HOMEWORK 3

COSC6377

Student: Wellington M. Cabrera

Assignment Description

In this homework, we will learn how to measure the performance of a web server with different settings. Among settings, we will just focus on serving pages with HTTP and HTTPS. Sign up for Amazon AWS account. Setup an Ubuntu instance (medium instance type) with Apache web server.

Configure your web server to serve over both HTTP and HTTPS. Create some binary files that are 200KB in length. This is your complete web server setup.

The client should consist of a command line tool such as `wget` or `curl`. You can run the client on the same machine as the server but we recommend running the client on a different machine.

Now that the server and client setup is ready, we are ready to perform some measurement. We want to understand the throughput offered by the server over HTTP vs HTTPS. To estimate the throughput with either protocol, you can launch as many concurrent downloads as possible using the command line clients. During your experiments, you may want to monitor system metrics such as CPU utilization to understand what the bottleneck may be.

Experimental Setup

Web Server

Our web server is a m3.medium instance Amazon EC2 server, with 3.75 GB of RAM, 1 vCPU. The server runs Ubuntu 10.04 and Apache 2. The characteristics of the CPU are presented below.

```
Architecture:          x86_64
CPU op-mode(s):        64-bit
CPU(s):                1
Thread(s) per core:    1
Core(s) per socket:    1
CPU socket(s):         1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 62
Stepping:              4
CPU MHz:               2500.042
Hypervisor vendor:     Xen
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              25600K
```

Amazon's servers have SSL enabled by default, since the access to the server console is through SSH. By a standard procedure described in popular web sites, we enabled HTTPS in the server. Note that we use a Self Certificate; we do not have to pay for it to a Certification Authority. We load a web page with several medium-sized pictures. The weight of the webpage, including pictures is 1.2MB

Client.

The client is an Intel Dual Core computer with 4 GB RAM, running Ubuntu.

Sending Massive HTTP/HTTPS request to the Web Server

We send 1000 HTTP and 1000 HTTPS simultaneous requests to the server , running wget in background. We confirmed that every operation download 1.2 MB from the server. The Bash script is presented below:

```
#!/bin/bash
for i in {1..1000}
do
 wget -r
http://52.6.196.132/Harness%20the%20power%20of%20the%20Internet%20of%2
0Things%20%20Pingdom%20Royal.htm &
done
```

Measuring Performance in the Web Server

While the server is being bombarded of HTTP/HTTPS request, we have to measure performance inidicators in the server. We focus in measurements of CPU utilization and RAM consumption. We try two performance evaluation tools: `nmon` and `top`.

We run top from the command line, with a refreshing rate of 3 seconds, and for a total time of 480 seconds. The output of top is redirected to a file, in order to get the data for analysis.

Results

By preprocessing the output of top, we obtain several measurements from the Web Server, which are useful to get insight about the computational load of HTTP and HTTPS. The dataset has the following structure:

Time elapsed, % CPU Utilization- HTTP, %CPU Utilization-HTTPS, Memory Used-HTTP, Memory Used- HTTPS

In Figure 1 we plot the Percentage of CPU's working time for HTTP and HTTPS. Even though the massive amount of connections, the CPU is idle most of the time for both HTTP and HTTPS. In Figure 2, we show the RAM consumption , under the same conditions explained for Figure 1. While the RAM consumption suddenly grows , it is steal a modest fraction of the total RAM of the server. We elaborate the analysis of the results in the conclusions.

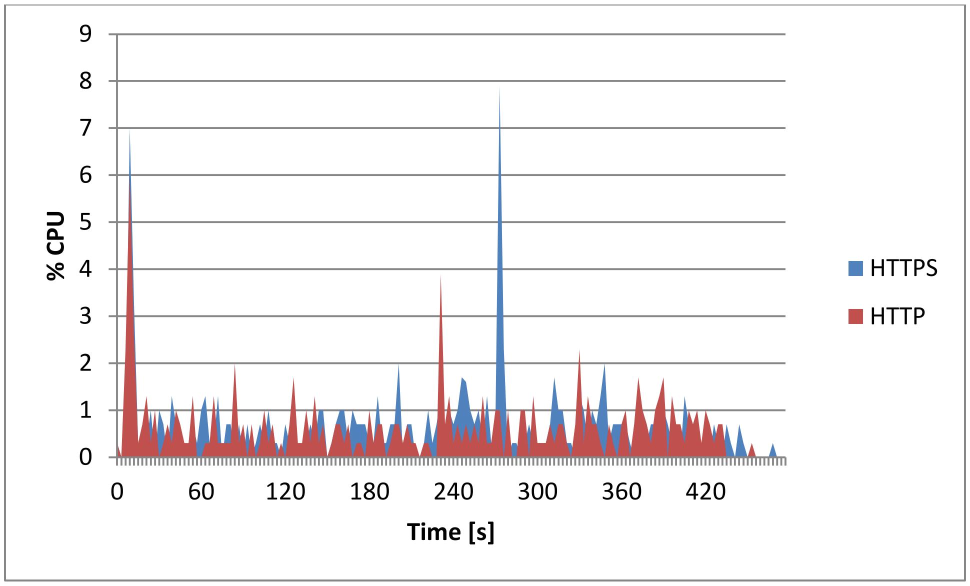

Figure 1 Web Server's CPU utilization comparison for massive HTTP/HTTPS requests

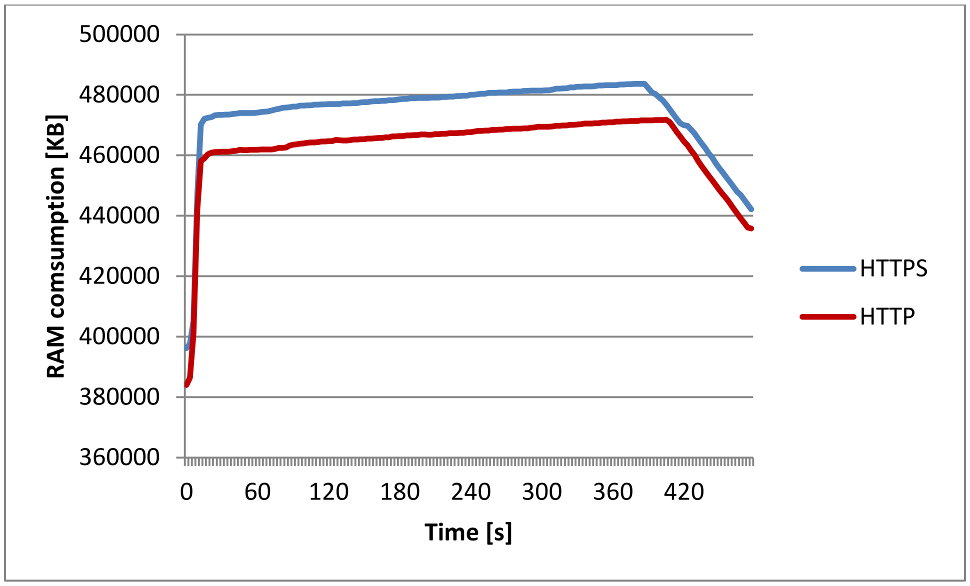

Figure 2 RAM consumption on Web Server; comparison for massive HTTP/HTTPS request

Observations.

1. Under the conditions of the experiments, a massive amount of HTTP/HTTPS requests does not present a major impact in the web server CPU
2. HTTPS produces a slightly highly CPU consumption, comparing to HTTP.
3. The increasing in CPU consumption due to HTTPS is negligible (under the experiments considerations)
4. One thousand simultaneous HTTPS connections produce an increasing of RAM consumption, that is about 100MB . One the other hand, the increasing of RAM for the HTTP experiment was 80MB.
5. The pattern of RAM consumption is similar for both HTTPS and HTTP

Conclusions.

Even though the observations in the previous section may give us the idea that the impact of HTTPS is neglegible, I think that we cannot make a definitive conclusion, due to some conditions of the current experiments :

- All the connections attempt to open the same web page. This condition may favor the Web Server performance, because a greater chance of taking advantage of caching.
- Since the webpage is small, the CPU may benefit of the unusual condition of high cache hits. The CPU's L3 cache is much bigger than the test web page.
- Since we tested with a static web page , the page rendering is a small workload for the Web Server

In my opinion, further experiments and analyses should be conducted in order to clarify the effects of HTTPS in the performance of Web servers and clients.