Homework 1 – A scalable environment

# Lab Information

## Due Date:

Homework 1 Dropbox Deadline

## Objectives/Goal:

To understand popular incarnations of interception technologies that deal with web applications.

You have learned about various web technologies this week. Your goal is to setup an environment that is designed to scale with a large amount of load. As you design this consider how each of these components interact with HTTP Requests. Your goal will be to setup an environment with 3 different types of components: A Load Balancer, A caching server, and at least two Web Servers.

## Deliverables:

* An image or scan of the completed signoff sheet.
* Configuration files
* A breakdown in support of your choices.

## Table of Contents:

[Lab Information 1](#_Toc460328030)

[Platform Choice: 2](#_Toc460328031)

[Vulnerability Scanning: 5](#_Toc460328032)

[Signoffs 6](#_Toc460328033)

# Activity 1: Webserver Setup:

Your first objective, if you choose to accept it, is to setup a web server. While Apache has the highest market share there are a number of possible, mature, alternatives. Webservers are simply servers that will respond to HTTP requests, although as we have gone over in class there are other supported protocols in many cases.

## Step 1: Choose a webserver

Research available webservers and choose one which you think best suits the need of the remainder of the lab. Generally you are given the choice of the following:

* Apache
* Nginx
* IIS

You may make, or choose a different technology with professor approval, however the aforementioned technologies may be used without additional approval.

## Step 2: setup a webserver

Now that you’ve made a decision on what platform you will use, it is time for you to install that webserver and configure it to serve a basic page. Configuration will vary based on which platform and OS you choose. Once you have the web server installed setup a basic text based page that simply displays “Hello World”. Your pages, source must be verified as HTML 5 compliant using the W3C validator (<https://validator.w3.org>). Show this to your instructor to receive your first signoff.

# Activity 2: Caching Server Setup:

Caching servers are important staples of computing security. At each layer where there are speed changes, caches can be used to mitigate the change if the right algorithm is used to take advantage of locality. Your second activity is to use such a technology to speed up your environment.

## Step 1: Choose a caching server

Research available caching servers and choose one which you think best suits the need of the remainder of the lab. Generally you are given the choice of the following:

* Varnish
* Nginx
* Squid

You may make, or choose a different technology with professor approval, however the aforementioned technologies may be used without additional approval.

## Step 2: setup a caching server

Now that you’ve made a decision on what platform you will use, you should install it. Come up with a simple method to verify that the caching server is in fact working and demonstrate this to your professor to get a signoff.

Hint: this may involve changing the page and demonstrating that the cache still displays on the web browser.

# Activity 3: Load Balancer Setup

Now we are going to complicate things. Your task is going to be to do some research and determine the best location for a load balancer within the environment you already created. This will in general require you to add another webserver.

## Step 1: Choose a load Balancer

Research available caching servers and choose one which you think best suits the need of the remainder of the lab. Generally you are given the choice of the following:

* Nginx
* HAProxy
* Varnish
* Apache
* Zen Load Balancer

You may make, or choose a different technology with professor approval, however the aforementioned technologies may be used without additional approval.

## Step 2: setup a load balancer

Now that you’ve chosen a load balancer, determine where you’ll put it. Once you’ve figured out where it goes in your topology (web server and caching server so far), you might find you need another webserver – deploy that as needed. Finally deploy your load balancer and demonstrate it is functional but having it server [slightly] different versions of the same page. Be sure that your original caching server is also still functional.

# Activity 4: Proxy Scanning

As we discussed in class HTTP has built in proxy capabilities. Often these are capabilities are disabled, but if you can find a webserver that supports HTTP proxying, it can be very useful for surfing the web anonymously. In this exercise you will be using an existing library to try and find these open proxies.

## Step 1: Using the requests library (or equivalent).

If you’ve never used Python requests now would be a great time to become familiar with it. The library makes it very easy to make standard HTTP requests to a site without having to know much about the underlying protocol. In the remainder of the course we will be creating a similar library. Use the requests library to request csec.rit.edu

## Step 2: Write a scanner that will try to find an anonymous HTTP proxy

Now that you have a basic HTTP request script running, all you’re going to need to do is slightly expand it. You should write this script such that it can take in a range of IP’s and scan each one looking for an open HTTP proxy.

We talked briefly about how these requests look but you will only need to know how to use Python Requests (or similar) in order to generate it (hint no need for a raw HTTP packet). Once you have it correctly generating scan ranges of the internet until you find and open proxy server. Be sure to discuss with the person signing off how you know that this server is an open proxy and not just a webserver.

Hint: You may wish to leverage threading to speed this up.

# Writeup

As you go through the remainder of your career you will often be faced with many choices. In this case you were given a set of options, all of which were free – but there are also many situations where cost will factor in. Given the above options, how did you select which technologies you would use. Do some of these work better in conjunction? Are some easier to use or deploy?

Also attach your topology and discuss why you chose this topology and what the alternatives might have been. Lastly, include a discussion about why these technologies might be needed. At the same time you may wish to discuss why the single text web page example we just created might not need all the complexity we developed.

# Signoffs

## Activity 1 – Demonstrate that you have a functional webserver

## Activity 2 – Demonstrate that you have a functional caching server

## Activity 3 – Demonstrate that you have a functional load balancer

## Activity 4 – Show that you can scan the internet for Proxies