Virtual Machine has been preconfigured with all of the software you will need for this class. The default username and password are:

Username :	**umucsdev**
Password:	**umuc$d8v**

Part 1 – Launch ZAP and view Web sites history and input parameters

This exercise will walk you through Launching ZAP and allow you to become comfortable with the GUI for ZAP within your virtual machine. We will use ZAP to begin to analyze some of the PHP Web applications we created in week 4.

1. Important: Disconnect your machine from the Internet when using ZAP. This will ensure you are only scanning applications on your localhost. You can do this by unplugging the Internet cable on your machine or if you have Wireless connection, you should turn your Wi-Fi off.
2. Start your Virtual Machine as you would normally by clicking the Oracle VM VirtualBox, starting the SDEV32Bit image and then logging into the machine.
3. Verify your Firefox browser has the Proxy properly configured. To do this, launch Firefox and go to Options -> Preferences menu.



Apache2 Ubuntu Default

ubuntu

It works!

This is the default welcome page used to test the correct operation of the Apa
installation on Ubuntu systems. It is based on the equivalent page on Debian,
Apache packaging is derived. If you can read this page, it means that the Apa
at this site is working properly. You should **replace this file** (located at /var/
before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is abo
that the site is currently unavailable due to maintenance. If the problem persi
site's administrator.

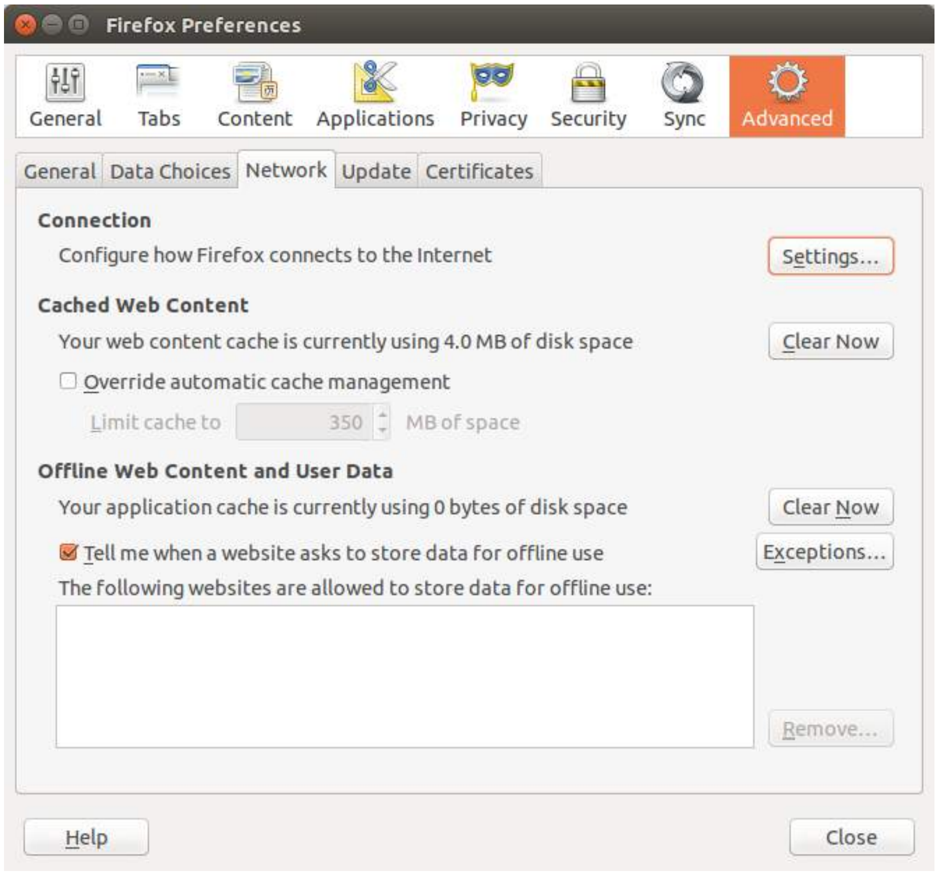
Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default
into several files optimized for interaction with Ubuntu tools. The configuratio
documented in /usr/share/doc/apache2/README.Debian.gz. Refer to thi
documentation. Documentation for the web server itself can be found by acce
apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu sys

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

4. Click on the Advanced icon and select the Network tab.



Connection

Configure how Firefox connects to the Internet [Settings...]

Cached Web Content

Your web content cache is currently using 4.0 MB of disk space [Clear Now]

☐ Override automatic cache management

Limit cache to 350 MB of space

Offline Web Content and User Data

Your application cache is currently using 0 bytes of disk space [Clear Now]

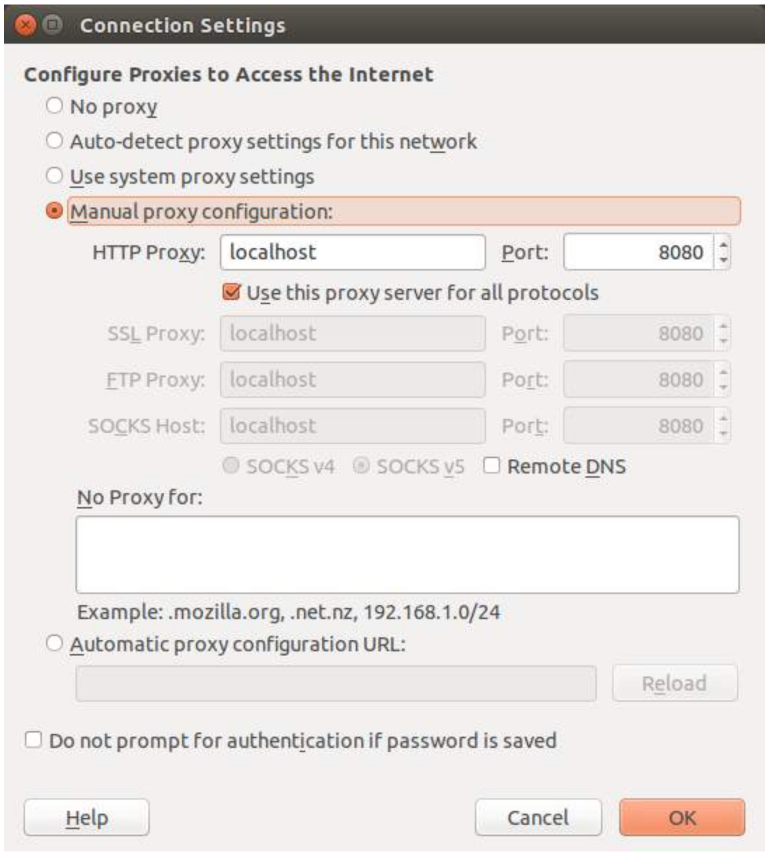
☑ Tell me when a website asks to store data for offline use [Exceptions...]

The following websites are allowed to store data for offline use:

[Remove...]

[Help] [Close]

5. Select the Settings button and then enter localhost for the HTTP Proxy with Port of 8080. Also, be sure Use this proxy server for all protocols is selected and the No proxy for: textarea is blank.



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: localhost Port: 8080

FTP Proxy: localhost Port: 8080

SOCKS Host: localhost Port: 8080

○ SOCKS v4 ● SOCKS v5 ☐ Remote DNS

No Proxy for:

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

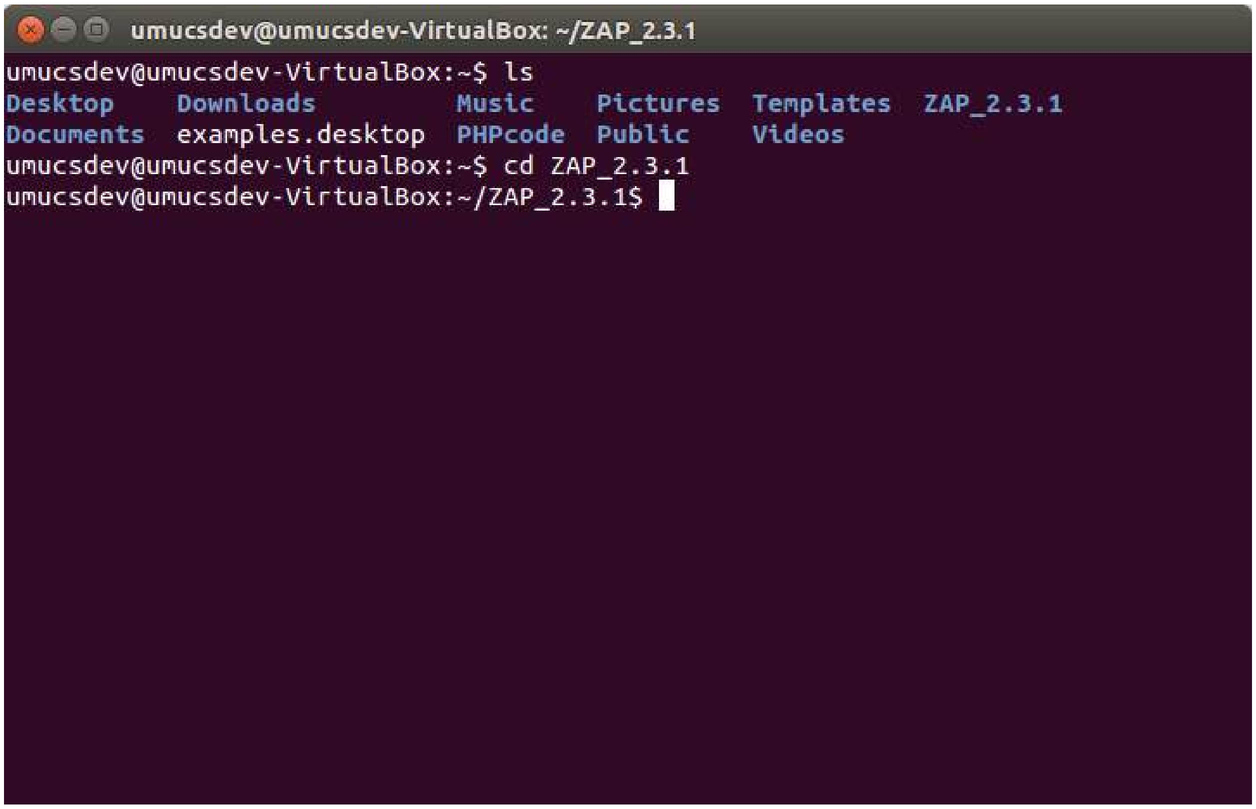
○ Automatic proxy configuration URL:

Reload

☐ Do not prompt for authentication if password is saved

6. Click Ok and your Browser will send HTTP messages to the already preconfigured ZAP proxy. Note: You will need to change your Browser settings back to “No proxy” once you are ready to perform normal Browsing to either the localhost or the Internet for the future.

7. To Launch ZAP, open up a shell prompt and change to the ZAP_2.3.1 folder.



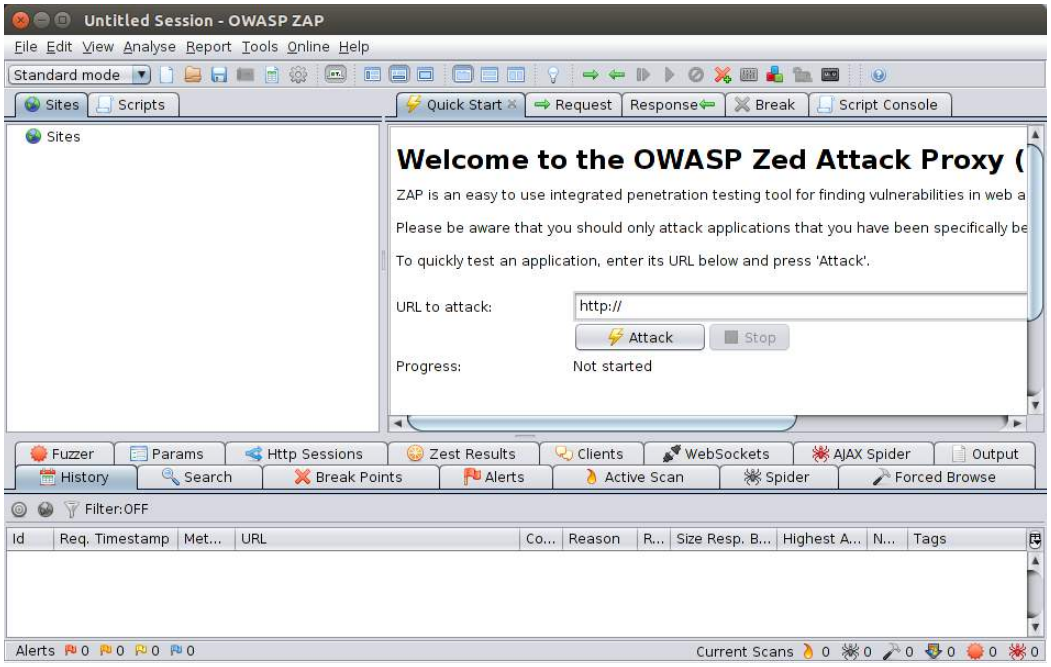
```
umucsdev@umucsdev-VirtualBox: ~/ZAP_2.3.1
umucsdev@umucsdev-VirtualBox:~$ ls
Desktop     Downloads         Music     Pictures   Templates  ZAP_2.3.1
Documents   examples.desktop  PHPcode   Public     Videos
umucsdev@umucsdev-VirtualBox:~$ cd ZAP_2.3.1
umucsdev@umucsdev-VirtualBox:~/ZAP_2.3.1$
```

8. Type the following command to start ZAP:

```
./zap.sh
```

The ./ characters are important for properly and securely launching programs in Linux.

After launching the ZAP GUI will be revealed.



Welcome to the OWASP Zed Attack Proxy (

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web a

Please be aware that you should only attack applications that you have been specifically be

To quickly test an application, enter its URL below and press 'Attack'.

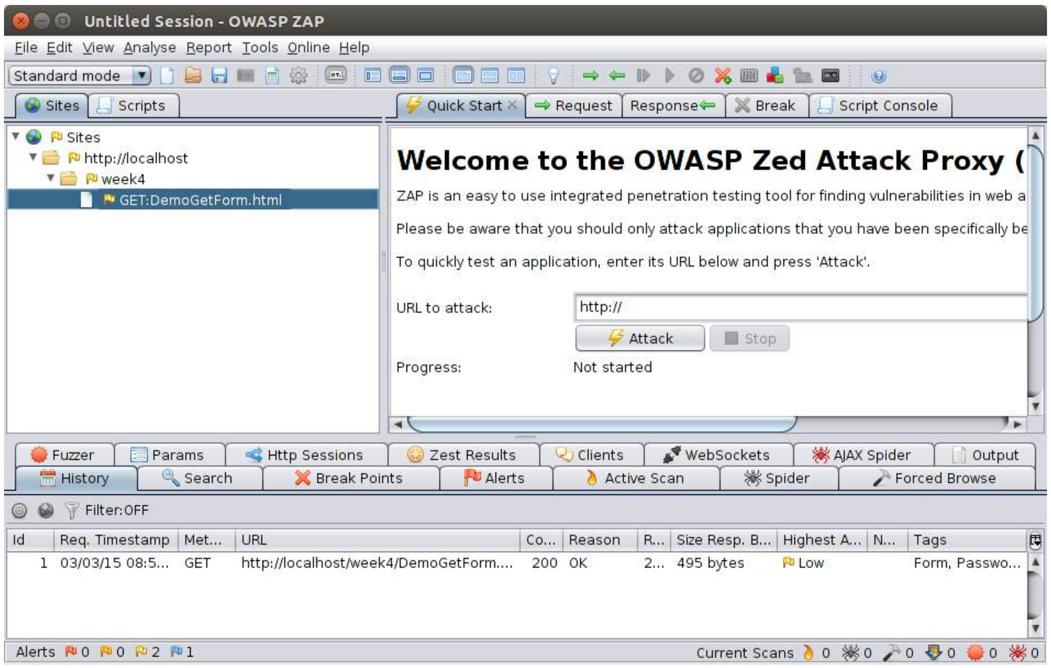
URL to attack: http://

Attack Stop

Progress: Not started

9. The GUI will look very complicated at first as there is much functionality that this tool can perform. Over the next couple of weeks you will be exposed to most of the critical features.

10. To use the Sites and History tabs, you will need to launch your Browser. Open your Browser and launch the DemoGetForm.html file from week 4. The URL should be: localhost/week4/DemoGetForm.html. If you review the ZAP tool you will see the Sites Icon has listed the GET:DemoGetForm.html file. Also, notice the History tab as information related to this site.



Welcome to the OWASP Zed Attack Proxy (

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web a

Please be aware that you should only attack applications that you have been specifically be

To quickly test an application, enter its URL below and press 'Attack'.

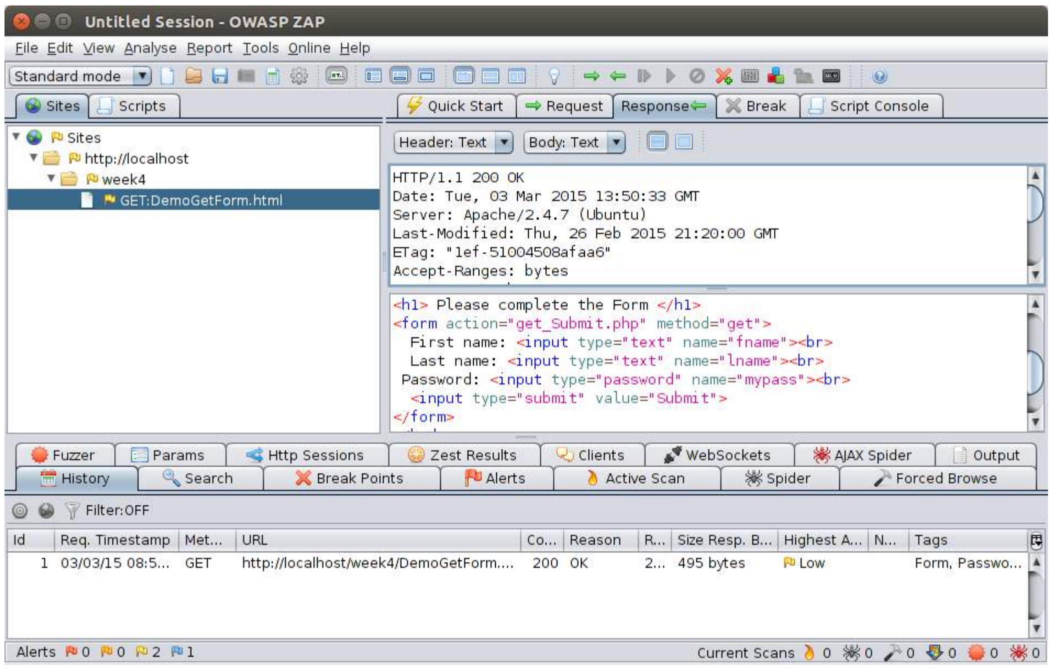
URL to attack: http://

Attack Stop

Progress: Not started

Id	Req. Timestamp	Met...	URL	Co...	Reason	R...	Size Resp. B...	Highest A...	N...	Tags
1	03/03/15 08:5...	GET	http://localhost/week4/DemoGetForm....	200	OK	2...	495 bytes	Low		Form, Passwo...

11. If you click on the Request and Response tabs on the right site of the tool, you will see both the header and body of the DemoGetForm.html file.



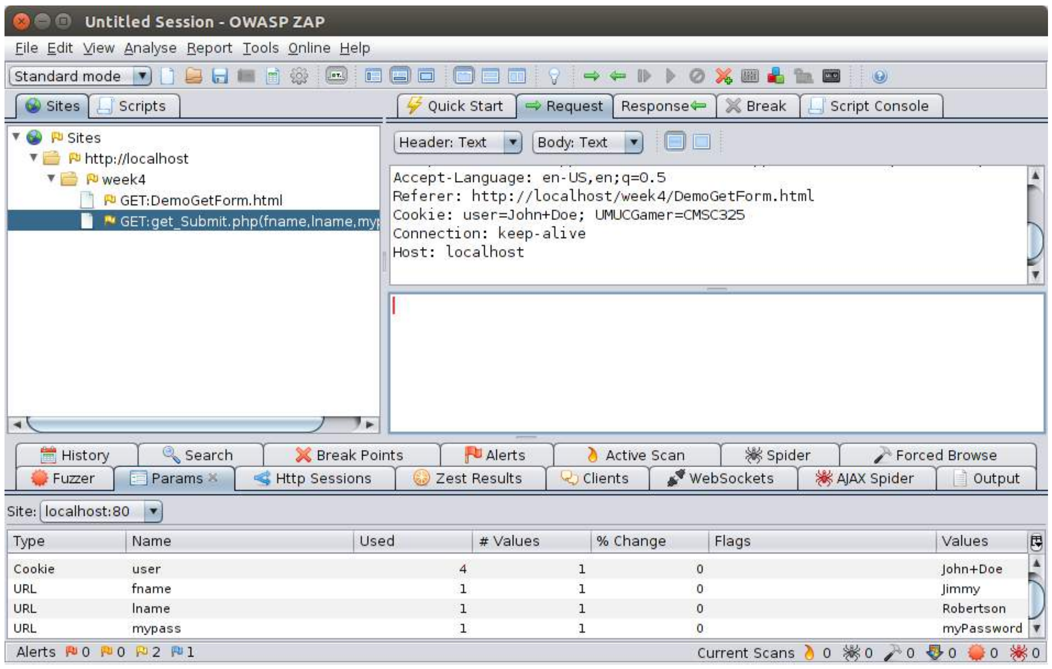
```
HTTP/1.1 200 OK
Date: Tue, 03 Mar 2015 13:50:33 GMT
Server: Apache/2.4.7 (Ubuntu)
Last-Modified: Thu, 26 Feb 2015 21:20:00 GMT
ETag: "1ef-51004508afaa6"
Accept-Ranges: bytes
```

```
<h1> Please complete the Form </h1>
<form action="get_Submit.php" method="get">
  First name: <input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
 Password: <input type="password" name="mypass"><br>
  <input type="submit" value="Submit">
</form>
```

Id	Req. Timestamp	Met...	URL	Co...	Reason	R...	Size Resp. B...	Highest A...	N...	Tags
1	03/03/15 08:5...	GET	http://localhost/week4/DemoGetForm....	200	OK	2...	495 bytes	Low		Form, Passwo...

12. You should carefully review the request and response details noting how much information is revealed from this transaction. Information such as Web Server and Operating system are revealed. All HTML source code and comments are also provided.

13. Complete the DemoGetForm.html and notice the addition of the get_Submit.php(fname,lname mypass) to the history and sites tab. Also, notice the Request and Response tabs as well the Params tab near the bottom.


Untitled Session - OWASP ZAP
File Edit View Analyse Report Tools Online Help
Standard mode
Sites Scripts
Quick Start Request Response Break Script Console
Sites
http://localhost
week4
GET:DemoGetForm.html
GET:get_Submit.php(fname,lname,my
Header: Text Body: Text
History Search Break Points Alerts Active Scan Spider Forced Browse
Fuzzer Params Http Sessions Zest Results Clients WebSockets AJAX Spider Output
Site: localhost:80
Alerts 0 0 2 1
Current Scans 0 0 0 0 0 0

```
Accept-Language: en-US,en;q=0.5
Referer: http://localhost/week4/DemoGetForm.html
Cookie: user=John+Doe; UMUCGamer=CMSC325
Connection: keep-alive
Host: localhost
```

Type	Name	Used	# Values	% Change	Flags	Values
Cookie	user	4	1	0		John+Doe
URL	fname	1	1	0		Jimmy
URL	lname	1	1	0		Robertson
URL	mypass	1	1	0		myPassword

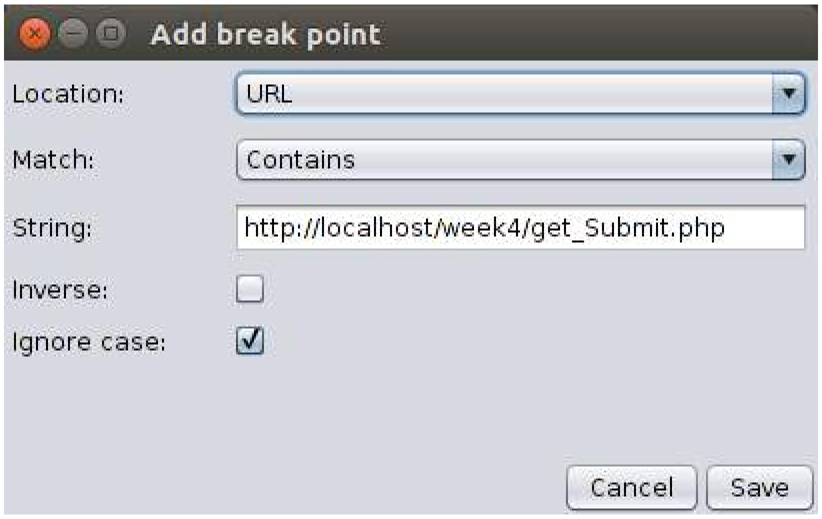
You should experiment with using ZAP by running the other Web applications you created in week 4. As you run these applications, be sure to review the Sites, History, Request, Response and Params tabs.

Part 2 Use ZAP to intercept http messages and change their content to Identify possible vulnerabilities

One of the strong features of the ZAP tool is the ability to interrupt HTTP message and change the values in an attempt to find software flaws. In this exercise, we will use the week4 PHP web applications and demonstrate how the input parameters sent from the form can easily be changed and redirected back to the application with the new parameters.

To interrupt an HTTP transmission, you use the Break Points functionality within ZAP.

1. As before, be certain you have disconnected your machine from the Internet, launch your virtual machine and then start ZAP.
2. To set a Break point, right mouse click the specific site you want to Break on. For example, if we want to put a Break point for the get_Submit.php file, we would right mouse click that site, select break and then select save.



Add break point

Location: URL

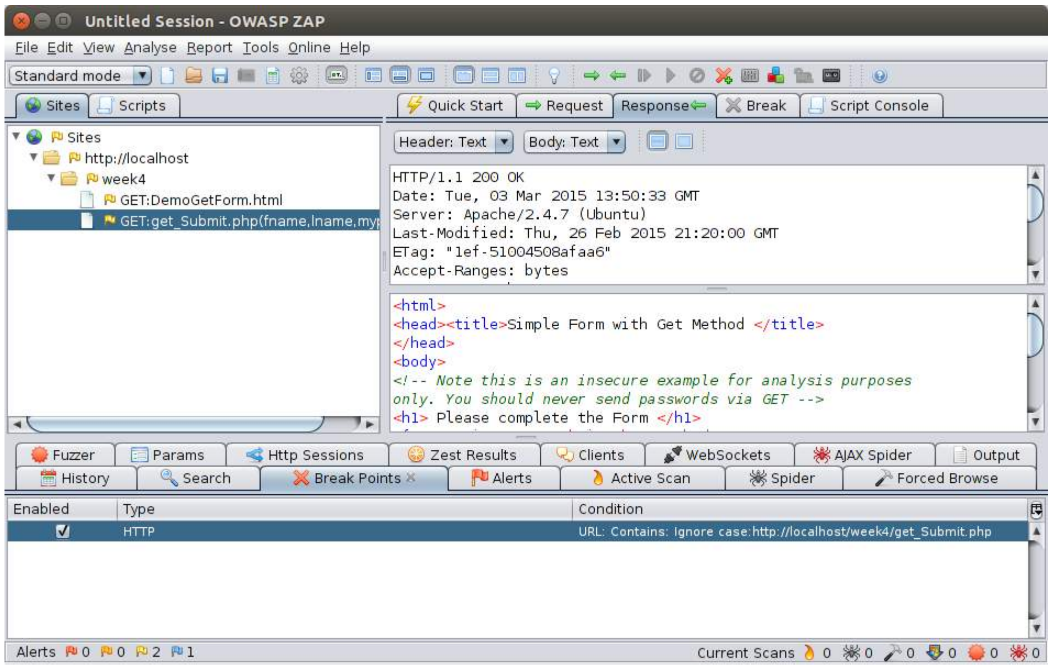
Match: Contains

String: http://localhost/week4/get_Submit.php

Inverse: ☐

Ignore case: ☑

3. Once the Break Point is set, it will display in the Break Points tab.



Untitled Session - OWASP ZAP

Sites
- http://localhost
  - week4
    - GET:DemoGetForm.html
    - GET:get_Submit.php(fname,lname,my

Header: Text | Body: Text

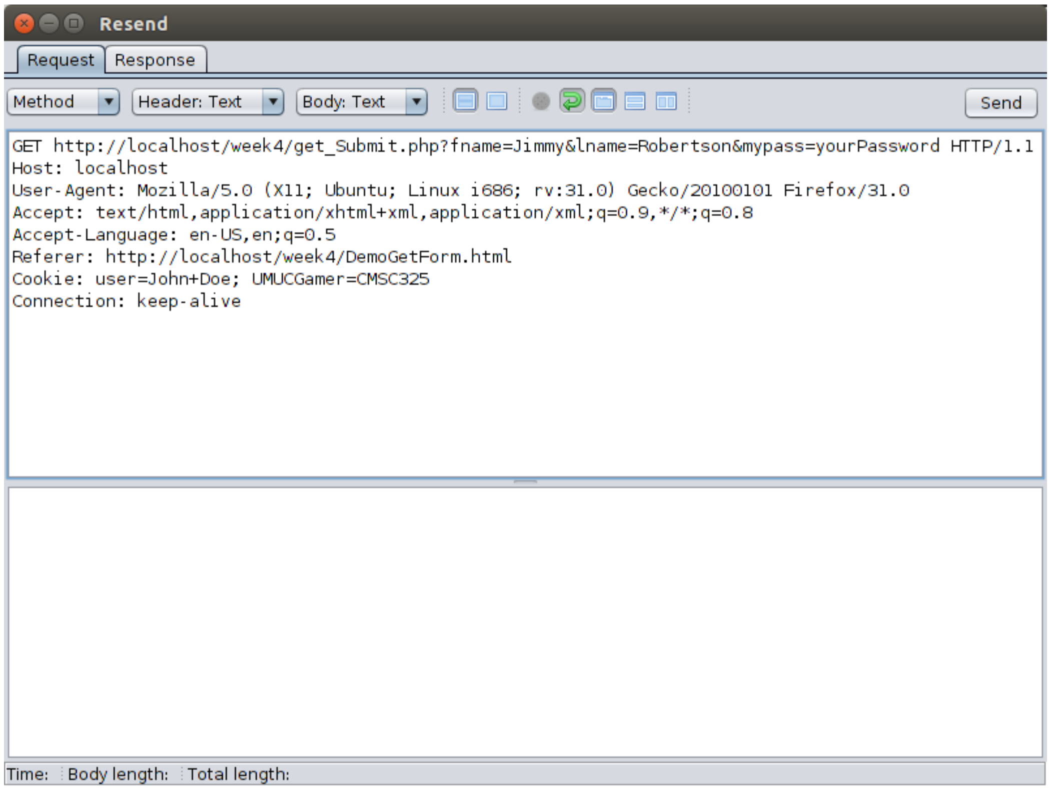
```
HTTP/1.1 200 OK
Date: Tue, 03 Mar 2015 13:50:33 GMT
Server: Apache/2.4.7 (Ubuntu)
Last-Modified: Thu, 26 Feb 2015 21:20:00 GMT
ETag: "1ef-51004508afaa6"
Accept-Ranges: bytes
```

```
<html>
<head><title>Simple Form with Get Method </title>
</head>
<body>
<!-- Note this is an insecure example for analysis purposes
only. You should never send passwords via GET -->
<h1> Please complete the Form </h1>
```

Enabled	Type	Condition
✓	HTTP	URL: Contains: Ignore case:http://localhost/week4/get_Submit.php

4. To test the break point, launch the DemoGetForm.html application, complete the form and then press submit. You will notice the browser seems to hang as the http request has been intercepted by the proxy and is awaiting action in ZAP.

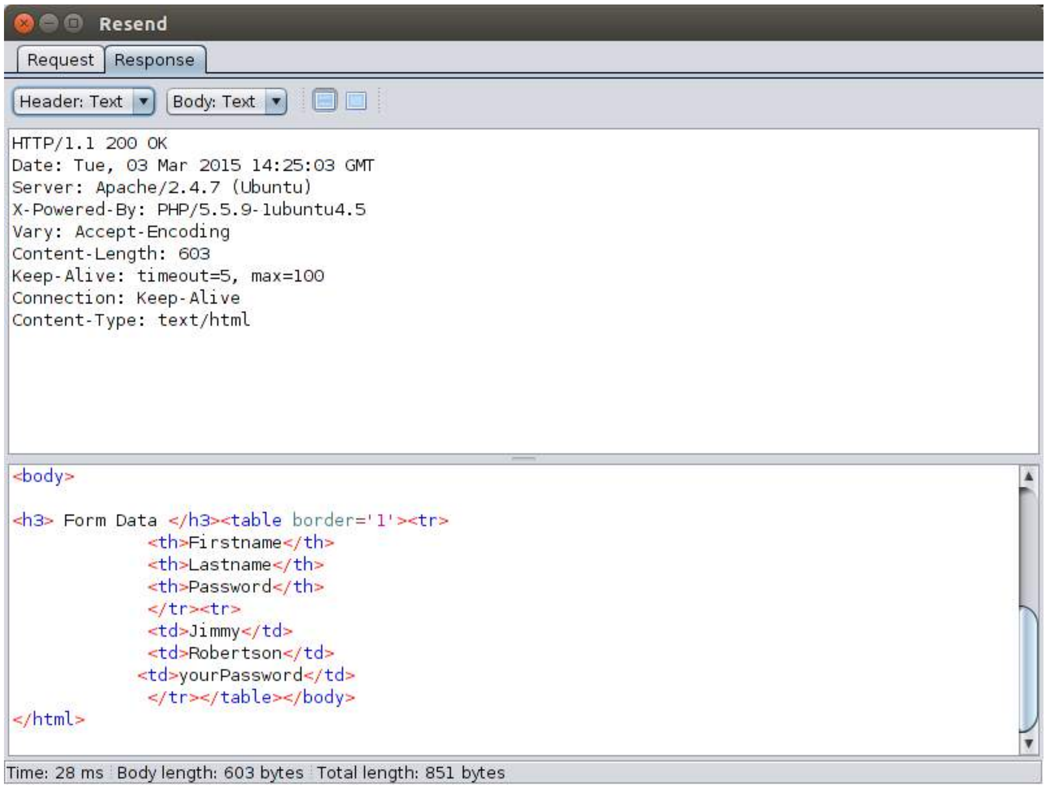
5. You can now modify the original data by right mouse clicking in the Break tab and selecting resend. When the resend frame appears, modify the GET string as appropriate and select the send button.



```
GET http://localhost/week4/get_Submit.php?fname=Jimmy&lname=Robertson&mypass=yourPassword HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://localhost/week4/DemoGetForm.html
Cookie: user=John+Doe; UMUCGamer=CMSC325
Connection: keep-alive
```

Time: Body length: Total length:

6. You will then be able to see the new response in the window.



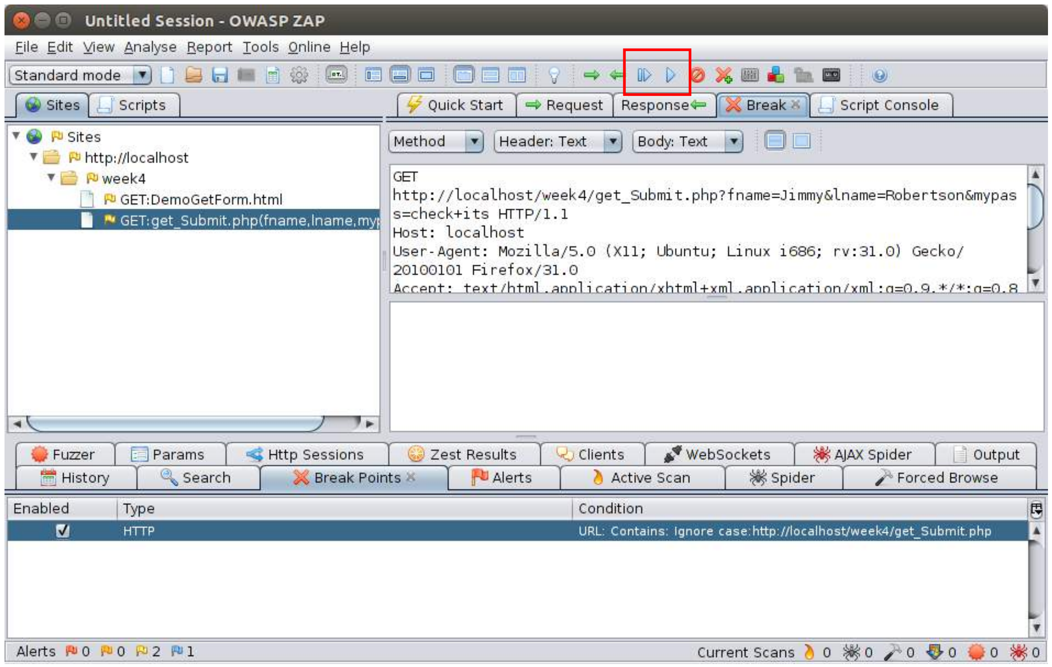
```
HTTP/1.1 200 OK
Date: Tue, 03 Mar 2015 14:25:03 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.5
Vary: Accept-Encoding
Content-Length: 603
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

```
<body>

<h3> Form Data </h3><table border='1'><tr>
             <th>Firstname</th>
             <th>Lastname</th>
             <th>Password</th>
             </tr><tr>
             <td>Jimmy</td>
             <td>Robertson</td>
            <td>yourPassword</td>
             </tr></table></body>
</html>
```

Time: 28 ms Body length: 603 bytes Total length: 851 bytes

7. To release the break point, you can click on the arrows at the top which allow you to Submit and step to the next break. Once you select those arrows, the data will be submitted and the browser will show the results. Note, you can also modify the text in the break tab itself and then click the submit arrows to show the results in your browser.



```
GET
http://localhost/week4/get_Submit.php?fname=Jimmy&lname=Robertson&mypas
s=check+its HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:31.0) Gecko/
20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

Enabled	Type	Condition
✓	HTTP	URL: Contains: Ignore case:http://localhost/week4/get_Submit.php

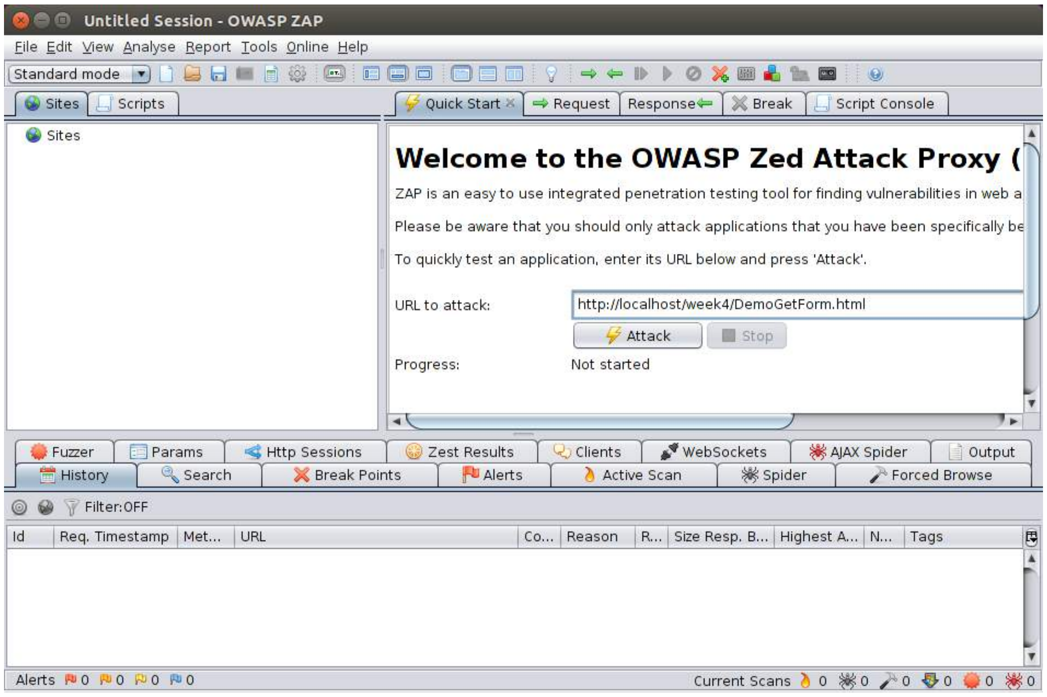
This is where the person-in-the-loop analysis and discovery takes place. Manipulating the data sent into the applications provides insight into the vulnerabilities of the application. For example, you may discover that sending in a null password allows access to the system. You may also discover providing an admin username and brute force password guess may provide additional system privileges.

You should experiment with all of the week4 PHP applications to see how parameters and information can be changed and the resulting impact on the application.

Part 3 Read and analyze reports produced from ZAP

In this lab, we will run the automatic scanning feature of ZAP and then generate HTML Alert reports for the DemoGetForm.html and DemoPostForm.html and discuss approaches to prioritize and mitigate the issues found in each Web applications.

1. As before, be certain you have disconnected your machine from the Internet, launch your virtual machine and then start ZAP.
2. Under the Quick Start tab of ZAP, enter the localhost/week4/DemoGetForm.html URL and click attack.


Untitled Session - OWASP ZAP
File Edit View Analyse Report Tools Online Help
Standard mode
Sites Scripts
Quick Start Request Response Break Script Console
Sites
Fuzzer Params Http Sessions Zest Results Clients WebSockets AJAX Spider Output
History Search Break Points Alerts Active Scan Spider Forced Browse
Filter:OFF
Id Req. Timestamp Met... URL Co... Reason R... Size Resp. B... Highest A... N... Tags
Alerts 0 0 0 0
Current Scans 0 0 0 0 0 0

Welcome to the OWASP Zed Attack Proxy (

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web a

Please be aware that you should only attack applications that you have been specifically be

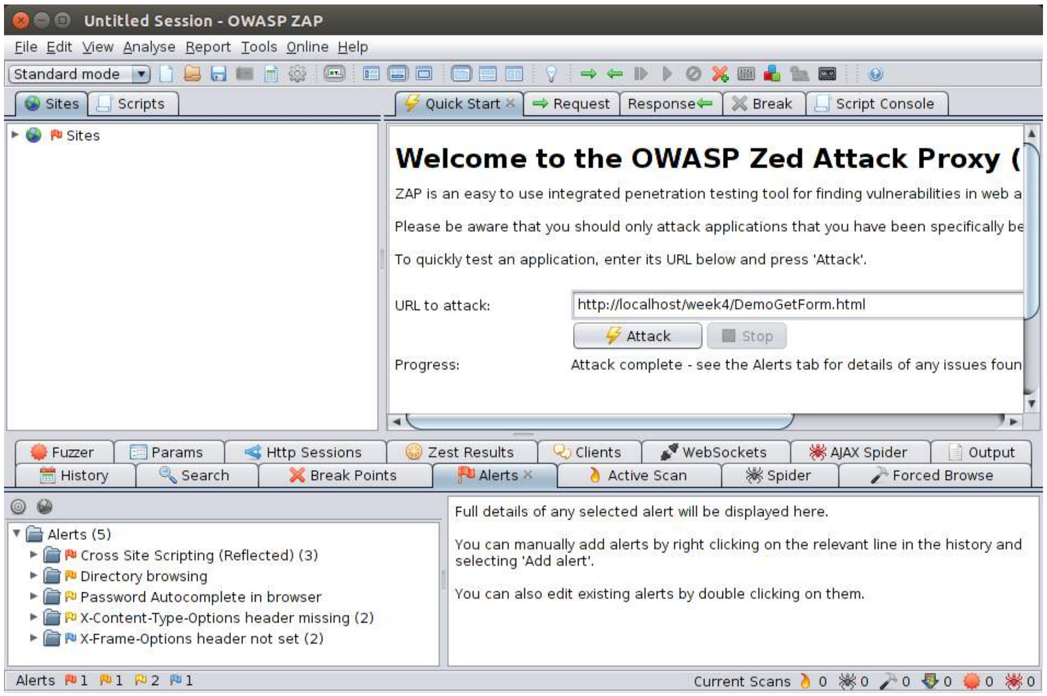
To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://localhost/week4/DemoGetForm.html

Attack Stop

Progress: Not started

3. As the scan runs, you will see hundreds of requests logged into the Active Scan tab. You will also see several Alerts. Alerts from the scan provide possible vulnerabilities. The color of flag indicates the risk level of issue found.


Untitled Session - OWASP ZAP
File Edit View Analyse Report Tools Online Help
Standard mode
Sites Scripts
Quick Start Request Response Break Script Console
Sites
Fuzzer Params Http Sessions Zest Results Clients WebSockets AJAX Spider Output
History Search Break Points Alerts Active Scan Spider Forced Browse
Alerts 1 1 2 1
Current Scans 0 0 0 0 0 0

Welcome to the OWASP Zed Attack Proxy (

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web a

Please be aware that you should only attack applications that you have been specifically be

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://localhost/week4/DemoGetForm.html

Attack Stop

Progress: Attack complete - see the Alerts tab for details of any issues foun

Alerts (5)
- Cross Site Scripting (Reflected) (3)
- Directory browsing
- Password Autocomplete in browser
- X-Content-Type-Options header missing (2)
- X-Frame-Options header not set (2)

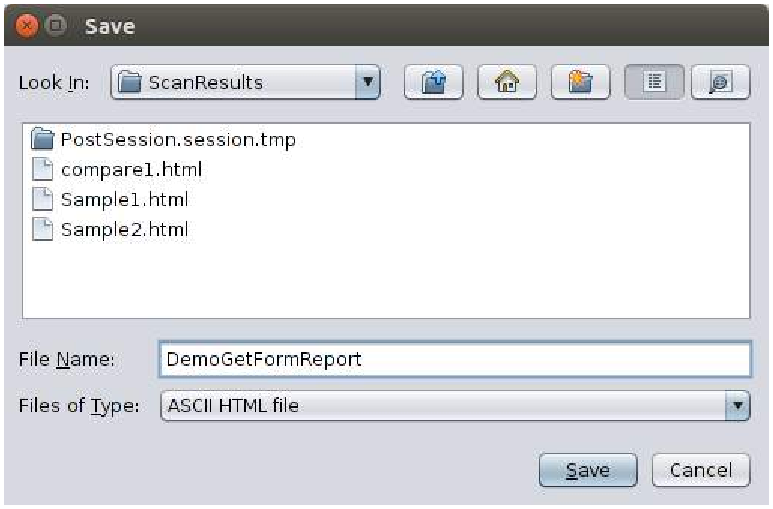
Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

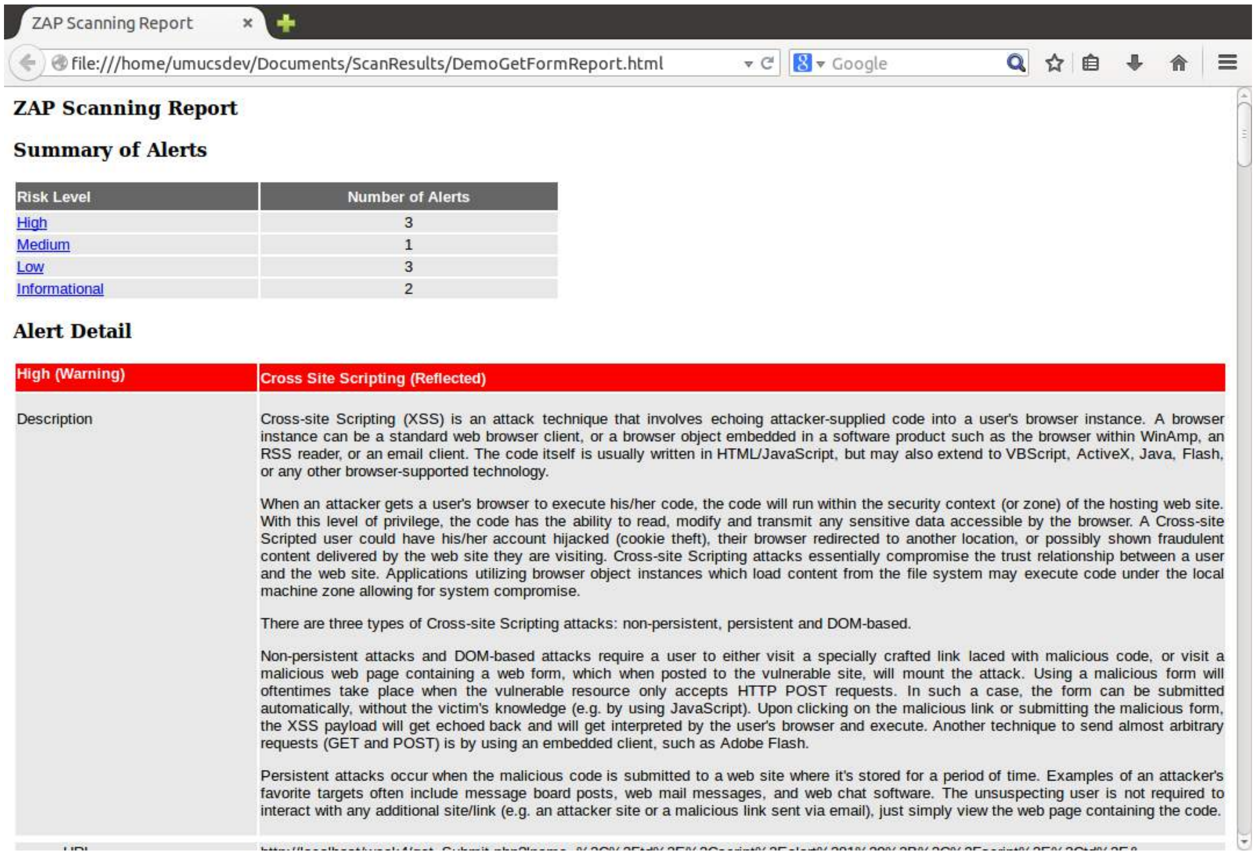
4. To generate a report from the scan in HTML format, open the Report menu from the top of tool bar and select Generate HTML report.

5. Save the report to a folder and filename of your choice. For this example, a ScanResults folder was created and the HTML report was saved to the DemoGetFormReport file. Click Save to continue.



File Name: DemoGetFormReport
Files of Type: ASCII HTML file

6. The HTML report should automatically open in your Browser. If not, use the file manager to open the report.


ZAP Scanning Report
file:///home/umucsdev/Documents/ScanResults/DemoGetFormReport.html
Google

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	3
Medium	1
Low	3
Informational	2

Alert Detail

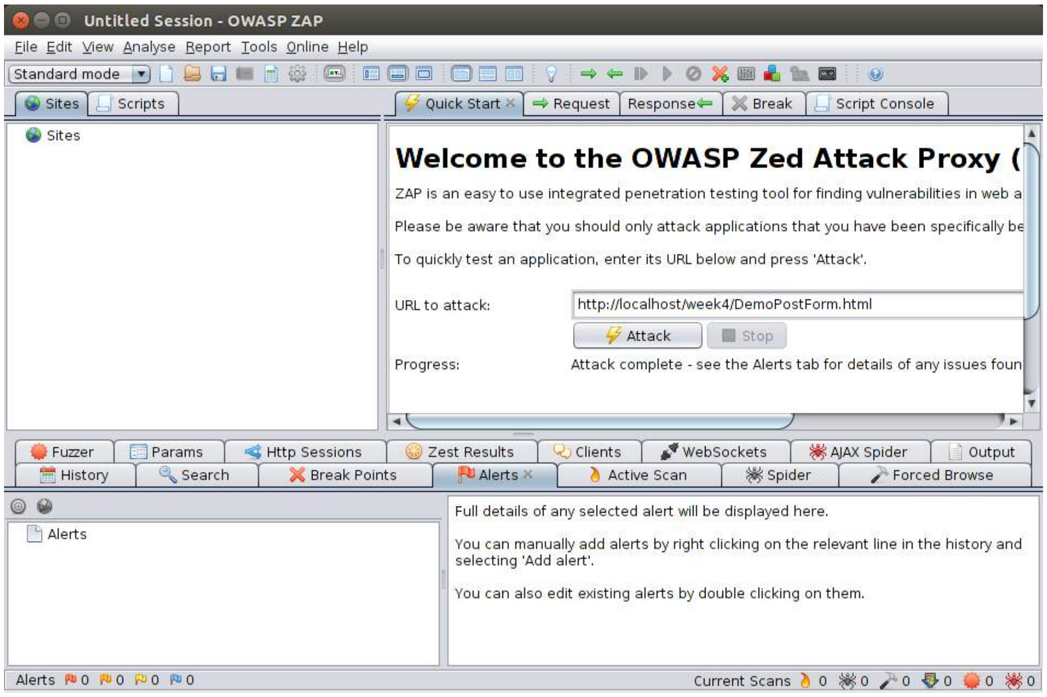
High (Warning)	Cross Site Scripting (Reflected)
Description	Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise. There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based. Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash. Persistent attacks occur when the malicious code is submitted to a web site where it's stored for a period of time. Examples of an attacker's favorite targets often include message board posts, web mail messages, and web chat software. The unsuspecting user is not required to interact with any additional site/link (e.g. an attacker site or a malicious link sent via email), just simply view the web page containing the code.

7. You should review the report carefully comparing the risk levels, descriptions, URL, Parameter, Attack, Solution, Reference CWE ID and WASC ID. When you prepare your lab for this week, you will use each of fields to describe your findings as well as formulate your solutions. The solutions provided may work but you may need to dig deeper into the references to mitigate the issue. Typically, you should work to mitigate the higher risk issues first.

8. After you have resolved the issues, you can rescan the application and see if any Alerts remain. You will find some of the warnings are easy to fix. Some are more challenging – particularly the Cross-Site Scripting issues.

9. You can use the File-> New Session option to clear the current session.

10. Next, run a similar scan and analysis of the week4/DemoPostForm.html application. Save the results to a file named DemoPostSubmitReport.



Sites

Welcome to the OWASP Zed Attack Proxy (

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web a

Please be aware that you should only attack applications that you have been specifically be

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://localhost/week4/DemoPostForm.html

Progress: Attack complete - see the Alerts tab for details of any issues foun

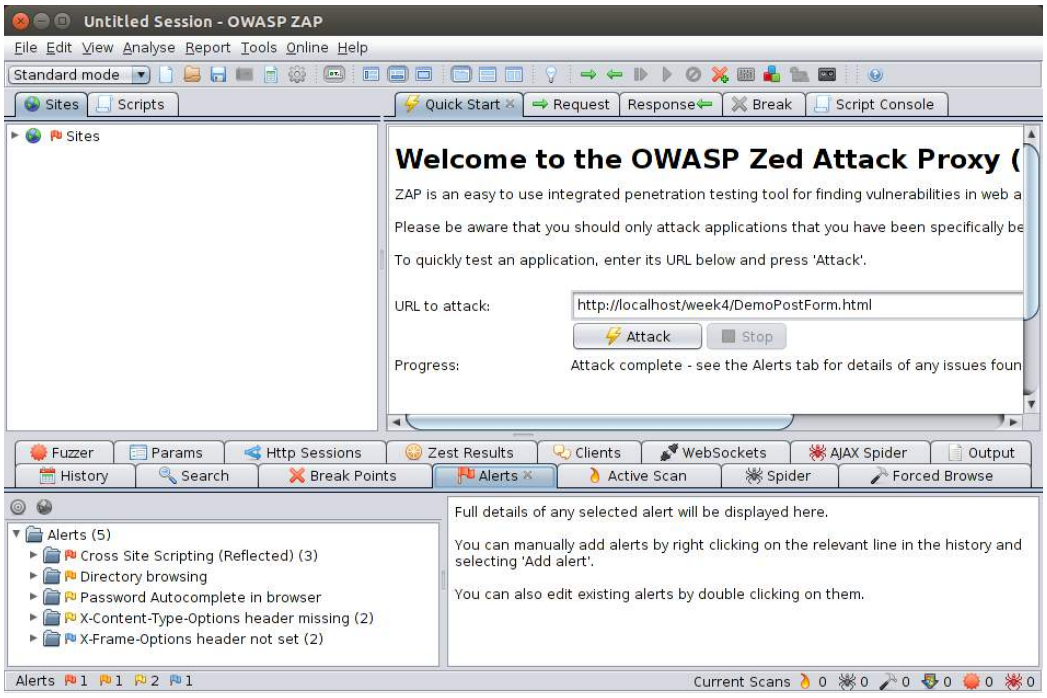
Alerts

Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.

You will find similar number and types of alerts for this web application:


Untitled Session - OWASP ZAP

Welcome to the OWASP Zed Attack Proxy (

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in web a

Please be aware that you should only attack applications that you have been specifically be

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://localhost/week4/DemoPostForm.html

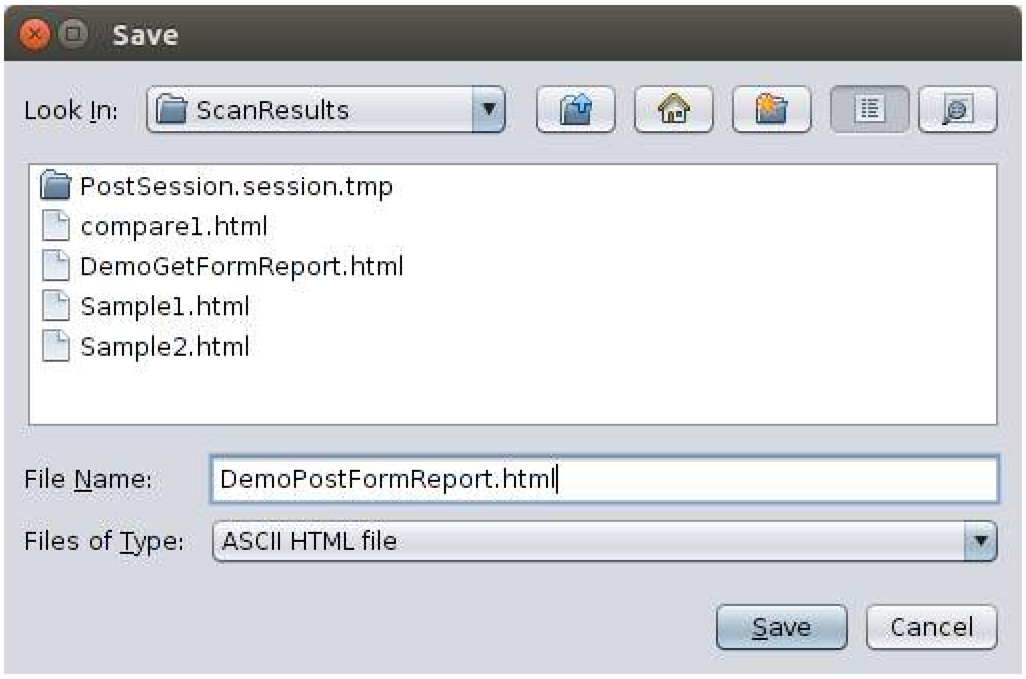
Progress: Attack complete - see the Alerts tab for details of any issues foun

Alerts (5)
- Cross Site Scripting (Reflected) (3)
- Directory browsing
- Password Autocomplete in browser
- X-Content-Type-Options header missing (2)
- X-Frame-Options header not set (2)

Full details of any selected alert will be displayed here.

You can manually add alerts by right clicking on the relevant line in the history and selecting 'Add alert'.

You can also edit existing alerts by double clicking on them.


Save

Look In: ScanResults

- PostSession.session.tmp
- compare1.html
- DemoGetFormReport.html
- Sample1.html
- Sample2.html

File Name: DemoPostFormReport.html

Files of Type: ASCII HTML file

11. Work through the report to analyze the results and remove alerts where possible.

The analysis and mitigation of issues is a repetitive process that should be done often in development and after each release to make sure issues are not introduced during updates. ZAP is just one tool for use in this process. Be aware that ZAP is not the only approach for finding software vulnerabilities nor will it eliminate or find all of the issues. Other scanners and techniques should be used to help secure your web application.

Lab submission details:

For this lab, you will provide a detailed analysis using both manual interception techniques and automatic scanner attacks on the http://localhost/week4/loginAuth.html. You should run the manual interception techniques first, and describe in detail the information revealed to you during your analysis. Be sure to provide screen captures of you running of the tool and analyze all files used for the application (loginAuth.html, authcheck.php and logout.php). Try to modify the http messages and look for possible vulnerabilities. This is the important discovery portion of your analysis.

When you run the automatic scan, be sure to generate an HTML report showing all alerts. Also, describe the active scan activity. For each alert, discuss all of the output and try possible solutions. Be sure to describe how you prioritized alert messages. Try to resolve all alerts and document specifically your process in resolving those alerts. Rerun the scanner after you have fixed as many issues as you can to demonstrate your success.

For your deliverables, you should submit a zip file containing your word document (or PDF file) with screen shots of your scans. Be sure to include the descriptions and analysis of your results, your prioritization and approach to mitigating the issues. Also, include the reports from your scan. Your report should be well-organized and clearly written. This report is aimed at your Chief Security officer who pays your salary. He is a technical geek and wants details, clarity and something he can pass on to others to make sure you have job security for years to come.

Have fun with this!

Include your full name, class number and section and date in the document.

Grading Rubric:

Attribute	**Meets**	**Does not meet**
ZAP analysis	**6 points** Runs the manual interception techniques, and describes in detail the information revealed to you during your analysis. (1 point) Analyzes all files used for the application (loginAuth.html, authcheck.php and logout.php). (1 point) Modifies the http messages and looks for possible vulnerabilities. (1 point)	**0 points** Does not run the manual interception techniques, and describes in detail the information revealed to you during your analysis. Does not analyze all files used for the application (loginAuth.html, authcheck.php and logout.php). Does not modify the http messages or look for possible vulnerabilities.

	Runs the automatic scan, and generates an HTML report showing all alerts. (1 point) Discusses all of the output and tries possible solutions for all alerts. (1 point) Reruns the scanner after you have fixed as many issues as you can to demonstrate your success. (1 point)	Does not run the automatic scan, or generate an HTML report showing all alerts. Does not discuss all of the output and tries possible solutions for all alerts. Does not rerun the scanner after you have fixed as many issues as you can to demonstrate your success.
Documentation and submission	**4 points** Submits a zip file containing your word document (or PDF file) with screen shots of your scans. (1 point) Includes the descriptions and analysis of your results, your prioritization and approach to mitigating the issues. (2 points) Includes the reports from your scan. (0.5 points) Your report is well-organized and clearly written. (0.5 points)	**0 points** Does not submits a zip file containing your word document (or PDF file) with screen shots of your scans. Does not includes the descriptions or analysis of your results, or the prioritization and approach to mitigating the issues. Does not include the reports from your scan. Your report is not well-organized or clearly written.