System Security Group Project #2

(10% of class grade, due 2/14 at midnight)

Go through interactive broken web application tutorials. Explain what you learn! This project is worth 10% of your class grade.

What you will need:

You will need access to the class vSphere environment. You will be using the following virtual machines:

- WebGoat
- Kali Linux

Directions:

Setup

Follow the project videos for accessing and running the WebGoat and the Kali Linux virtual machines.

Remember, you only need to use the VPN to access the virtual environment if you are off campus. Starting the VPN while on campus can cause issues getting to other websites.

Start the WebGoat image (if not already started). Start and log into your Kali Linux virtual machine (username root, password toor). To get the IP address for WebGoat, go to vsphere and click on the WebGoat virtual machine to select it after it has booted. Look under the “Summary” tab.

Open up Firefox in the Kali Linux machine. Put the following URL in your web browser, where the IP address (before the :8000) is the IP address of your WebGoat machine:

http://10.161.20.5:8000/WebGoat/

(In this example, in my folder, which is folder 20, webgoat had the IP address of 10.161.20.5. Your webgoat address will start with 10.161.x.y), where x is your folder number and y is assigned by the dhcp server.)
Create a username and password for yourself, and **be sure to put it in your group documentation.** You don’t want to lose your work if you forget your password. I can’t/won’t reset it for you…

What is WebGoat? (From the OWASP website)

“WebGoat is a deliberately insecure web application maintained by OWASP designed to teach web application security lessons. In each lesson, users must demonstrate their understanding of a security issue by exploiting a real vulnerability in the WebGoat applications. For example, in one of the lessons the user must use SQL injection to steal fake credit card numbers. The application aims to provide a realistic teaching environment, providing users with hints and code to further explain the lesson.” – WebGoat wiki at [https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project](https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project)

On the left, click on the Introduction link. It will expand. Click on and read the Webgoat lesson. (Note: The next lesson called WebWolf talks about a tool that is not yet installed on your VM. You can skip this for now.)

**Lessons in WebGoat**

Start with the General lesson. Complete both of the following.

- HTTP Basics
- HTTP Proxies (Note, you may have to do this lesson first to complete the HTTP Basics lesson. ZAP is already installed on your Kali Linux machine, so follow the directions to hook it up to Firefox.)

You’ll notice that there are **lessons** and **assignments** in WebGoat. Lessons are the links on the left-hand menu. Assignments are usually numbered in red and require you to solve the puzzle. There are 54 total assignments in this version of WebGoat. You can see how many lessons are completed by clicking on the report card icon in the upper right:

At the top of the report card will be a summary of what you have completed (see next page):
For groups of 3, you should successfully complete 36 **assignments** in addition to the 3 assignments in general. That means, in total, you should complete 39 assignments.

**The deliverable**

I am expecting 2 (optionally 3) parts to the deliverable.

1. **Paragraphs:** For each **assignment you complete**, take a screenshot (pic) of successfully completing the assignment. Then type up at least one paragraph on what you learned. Don’t just copy and paste text from the lesson plans (I will be checking). In your paragraph, let me know what was easy, what was hard, and what you couldn’t get to work quite right. We will use class periods as work periods to ask questions about things you can’t quite get and get advice from other teams. (All advice from other teams should get a citation/shout out in your paragraph.)

2. **Report cards:** Take screenshots of your final report card(s). Each group member should be participating. For example, if you split the project up into three equal pieces, each member should submit a score card. If you all worked together on all the activities, just submit one score card. If you worked together on some things and individually other things, you will need to make clear who did what and submit all the score cards. Some lessons contain much more work than other lessons, so keep that in mind when breaking up your project.

3. **[Optional] How could this project be made better in the future?** If you have some ideas on how to make this assignment better or anything more helpful in the future, let me know!

Your deliverable should be in your team google folder. Create a new google doc called “WebGoat” directly under your main team folder and place your paragraphs and score cards in there.
In addition, you will be required to fill out an internal peer review before you receive a grade. (You cannot skip this part.)

**Other Things**

- I do not have all the answers. (I know, I look like I do...)
- This is the newest version of Webgoat being tested on you. Expect some bugs.
- I’m expecting things to sometimes be difficult, even with the hints and solution videos available. Sometimes the difficulty will be in getting things set up. Other times it might be in figuring out what you should be watching or modifying. Sometimes a new term will come up that you need to look up.
- If you don’t know what something is, Google it to find out!
- I expect you to have to use Google to figure out some features of ZAP, wireshark, Firefox dev tools, etc.
- I know other walkthroughs exist on the Internet. If you use another source, cite it!
- Make sure you are taking the time to learn. If your paragraphs don’t convince me you really understand what’s going on, I will award less points.
- Doing some things in the “Challenges” lesson may earn you extra credit, but be careful! (E.g. some students have accidentally reset the WebGoat VM in the past by doing some things, so take a working VM snapshot of the machine before you play around in here.)