

Distributed Denial of Service (DdoS) simulation

* In a peer-to-peer (p2p) network & a botnet of virtual machines



COURSE: MGS 655: Distributed Computing, Professor: Dr. Haimonti Dutta

November 26, 2015

MS-MIS, University at Buffalo

TEAM MEMBERS:

Suvir Singh (50161742)

Potuluri Reddy Ganta  (50166764)

ABSTRACT:

The problem in hand is to understand the gravity of ‘Distributed Denial of Service’ (DDoS) and to apply the concepts of ‘Distributed Computing’ to simulate DDoS in a controlled setup.

Understanding the severity of the problem:

(These data as taken over a year 2014-2015)

- 132.43% increase in total DDoS attacks

- 122.22% increase in application layer (Layer 7) DDoS attacks

- 133.66% increase in infrastructure layer (Layer 3 & 4) DDoS attacks

From Akamei Reports [1] we can see that in Q2 2015, the largest DDoS attack measured nearly 250 Gbps, an increase in size from the largest (170 Gbps) attack in Q1 2015. To see a live DDOS attack, we can go to <http://map.norsecorp.com/> [A1] where the attack origin, attach type, IP, timestamp, port numbers is listed.

Why the intersection of DDoS and p2p is important for us:

1. The history of p2p (peer-to-peer networks) and its never ending problem with security.
2. The revival of p2p and mesh networks. Industry is banking heavily on them to gear up for the biggest wave of the Internet’s history – ‘*Internet of Things*’ (IoT)
3. Though DDoS attacks are not something new and the industry has already a lot of proven mechanisms established against them, still DDOS attacks have been on the rise in the recent years.

Our work and results:

We implemented the simulation of DDoS by creating a botnet on a *peersim* simulator*,* which is a p2p simulator and on a botnet created by using the concept of virtualization.

INTRODUCTION:

According to Forbes: ‘*The worldwide cybersecurity market is defined by market sizing estimates that range from $77 billion in 2015 to $170 billion by 2020.* *Cyber-attacks are costing businesses $400 to $500 billion a year, and does not include the large number of cyber-attacks which are not reported.*’

Getting motivated by such huge numbers, we thought of simulating the biggest contributor to all the attacks: DDoS

We split the problem in non-overlapping even parts:

|  |  |  |
| --- | --- | --- |
| Scope | Suvir Singh | Potuluri Reddy Ganta |
| Part of DDoS under consideration | Severity of DDoS attacks  Types of DDoS attacks | Prevention mechanisms in place for DDoS |
| Simulation type | Volumetric TCP connections attacks on a p2p network, thereby creating a botnet | Virtualization: Creating a botnet by using Virtual machines |
| Summary of work done | Index Poisoning and Routing Table Poisoning on peersim, using a chord protocol | A WAMP server was created in virtual machine and the web server was ping from another system by creating a botnet to attack the victim. |

RELATED WORK / NOVELTY OF YOUR TECHNIQUE:

The concept of DDoS can into mind after carefully understanding the security reports of world’s leading information security companies: Arbor Networks and Akamai Technologies. We attended a lot of webinars conducted by BrightTalk and Arbor Networks to understand the industry viewpoint DDoS types and the prevention mechanisms as well.

We studied in detail a paper ‘Exploiting P2P Systems for DDoS Attacks’ [2] by Naoum Naoumov and Keith Ross, which talks of simulation and detection of DDoS on a peer to peer network using *Overnet*  which is a DHT-based file sharing system. With more than one million concurrently active peers, Overnet is probably the largest DHT deployed to date. It is also a core component of eDonkey, which generates today more traffic than any other content-distribution system, including BitTorrent.

We took the approach to peersim and did our simulation on the software introduced to us in the class. Hence we added our novelty to it. Going forward we decided to use all the concepts that we learnt in the class. We shortlisted chord protocol after carefully examining the paper ‘Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications’ [3]. Also we brought in the concepts of indexing, key-value pairs and the routing tables to the idea and applied the problem on to these concepts.

DESIGN ISSUES:

DDoS Simulation on P2P Networks (Suvir Singh):

The design I had used is entirely based on the distributed computing concepts learnt in the class. I created a peer-to-peer network using the peersim simulator. I changed the configuration file to cater a small problem in hand first. I used the chord protocol for maintaining look-up tables and used the concept of finger tables also referenced as routing tables in this document. Each node need to keep a track of only K/N nodes, where K is the keys and N is the number of nodes used in the configuration file.

The complexity of the look-ups comes to O(logN). The reduced lookup time helps the attacker to target the victim quickly. Therefore before the victim identifies that it has been attacked by millions of nodes and starts a defense mechanism, the fast lookup time of the protocol chokes the bandwidth of the victim and floods it with millions of open TCP connections.

The two approaches used to poison the p2p network are:

- Poisoning Indexes: A new node with a bogus record is inserted in the p2p index system saying that some popular file is present at the target system and provide the target system's port number and IP address. When other peers search for that file, they all try to communicate the target machine (which does not have the file). If many peers attempt to download from the victim host, the victim host becomes subject to TCP-connection DDoS attack.

Therefore, in this approach the attacker tells the p2p system that the popular files are the target's machine.

- Poisoning Routing tables: If we announce that there exists a 'bogus peer' and poison one node with this information. In this we wish to trick peers into adding bogus neighbors into their routing tables using finger table synchronization function (where the IP address of these bogus neighbors is the IP address of the victim). Later when other poisoned peers wish to forward a message to the bogus peer, they will send a lot of TCP messages to the target machine, thereby simulating a DDoS attack.

Therefore, in this approach I try to add the target machine in the finger tables of all the other nodes.

**DDoS Simulation on Virtual Machines (Potuluri Reddy Ganta):**

A "flood attack" is when you drown a target server under a lot of request. Each request entails some effort from the client, and some effort from the server; the DoS is effective when the server gives up before the client. This means that either the per-request effort from the client was less than the per-request effort from the server, or, more often, that the client mustered more CPU and network bandwidth (that's the idea behind a Distributed Denial of Service).

If you attack "localhost", then you run both the server and the client on the same machine, which means that:

1. The client and server work share the CPU resources, and will drown simultaneously.[A5],[A6]
2. The network bandwidth between client and server will be extremely fast.

Both conditions mean that whatever you measure in such a situation will not be representative of what a flood attack is, how long your server would resist such a flood, and how efficient a given tool is.

To simulate the attack using virtual machine, I installed WAMP web server in virtual machine and tried to create a website in localhost and attack the localhost from another machine.Below are the steps to create the same.

**Install/Run XAMPP**

Go to the XAMPP download page and grab the Mac OS .dmg file. Run the installer. The

default install path is /Applications/XAMPP. That will be important later. When you launch

the application, select the tab that says ”Manage Servers.” Start Apache and MySQL.

Open up a web browser and go to http://localhost/xampp/index.php and you should see

xampp.

**Install DVWA**

Download DVWA (linked version in this PDF is 1.0.8 - the latest as of the time of this writing,

but may not be the latest for you). Extract the zip file. Now copy and paste the dvwa folder

into \Applications\XAMPP\xamppfiles\htdocs You should be able to see the mysql error

if you navigate to <http://localhost/dvwa/login.php>. [A4]

**Configure DVWA**

To fix the mysql error, open up \Applications\XAMPP\xamppfiles\htdocs\dvwa\config\config.inc.php

and find the line that says:

$ DVWA[ ’db password’ ] = ’p@ssw0rd’;

and change it to:

$ DVWA[ ’db password’ ] = ”;

Now you should be able to set up the sql tables in dvwa in the browser.

After this we can successfully installed DVMA on our machine and we can test and attack the host which is located in localhost/dvma from other machine which is running on Mac OS and another virtual Machine which runs Kali Linux.

We install Hping3 using command shell or terminal of Linux and Mac Terminal respectively and SYN flood the localhost running on Virtual Machine on Windows using the below command line and same thing was done using python script for creating a botnet.

sudo hping3 -i u1 -S -p 80 local host name [A7],[A8]

ALGORITHM DESCRIPTION:

1. Create a p2p network using peersim. Run the project in Eclipse IDE [A2]
2. Modify the network.size parameter in the config-example1.txt. Change the number to 15
3. Past the chord folder in the peersim directory and run:

‘java -cp peersim-1.0.5.jar;jep-2.3.0.jar;djep-1.0.0.jar;chord/classes peersim.Simulator chord/example.cfg’

1. Capture the chord output [A3]. Play around with the config and the source files for the below mentioned approaches.
2. Customize the following function for better User Interface:

public boolean execute(): Returns statistics over minimal path length and clustering. The output is the average over the set of clustering coefficients of randomly selected nodes, and the set of distances from randomly selected nodes to all the other nodes. The output is always concatenated in one line, containing zero, one or two numbers (averages) as defined by the config parameters.

1. In the execute() method we must call updateGraph() in order to check if some change has occurred on the actual graph.

The steps for index followed by Chord are listed below. By understanding these key parameters and key functions we can tweak them to do the **index poisoning** simulation for us.

* private static boolean executeNext()

/\*Execute and remove the next event from the ordered event list.\*/

Calls processEvent(ev.node, pid, ev.event);

public void processEvent(Node node, int pid, Object event)

It sets dest = find\_successor(target);

LookUpMessage message = (LookUpMessage) event;

t.send(message.getSender(), dest, message, pid);

If we write our own function, say my\_ processEvent and have arguments like:

node, pid, event, ip\_address, port\_number, file\_name -> therefore we can mention target’s IP address, its port number and the announce that it has a popular file.

Most of the remaining code will adjust itself and create this bogus node and add its index to the key-value pair information.

The steps for **routing table poisoning** are:

public peersim.core.Node[] fingerTable

Our aim is to poison the enteries of this variable. We can have print statement in between for better logging and debugging. Now we see how are actually populating this variable in the existing code:

initialize(Node n)

join(n)

cp.fingerTable = new Node[cp.m];

stabilization

cp.fixFingers();

Hence, if we want to poison the finger tables of every nodes, we need to target the initialization() function and then write our own my\_join() function. We can add bogus nodes through the initialization function and the my\_join() function will automatically update the global fingerTable array by calling the stabilization() function. The bogus node will have the target’s IP address.

Finally we run the below mentioned function to create the finger table for the whole network:

public void nextCycle( Node node, int protocolID )

execute();

createFingerTable();

This is our own approach and has not been covered by any paper. The time complexity for the look-ip table will be O(logN) and each node will store K/N entries in its finger table, where K is the number of keys and N is the number of nodes mentioned in the configuration file.

SOFTWARE:

- Peersim Software using chord jar file and using the graph module (<http://peersim.sourceforge.net/>)

- Eclipse IDE for running peersim, java language

- Uniformserver, A lightweight WAMP Server solution Software

- LOIC (Low Orbit Ion Cannon) - DDoS attack tool (<https://en.wikipedia.org/wiki/Low_Orbit_Ion_Cannon> )

RESULTS:

After poisoning the indexes or the routing table of the p2p network, the attacker can mention any ip address at will and can choke any system or user in the world. The attacker moves from a million nodes p2p network to another, thereby creating more and more botnets. The systems in the p2p network do not come to know that they have been made a part of this attack.

Also when the victim starts a defense mechanism, it is the node of the p2p network who are punished and the attacker still walks away free.

**DDoS Simulation on Virtual Machines (Potuluri Reddy Ganta):**After flooding the localhost using SYN FLOOD attack from Mac OS and Linux on Virtual Machine attacking the localhost on Virtual Machine on Windows. There were few observations made.

1. The LOCALHOST which took no time to load, now takes one second or two seconds to load as we are performing the attack from a very powered local machine and not even a server. [A9]
2. Once the attack was started I could not even open a google page on my laptop as all the network is consumed by the packets sent for attacking the LOCALHOST. [A9]
3. The SYN Flood can be simulated with the same tools on bigger machines and we can successfully replicate DDoS attack.

PRACTICAL EXPERIENCES:

We had the problem of simulation in a controlled environment. DDoS experiments can turn out to be nasty and hence can disturb the topologies used by the University at Buffalo. So we talked to a lot of professors and reached out to many students as well. We had valuable discussions with Professor Haimonti Dutta, Professor H.R. Rao, Professor Dave Murray, Teaching Assistant Tamal Biswas, Teaching Assistant Jake Lee and Sleiman Lab In-charge Stephen.

The other challenge for us was the unfamiliarity with the java language and the usage of peersim. But then we always found help in this from Tamal. We read the code and identified the exact functions and variables that we wish to change for the successful simulation of the project.

If we get more resources, in terms of time and familiarity with java language we would love to play around more with the config files of chord. We wish to write our own functions for node insertion and finger table synchronization. A customized UI that shows the target machine load getting up and how the poisoning happens in O(logN) time. We also wish to publish a paper on this, as this approach on peersim is novel.

Yes, we reduced our scope from the project proposal. Initially we had proposed that we will do simulation on peer-to-peer and cloud, DDoS detection and then DDoS prevention mechanisms as well. After careful consideration of the complexities involved and the time we felt that we had too much in our plate. We then reduced the project scope to DDoS simulation on peer-to-peer networks and to virtual machines. We informed the same in our project intermediate report and got excellent motivating comments from the professor.

CONCLUSION AND FUTURE WORK:

DDoS is fun! We feel lucky to select such a problem which has a great role in the future. With IoT coming into our lives, changing the business models and the existing security models of all the established organizations, it is really good that we looked into the problems that may hamper our growth.

Going forward, we would like to do the following things:

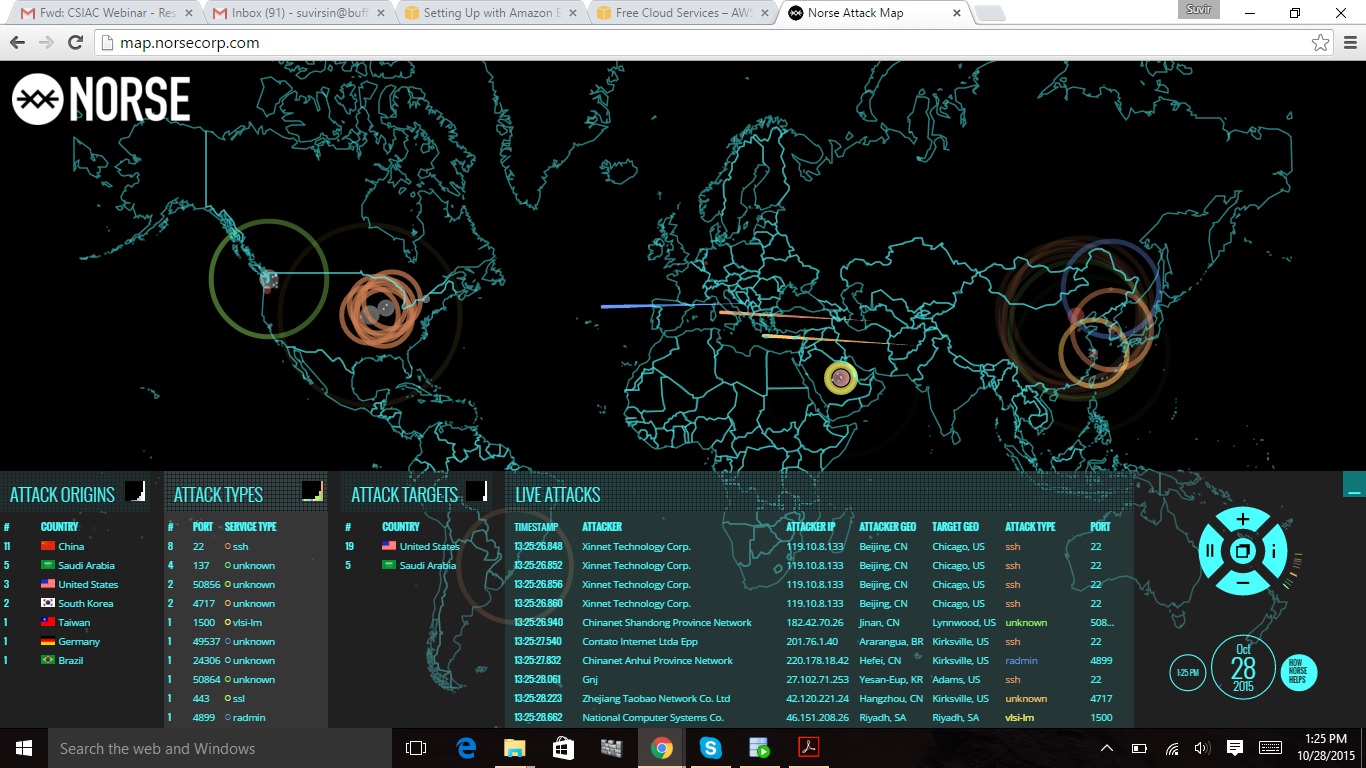
1. Take the simulation to Cloud. We would like to setup a cluster on the cloud and then try to poison that cluster to act as a botnet for the attacker. We studied few papers and watched a couple of webinars to tackle this.
2. Learn more about the DDoS detection mechanisms.
3. Our ultimate goal is to propose a DDoS prevention mechanism to the industry and thus contributing our bit to the society.

To come up with a prevention mechanisms, we need to know where and how DDoS can affect in different environments. Thus the current problem in hand, where we are simulating the DDoS on a p2p network and virtual machines act as a perfect starting point for us.

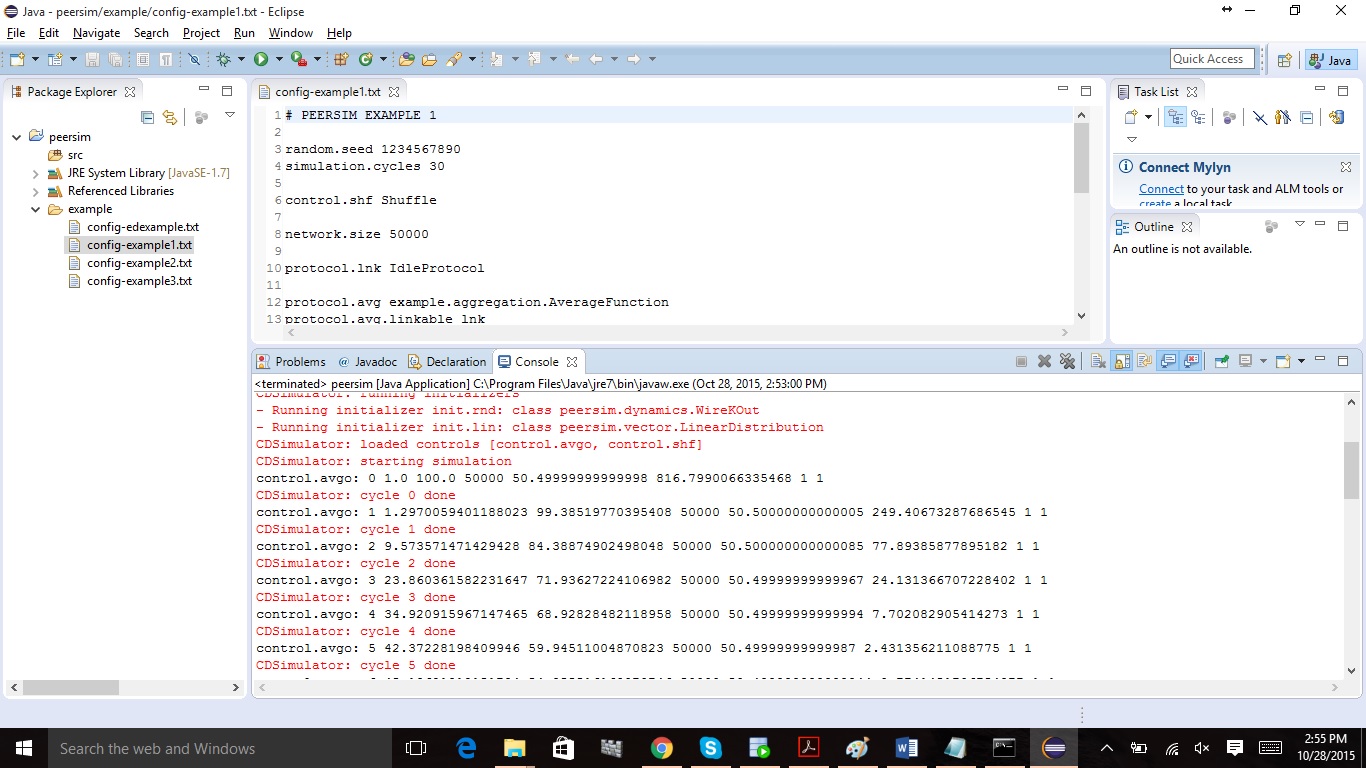
BIBLIOGRAPHY:

1. Akamai’s [state of the internet] / security Q2 2015 report
2. Exploiting P2P Systems for DDoS Attacks, by Naoum Naoumov and Keith Ross Department of Computer and Information Science Polytechnic University, Brooklyn, NY 11201
3. Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications Ion Stoicay , Robert Morrisz, David Liben-Nowellz, David R. Kargerz, M. Frans Kaashoekz, Frank Dabekz, Hari Balakrishnanz
4. A survey of Peer-to-Peer Network Security Issues by James Li - <http://www.cse.wustl.edu/~jain/cse571-07/ftp/p2p/#ddos07>
5. <http://www.uniformserver.com/>
6. <http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_7-4/dos_attacks.html>
7. <http://resources.infosecinstitute.com/dos-attacks-free-dos-attacking-tools/>
8. Worldwide Infrastructure Security Report – Arbor Networks
9. <http://peersim.sourceforge.net/tutorial2/tutorial2.pdf>

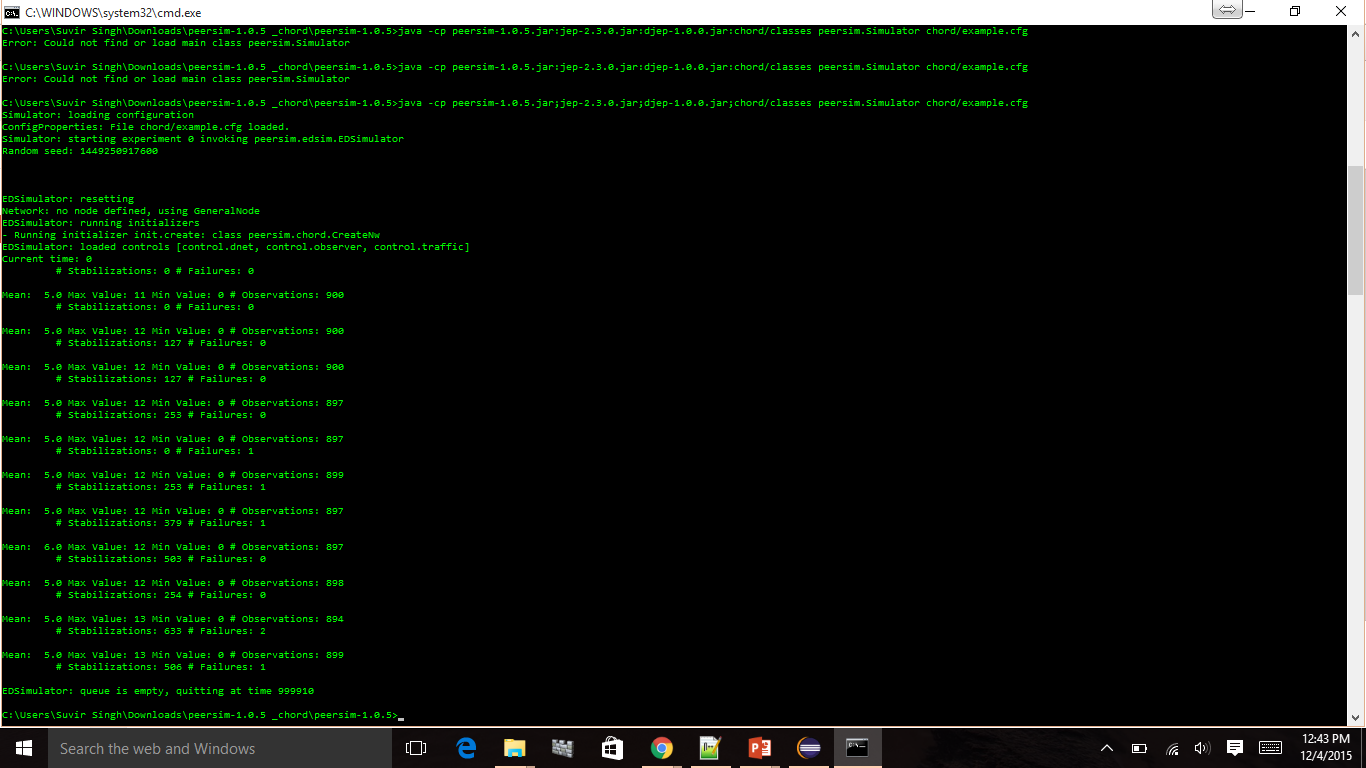
APPENDIX:



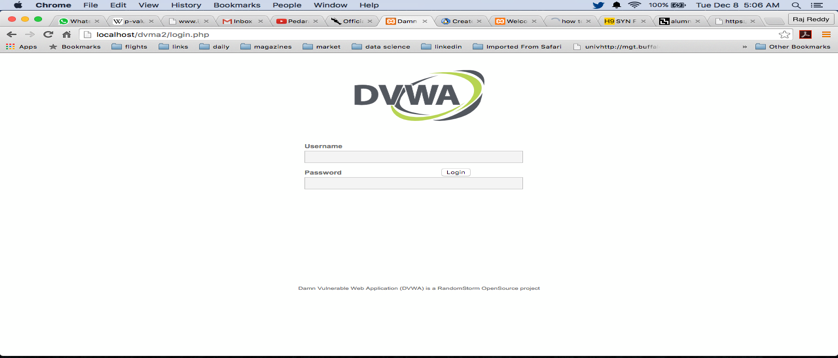
[A1]

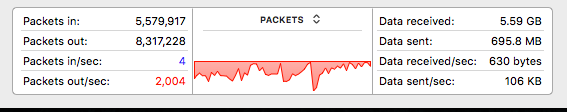


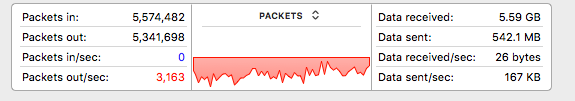
[A2]

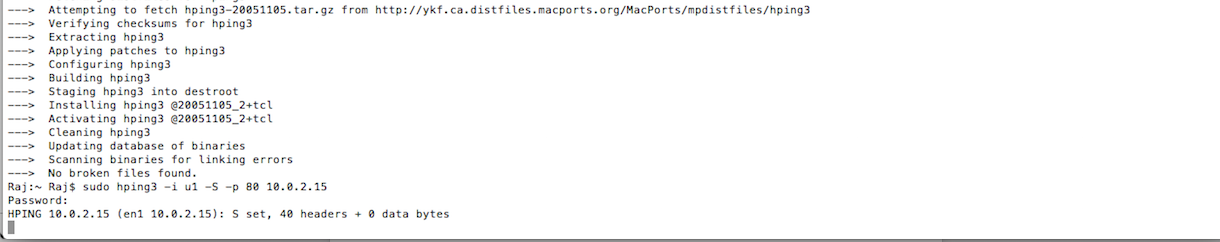
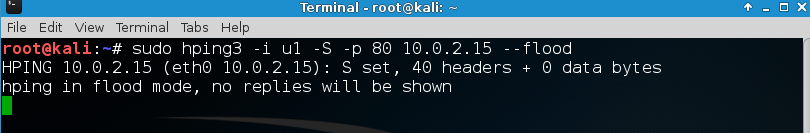


[A3]

[A4]

[A5]

[A6]

[A7] [A8]

[A9]