SE-4930: Developing Secure Software

Lab 9: Penetration testing with OWASP ZAP

Due by 23:59 November 11, 2014

Introduction

ZAP is a tool for aiding in penetration testing. It allows the user to analyze a web system and through a multitude of approaches allows the user to determine the security of a given website.

In this lab, you are going to use ZAP to attack a website and determine its vulnerabilities. You'll be able to determine some of the insecurities of the website as well as automatically scan for significant vulnerabilities.

Step 1 – Downloading and Installing the tools

To start, we will need to metasploitable virtual machine form last week as well as a new tool. This new tool, ZAP, will run under windows. You'll need to download the installer at https://code.google.com/p/zaproxy/wiki/Downloads?tm=2. (Note: There is a MAC version available as well as a Linux version. You are welcome to try either of these versions, though only the Windows version has been tested with these lab activities.)

Once the tool has been installed, start the OWASP ZAP tool running. Open up firefox and setup ZAP to act as an internet proxy. You'll need to make the setup changes shown in Figure 1.

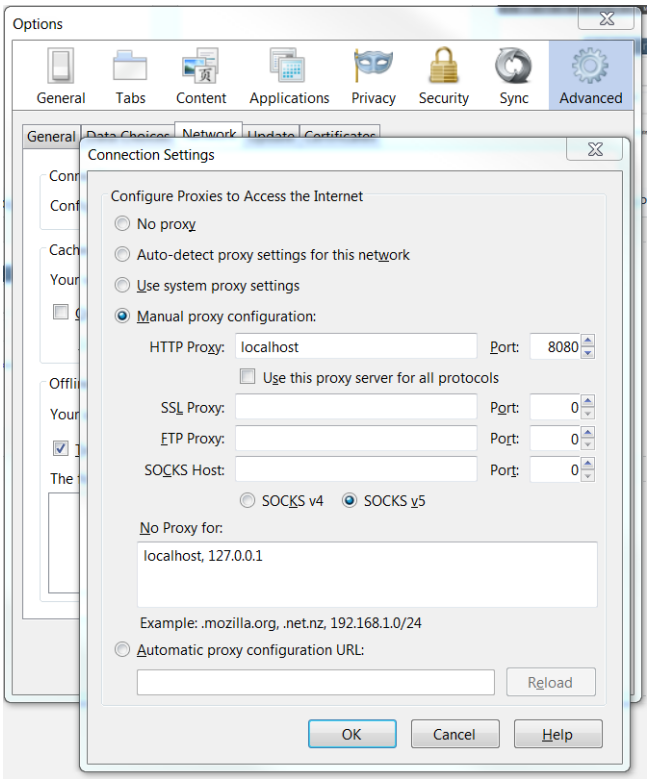

Figure 1: Proxy configuration

Once this is done, start up Metasploitable and determine the IP address of the machine. Open up firefox and browse to the main site. In your browser, you should see a screen similar to that shown in Figure 2.

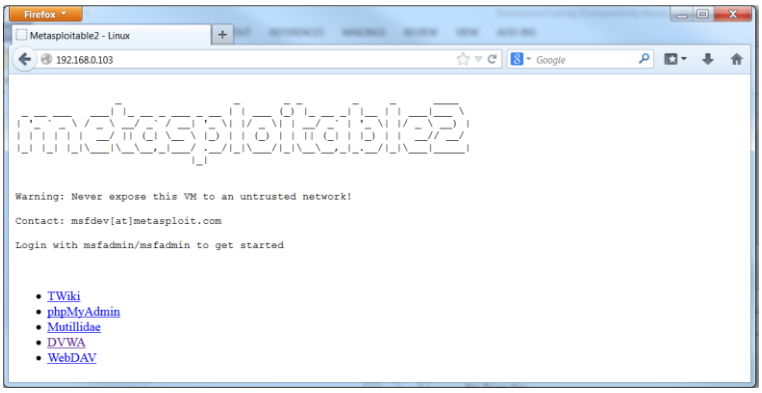

Figure 2: Metasploitable web browser

Select the DVWA link and log on with the credentials "admin" and "password". From this screen , select the "command execution" button and enter the ip address 127.0.0.1 into the dialog box. This will ping the local host.

Step 2 Creating our first Fuzzer

As was discussed in class, fuzz testing involves changing parts of a call to include random data. One of the things we often want to test for is what happens with commands. To do this, we want to create a custom fuzzer that has certain commands in it that might be useful. In a text editor of choice, create a file with the following lines:

;ls
;cd /;ls -al
;cat /etc/passwd
;ifconfig
;cd /root; ls -al

Save the file to a file "commands.txt" somewhere on your machine. In ZAP, select the "Tools->Options->Fuzzer" menu and add a custom file to the fuzzer by selecting the file you just created. This will add the permutations you entered as potential fuzz commands.

In ZAP, find the "POST" command in the vulnerabilities segment on the Sites tab. If you then click on the request tab, you'll see the actual POST that occurred when you typed the command, as is shown in Figure 3.

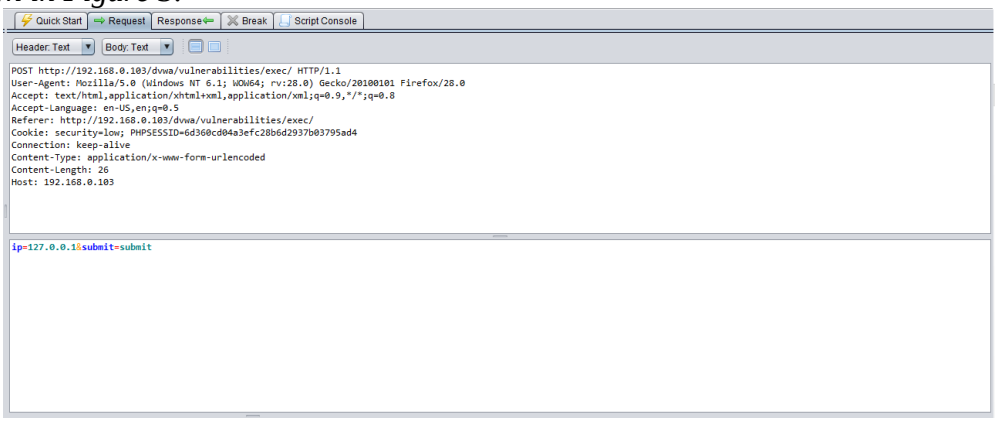

Figure 3: POST Request tab.

Take the cursor and place it in the parameter area for the post operation right after the ip address and before the &submit. Right click and select "fuzz" Browse the category for a custom fuzzer and select "commands.txt" and "fuzz". In the window below, take a look at the results. If you click on the results, you can see some interesting things got sent back.

Question 1: What files are in the directory with the query?
Question 2: What information about passwords is disclosed by this fuzz attempt?
Question 3: What can you determine about the root of the system from this?

Once you have answered the questions from the response, try entering on of these commands directly into the web browser and see what it does.

Step 3: Creating our Second Fuzzer

Now that we have done this, we'd like to determine what other machines are present on the network from this mechanism. Create another file with the name "numbers.txt" and the numbers 0 – 255 each entered on a line. Go back in the web browser to the siteand enter the ip address 192.168.0.100 and perform the ping. Now in ZAP, add this fuzzer as a custom fuzzer, select the last part of the IP address, and determine which other machines are on the network.

Step 4: SQL Injection

We'd like to learn a bit about which users exist on the system. Click on the SQL injection tab and enter a user ID of 1 in the window. Click on submit. You should see a first name and a last name for the user 1. On the Sites tab, select the GET request which submits and id. Inh the text, find the number '1' and select it. Right click on the text and fuzz. Use the customer fuzzer you created before with the numbers. Look at the results of the method.

Question 4: *How many users are there? (Hint: Reflected indicates that the parameter passed is part of the response. Successful does not show this.)*

We'd like to learn a bit more from this tab, potentially the user ids and other things. Create another text file "sql.txt" with the following text in it:

%' or '0'='0
%' or 0=0 union select null, version() #
%' or 0=0 union select null, user() #
%' or 0=0 union select null, database() #
%' and 1=0 union select null, table_name from information_schema.tables #
%' and 1=0 union select null, table_name from information_schema.tables where table_name
%' and 1=0 union select null, concat(table_name,0x0a,column_name) from
%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #

Question 5: *Based on this information, can you find out anything important about the system?*

Step 4: Scanning for Vulnerabilities

Up until this point, we've mainly focused on simple scans and fuzzing which has revealed quite a bit about our system. We'd like to take advantage of the tool a bit more. Start by logging out of the web app in the browser. Then, start a new session, as we want to clear away anything else from our history in ZAP. Relog into the system with the credentials you used before. Right click on the dwa folder on the sites tab and select "Attack->Active Scan Advanced" Set the policy up to match that shown in Figure 4.

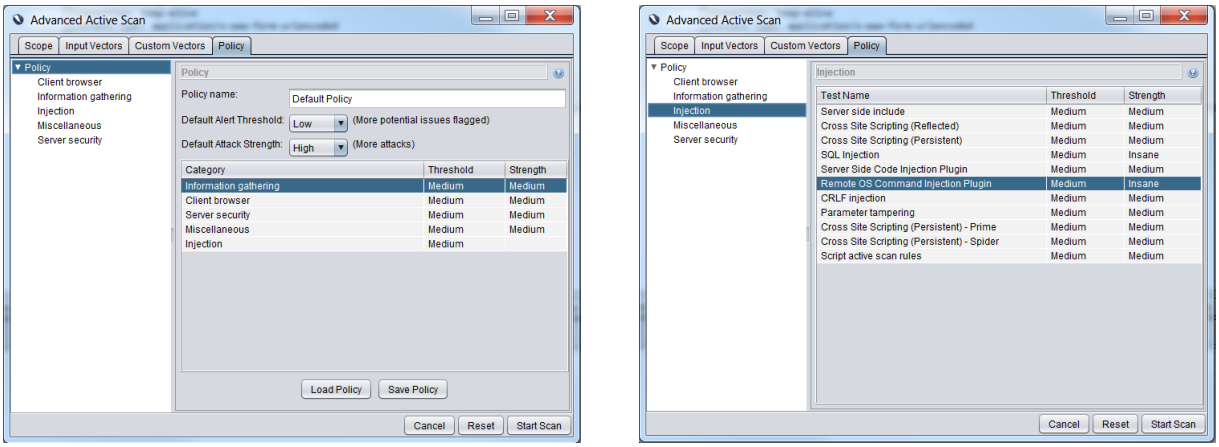

Figure 4: Active scan configuration

Once this is done, click "start scan" and wait while the system scans. While you are doing this, watch the youtube video at https://www.youtube.com/watch?v=dqKGGCVFTvI#t=35 which explains a bit of how to work with ZAP while doing injection (though it is an older version). When the tool is finished scanning (and it may take a while), determine what are the most serious vulnerabilities.

Question 6: *What are the most serious vulnerabilities in this system?*
Question 7: *How many vulnerabilities does the tool discover?*

Step 5: Scanning twiki or Mutillidae

When you are done with this task, try scanning twiki or Mutillidae.

Question 8: *What do you find for these systems and how do they compare with DVWA?*

Step 6: Scanning another system for testing purposes

When you are done with this step, repeat the last step on a website which you have been involved in its development. Examples might include your SDL projects, your senior design projects, or something you created in class (i.e. your web applications course). What does the tool find? (Ethical note: Choose wisely for this. DO not choose a project in which crashing might have extremely negative consequences. For example, scanning my.msoe.edu would not be wise.)

Deliverables

Each team shall submit a report in pdf format with the following information

1. *Title Page* - Name of all team members, course, and date details.
2. *Introduction* - Summarize what you did in lab and the problem your were trying to solve.
3. Questions

   a. Answer each of the 8 questions in the narrative
4. Independent analysis
   a. What system did you analyze and why did you choose it?
   b. Prior to testing, what was the quality level of the system?
   c. How secure was the system based on the scan results?
5. Discussion
   a. What pieces of information does this lab provide that might be useful to securing the deployment of a software system?
6. *Things gone right / Things gone wrong* –
   a. Discuss the things which went well with this lab, as well as the problems you had, either with the tools, the process, etc.
7. *Conclusions*
   a. What have you learned from this experience